Arkady Zaslavsky
Zhaolong Ning
Vana Kalogeraki
Dimitrios Georgakopoulos
Panos K. Chrysanthis (Eds.)

594

Mobile and Ubiquitous Systems

Computing, Networking and Services

LNICST

20th EAI International Conference, MobiQuitous 2023
Melbourne, VIC, Australia, November 14–17, 2023
Proceedings, Part II

Part 2

Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 594

Editorial Board Members

Ozgur Akan, *Middle East Technical University, Ankara, Türkiye*
Paolo Bellavista, *University of Bologna, Bologna, Italy*
Jiannong Cao, *Hong Kong Polytechnic University, Hong Kong, China*
Geoffrey Coulson, *Lancaster University, Lancaster, UK*
Falko Dressler, *University of Erlangen, Erlangen, Germany*
Domenico Ferrari, *Università Cattolica Piacenza, Piacenza, Italy*
Mario Gerla, *UCLA, Los Angeles, USA*
Hisashi Kobayashi, *Princeton University, Princeton, USA*
Sergio Palazzo, *University of Catania, Catania, Italy*
Sartaj Sahni, *University of Florida, Gainesville, USA*
Xuemin Shen, *University of Waterloo, Waterloo, Canada*
Mircea Stan, *University of Virginia, Charlottesville, USA*
Xiaohua Jia, *City University of Hong Kong, Kowloon, Hong Kong*
Albert Y. Zomaya, *University of Sydney, Sydney, Australia*

The LNICST series publishes ICST's conferences, symposia and workshops.

LNICST reports state-of-the-art results in areas related to the scope of the Institute. The type of material published includes

- Proceedings (published in time for the respective event)
- Other edited monographs (such as project reports or invited volumes)

LNICST topics span the following areas:

- General Computer Science
- E-Economy
- E-Medicine
- Knowledge Management
- Multimedia
- Operations, Management and Policy
- Social Informatics
- Systems

Arkady Zaslavsky · Zhaolong Ning ·
Vana Kalogeraki · Dimitrios Georgakopoulos ·
Panos K. Chrysanthis
Editors

Mobile and Ubiquitous Systems

Computing, Networking and Services

20th EAI International Conference, MobiQuitous 2023
Melbourne, VIC, Australia, November 14–17, 2023
Proceedings, Part II

Springer

Editors
Arkady Zaslavsky
Deakin University
Melbourne, VIC, Australia

Vana Kalogeraki
Athens University of Economics
and Business
Athens, Greece

Panos K. Chrysanthis
University of Pittsburgh
Pittsburgh, PA, USA

Zhaolong Ning
Chongqing University of Posts
and Telecommunications
Chongqing, China

Dimitrios Georgakopoulos
Swinburne University of Technology
Melbourne, VIC, Australia

ISSN 1867-8211 ISSN 1867-822X (electronic)
Lecture Notes of the Institute for Computer Sciences, Social Informatics
and Telecommunications Engineering
ISBN 978-3-031-63991-3 ISBN 978-3-031-63992-0 (eBook)
https://doi.org/10.1007/978-3-031-63992-0

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

If disposing of this product, please recycle the paper.

Preface

For the first time after the COVID-19 world pandemic, the 20th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (EAI MobiQuitous 2023) was held as an on-site conference with an option to present remotely. The conference took place on November 14–17, 2023, in Melbourne, Australia, at the RMIT University, which acted as a co-sponsor of the event. This year's MobiQuitous was also special as it celebrated its 20th edition as a premier mobile and ubiquitous computing forum. We are delighted to introduce the proceedings, which match MobiQuitous's longstanding tradition of providing unique opportunities for participants, researchers, industry representatives, and students from around the world to explore new ideas, techniques, and tools, and to exchange experiences in all aspects of mobile and ubiquitous computing including systems, applications, social networks, middleware, networking, sensing, IoT, data management, data processing, and services. This year, some emerging themes were federated machine learning, mobile edge and fog computing, mobile augmented reality, and interaction technologies.

This year, 161 research papers were submitted, which were subject to a rigorous review procedure; every paper had three to five independent reviews. 57 full-length papers (i.e., an acceptance rate of 35%) and 4 short papers (i.e., a total acceptance rate of 37%) were selected for publication in the proceedings. The accepted papers were organized into 15 research sessions based on research on tracking systems, Internet of Things (IoT), federated learning, networks, activity recognition, cyber and internet security, urban and mobile crowdsensing, edge computing, crowdsourcing platforms and localizations, activity recognition and prediction, machine learning, mobile edge and fog computing, mobile augmented reality, and applications for mobile computing and interaction technologies. In addition to the research sessions, the program included one workshop and a PhD track event on Day 1. The workshop discussed 5 papers under the theme of *AutoQuitous*.

MobiQuitous 2023 featured two keynote speeches, and three invited presentations. The first keynote speech is titled "*Emerging haptically-enabled systems for immersive simulation-based training*" by Saeid Nahavandi, Distinguished Professor and inaugural Associate Deputy Vice-Chancellor Research and Chief of Defence Innovation at Swinburne University of Technology, Australia. The second keynote speech is titled "*MinerFinder: A GAE-LSTM method for predicting spatio-temporal locations of miners in underground mines*" by Sanjay K. Madria, a Curators' Distinguished Professor in the Department of Computer Science at the Missouri University of Science and Technology, USA. The first invited talk was on *The Online Dynamic Path Planner for UAVs*, submitted by a group of researchers and officially presented by the keynote speaker Sanjay K. Madria. The second one was on *Securing Wireless Communication in Critical Infrastructure: Challenges and Opportunities*, officially submitted by Avinaba Mistry from the University of New Brunswick, Canada. The last one was on *Scalable Contact*

Tracing Over Uncertain Trajectories, submitted and presented by Martin Henze from RWTH Aachen University & Fraunhofer FKIE, Germany.

We owe our sincere gratitude to the members of all the Technical Program Committees who contributed in assembling such a high-quality program, working within a very tight schedule. Several additional people contributed to the successful organization of the MobiQuitous 2023 conference. Special thanks go to the following valued collaborators for their dedication to excellence: Workshop Chairs Gregor Shiele (Duisburg-Essen University, Germany) and Herodotos Herodotou (Cyprus University of Technology, Cyprus); Demonstration Chair Manos Papagelis (York University, Canada); Panel Chairs Michael Sheng (Macquarie University, Australia) and Kyriakos Moraitides (Singapore Management University, Singapore); Tutorials Chairs Xiaojie Wang (Chongqing University of Posts and Telecommunications, China), Dali Kaafar (Macquarie University / CSIRO Data61, Australia) and Demetris Zeinalipour (University of Cyprus, Cyprus); Sponsorship & Exhibits Chairs Wei Xiang (La Trobe University, Australia) and Xu Jianliang (Hong Kong Baptist University, Hong Kong); Posters and PhD Track Chairs Jinho Choi (Deakin University, Australia) and Lina Yao (UNSW, Australia); Publications Chair Saguna Saguna (Lulea University of Technology, Sweden); Web Chair Maria Spichkova (RMIT University, Australia); Publicity and Social Media Chair Azadeh Ghari Neiat (Deakin University, Australia); and Local Chairs Harindu Korala (Monash University, Australia) and Anas Dawod (Swinburne University of Technology, Australia).

We would also like to express our special appreciation to the steering committee: Tao Gu (Macquarie University, Australia, Chair), Takahiro Hara (Osaka University, Japan) and Max Mühlhäuser (Technische Universitat Darmstadt, Germany), and to the European Alliance for Innovation and its conference management team – Martin Hochel, Marica Scevlikova, and Radka Vasileiadis – for the overall support provided throughout the process. Finally, special thanks go to the authors who submitted papers and to the presenters; without their contribution, MobiQuitous 2023 would not be possible.

We hope you will find the papers included in the proceedings interesting and stimulating, and that they will lead to new discoveries, innovations, and collaborations.

Dimitrios Georgakopoulos
Panos Chrysanthis
Arkady Zaslavsky
Zhaolong Ning
Vana Kalogeraki

Organization

Steering Committee

Tao Gu	Macquarie University, Australia
Takahiro Hara	Osaka University, Japan
Max Mühlhäuser	Technische Universitat Darmstadt, Germany

Organizing Committee

General Chairs

Dimitrios Georgakopoulos	Swinburne University of Technology, Australia
Panos K. Chrysanthis	University of Pittsburgh, USA

TPC Chairs and Co-chairs

Arkady Zaslavsky	Deakin University, Australia
Zhaolong Ning	Chongqing University of Posts and Telecommunications, China
Vana Kalogeraki	Athens University of Economics and Business, Greece

Sponsorship and Exhibit Chairs

Wei Xiang	La Trobe University, Australia
Xu Jianliang	Hong Kong Baptist University, China

Local Chairs

Harindu Korala	Monash University, Australia
Anas Dawod	Swinburne University of Technology, Australia

Workshops Chairs

Gregor Shiele	Duisburg-Essen University, Germany
Herodotos Herodotou	Cyprus University of Technology, Cyprus

Publicity and Social Media Chair

Azadeh Ghari Neiat	Deakin University, Australia

Publications Chair

Saguna Saguna	Luleå University of Technology, Sweden

Web Chair

Maria Spichkova	RMIT University, Australia

Posters and PhD Track Chairs

Jinho Choi	Deakin University, Australia
Lina Yao	UNSW Sydney, Australia

Panels Chairs

Michael Sheng	Macquarie University, Australia
Kyriakos Mouratidis	Singapore Management University, Singapore

Demos Chair

Manos Papagelis	York University, Canada

Tutorials Chairs

Xiaojie Wang	Chongqing University of Posts and Telecommunications, China
Dali Kaafar	Macquarie University/CSIRO Data61, Australia
Demetris Zeinalipour	University of Cyprus, Cyprus

Technical Program Committee

Rakib Abdur	Coventry University, UK
Christian Becker	University of Stuttgart, Germany
Nikos Bikakis	HMU & Athena Research Center, Greece
Cristian Borcea	New Jersey Institute of Technology, USA
Kevin Matthe Caramancion	University of Wisconsin-Stout, USA

Zhichao Cao	Michigan State University, USA
Jagmohan Chauhan	University of Southampton, UK
Junxin Chen	Dalian University of Technology, China
Tao Chen	University of Pittsburgh, USA
Maria Luisa Damiani	University of Milan, Italy
Klaus David	University of Kassel, Germany
Pari Delir Haghighi	Monash University, Australia
Alexis Delis	University of Athens, Greece
Christos Efstratiou	University of Kent, UK
Xiaoran Fan	Google, USA
Stefan Fischer	University of Lübeck, Germany
Francesco Flammini	Mälardalen University, Sweden
Thippa Reddy Gadekallu	Vellore Institute of Technology, India
Honghao Gao	Gachon University, South Korea
Akshay Gadre	Carnegie Mellon University, USA
Jean Philippe Georges	Université de Lorraine, France
Giorgos Giannopoulos	Athena Research Center, Greece
Voncenzo Gulis	Chalmers University, Sweden
Xiuzhen Guo	Tsinghua University, China
Stathes Hadjiefthymiades	University of Athens, Greece
Armin Haller	Australian National University, Australia
Takahiro Hara	Osaka University, Japan
Martin Henze	RWTH Aachen University, Germany & Fraunhofer FKIE, Germany
Prem Prakash Jayaraman	Swinburne University, Australia
Beihong Jin	Institute of Software, Chinese Academy of Sciences, China
Salil Kanhere	UNSW, Australia
Sylvain Kubler	University of Luxembourg, Luxembourg
Satoshi Kurihara	Keio University, Japan
Brano Kusy	CSIRO, Australia
Brent Lagesse	University of Washington Bothell, USA
Guohao Lan	Delft University of Technology, Netherlands
Stephen Lee	University of Pittsburgh, USA
Kevin Lee	Deakin University, Australia
Wang-Chien Lee	Pennsylvania State University, USA
Chi Lin	Dalian University of Technology, China
Sicong Liu	Northwestern Polytechnical University, China
Sanjay Madria	Missouri University of Science and Technology, USA
Sajib Mistry	Curtin University, Australia
Karan Mitra	Luleå University of Technology, Sweden

Valeh Moghaddam	Deakin University, Australia
Mohamed Mokbel	University of Minnesota, USA
Manzur Murshed	Deakin University, Australia
Amiya Nayak	University of Ottawa, Canada
Anne Ngu	Texas State University, USA
VP Nguyen	University of Texas at Arlington, USA
Andrea Passarella	IIT-CNR, Italy
Tomasz Pieciak	Universidad de Valladolid, Spain
Evaggelia Pitoura	University of Ioannina, Greece
Suprio Ray	University of New Brunswick, USA
George Roussos	Birkbeck, University of London, UK
Stephan Sigg	Aalto University, Finland
Amit Kumar Singh	National Institute of Technology Patna, India
Dimitris Skoutas	Athena Research Center, Greece
Dimitris Spathis	Bell Labs, USA
Sanjib Sur	University of South Carolina, USA
Gautam Srivastava	Brandon University, Canada
Chiu Tan	Temple University, USA
Mohammad Tanveer	Indian Institute of Technology Indore, India
Zahir Tari	RMIT University, Australia
My Thai	University of Florida, USA
Arm Tolba	King Saud University, Saudi Arabia
Dimitrios Tomaras	Athens University of Economics and Business, Greece
Jeffrey Tweedale	Defence Science and Technology Group, Australia
Zhongnan Qu	ETH Zurich, Switzerland
Ju Wang	Samsung AI Center-Montreal, Canada
Minmei Wang	University of Connecticut, USA
Chenshu Wu	University of Hong Kong, China
Marcin Woźniak	Silesian University of Technology, Poland
Yaxiong Xie	Princeton University, USA
Zongxing Xie	Stony Brook University, USA
Jingao Xu	Tsinghua University, China
George Xylomenos	Athens University of Economics and Business, Greece
Lei Yang	Hong Kong Polytechnic University, China
Vladimir Zadorozhny	University of Pittsburgh, USA
Fusang Zhang	Institute of Software, Chinese Academy of Sciences, China
Leo Zhang	Griffith University, Australia

Xiaolong Zheng	Beijing University of Posts and Telecommunications, China
Yuanqinq Zheng	Hong Kong Polytechnic University, China
Yanmin Zhu	Shanghai Jiao Tong University, China

Contents – Part II

Urban/Mobile Crowdsensing

A Data-Driven Study on Pedestrian Walking Behaviour as Transitioning Different Spaces 3
Mengnan Cai and Xinling Shen

Standardizing Your Training Process for Human Activity Recognition Models – A Comprehensive Review in the Tunable Factors 15
Yiran Huang, Haibin Zhao, Yexu Zhou, Till Riedel, and Michael Beigl

MyPath: Accessible Route Generation Using Crowd-Sensed Surface Information 28
Thomas Nguyen, Md Fourkanul Islam, Rochishnu Banerjee, Hanna M. Noyce, Emily M. Olejniczak, Roger O. Smith, Md Osman Gani, and Vaskar Raychoudhury

Edge Computing

`Shepard`: Dynamic Placement of Microservices in the Edge-Cloud Continuum 43
Farhan Asghar, Tehreem Fatima, Junaid Haroon Siddiqui, Naveed Anwar Bhatti, and Muhammad Hamad Alizai

Minimizing Data Retrieval Delay in Edge Computing 63
Kolichala Rajashekar, Souradyuti Paul, Sushanta Karmakar, and Subhajit Sidhanta

Bringing the Edge Home: Edge Computing in the Era of Emerging WLANs . . . 86
Sampath Edirisinghe and Chathurika Ranaweera

Crowdsourcing, Platforms and Localization

A Novel Task Assignment Adjustment Method in Spatial-Temporal Crowdsourcing 105
Bingyi Sun, Jiaxu Cui, Hongtao Bai, and Yonggang Zhang

PPS: A Publish-Process-Subscribe Middleware for Predictive Supply Chains 125
Amir Jabbari, Gowri Ramachandran, Sidra Malik, and Raja Jurdak

Gestalt Perceptual Calibration for Multi-agent Collaborative Localization 145
Yan Zhang and Rong Xie

Activity Recognition and Prediction

Traffic Flow Prediction Using Uber Movement Data 167
Daniele Cenni and Qi Han

Fingertip Air-Writing with Ambient Light 179
Hao Liu, Hanting Ye, Xiangxie Zhang, Jie Yang, and Qing Wang

Global Optimal PnP Algorithm with Reverse Projection Error 192
Xuhao Jin

Proposal of Word Prediction Method for Gaze Swipe Text Entry 204
Hisanori Fujii and Ryo Katsuma

AI and Machine Learning

Integrating Social Environment in Machine Learning Model for Debiased Recommendation 219
Yihong Zhang, Lina Yao, and Takahiro Hara

A Novel Deep Learning Model for Smartphone-Based Human Activity Recognition 231
Nadia Agti, Lyazid Sabri, Okba Kazar, and Abdelghani Chibani

AIMSafe: EEG-Based Driver Behavior Understanding via Attention and Incremental Learning Mechanisms 244
Landu Jiang, Cheng Luo, Tao Gu, Kezhong Lu, and Dian Zhang

A Multi-task Learning Framework with Features Based on Behavioral Pattern Conversation 256
Bo Tang, Nan Wang, Jinbao Li, and Zhonghui Shen

Efficient Parcel Damage Detection via Faster R-CNN: A Deep Learning Approach for Logistical Parcels' Automated Inspection 268
Zhi Chen, Cuifeng Du, Quanlong Guan, Yuyu Zhou, Vichen Hoo, Xiujie Huang, Zhefu Li, Shuanghuan Lv, Xiaofeng Wu, and Xiaotian Zhuang

Fall Detection and Assessment Using Multitask Learning and Micro-sized LiDAR in Elderly Care 280
Shota Yamada, Hamada Rizk, Tatsuya Amano, and Hirozumi Yamaguchi

Online Dynamic Path Planner for UAVs 294
Manisha Wadhwa, Neelima Gupta, and Sanjay Madria

Mobile Edge and Fog Computing

Achieving Observability on Fog Computing with the Use of Open-Source Tools 319
Breno Costa, Abhik Banerjee, Prem Prakash Jayaraman, Leonardo R. Carvalho, João Bachiega Jr., and Aleteia Araujo

Adaptive QoS-Aware Task Offloading in Dynamic Mobile Edge Computing Environment 341
Jacob Don, Sajib Mistry, Redowan Mahmud, and Aneesh Krishna

Discretization-Based Ensemble Model for Robust Learning in IoT 353
Anahita Namvar, Chandra Thapa, and Salil S. Kanhere

Flute: Enabling a Battery-Free and Energy Harvesting Ecosystem for the Internet of Things 368
Van Vu Bui, Shuaibu Musa Adam, Brendan J. Mackenzie, Ashok Samraj Thangarajan, Mengyao Liu, Sam Michiels, Nelson Souto Rosa, Huynh Nguyen Bao Phuong, and Danny Hughes

Exploring Optimal Placement of Head-Based Hierarchical Marking Menus on Smartphones 381
Sabiha Tahsin Soha, Aritra Mazumder, Forhad Rabbi, Farida Chowdhury, and Khalad Hasan

SDN-DMQTT: SDN-Based Platform for Re-configurable MQTT Distributed Brokers Architecture 393
Fatma Hmissi and Sofiane Ouni

Mobile Augmented Reality and Applications for Mobile Computing

Quality Evaluation of Image Segmentation in Mobile Augmented Reality 415
Shneka Muthu Kumara Swamy and Qi Han

Reading the Mind's Eye: Detecting Trauma-Vulnerability in Individuals by Analyzing Attention Through Eye-Tracking 428
Taseen Mubassira, Mehedi Hasan, and Sadia Sharmin

GUI2DSVec: Detecting Visual Design Smells Based on Semantic Embedding of GUI Images and Components 445
Bo Yang and Shanping Li

Interaction Technologies

Mobile User Experience from the Lens of Project-Based Learning 461
Maria Spichkova

Usability Evaluation of NOW Interactions 473
Rosaline Barendregt, Barbara Wasson, and Martin Heitmann

QA Reasoning Enhancement Model Based on the Fusion of Dictionary and Hierarchical Directed Graph 485
Yuhang Bie, Meiling Liu, and Jiyun Zhou

AutoQuitous Workshop Session 1

Towards Distributed Control Under Deficient Communication with Multi-agent Reinforcement Learning 503
Arne Kummerow and Torben Weis

Deep Learning Based Radio Frequency Fingerprint Identification by Exploiting Spatial Stereoscopic Features 511
Shunliang Zhang, Jing Li, and Xiaolei Guo

Modeling Emergent Behaviour for Enhanced Autonomy in Cyber-Physical Systems 520
Matthias Schaffeld and Torben Weis

On-Device Soft Sensors: Real-Time Fluid Flow Estimation from Level Sensor Data 529
Tianheng Ling, Chao Qian, and Gregor Schiele

Cultural Influence on Autonomous Vehicles Acceptance 538
Chowdhury Shahriar Muzammel, Maria Spichkova, and James Harland

Author Index 549

Contents – Part I

Tracking and Detection

NOD: Lightweight Continuous Neighbor Discovery on Everyday Devices 3
Hsiao-Yuan Chen, Evan King, and Christine Julien

Decentralized Collaborative Inertial Tracking 26
Alpha Diallo and Benoît Garbinato

High-Performance Features in Generalizable Fingerprint-Based Indoor Positioning 46
Andrea Brunello, Angelo Montanari, Nicola Saccomanno, and Joaquín Torres-Sospedra

SCORE: Scalable Contact Tracing over Uncertain Trajectories 68
Avinaba Mistry, Xichen Zhang, Suprio Ray, and Sanjeev Seahra

IoT

FaultBit: Generic and Efficient Wireless Fault Detection Using the Internet of Things 97
Koustabh Dolui, Ashok Samraj Thangarajan, Sergii Morshchavka, Zhaoyi Liu, Sam Michiels, and Danny Hughes

DeepHeteroIoT: Deep Local and Global Learning over Heterogeneous IoT Sensor Data 119
Muhammad Sakib Khan Inan, Kewen Liao, Haifeng Shen, Prem Prakash Jayaraman, Dimitrios Georgakopoulos, and Ming Jian Tang

Federated Reinforcement Learning for Automated LoRaWAN Management in Industrial IoT 136
Ameer Ivoghlian, Zoran Salcic, and Kevin I-Kai Wang

A Hybrid Approach to Monitor Context Parameters for Optimising Caching for Context-Aware IoT Applications 156
Ashish Manchanda, Prem Prakash Jayaraman, Abhik Banerjee, Arkady Zaslavsky, Shakthi Weerasinghe, and Guang-Li Huang

LOADHoC: Towards the Automatic Local Distribution of Computation Using Existing IoT Devices 175
Shaine Christmas, Kevin Lee, and Jean-Guy Schneider

E-Go Bicycle Intelligent Speed Adaptation System for Catching the Green Light 201
Khalil Ben Fredj, Akhil Reddy Pallamreddy, Geert Heijenk, Paul Havinga, and Yanqiu Huang

Federated Learning

FedGCS: Addressing Class Imbalance in Long-Tail Federated Learning 223
Guozheng Liu, Wei Zhang, Huiling Shi, Lizhuang Tan, Chang Tang, and Meihong Yang

FedRC: Representational Consistency Guided Model Uploading Mechanism for Asynchronous Federated Learning 239
Sheng Liu, Linlin You, and Yuren Zhou

RADEAN: A Resource Allocation Model Based on Deep Reinforcement Learning and Generative Adversarial Networks in Edge Computing 257
Zhaoyang Yu, Sinong Zhao, Tongtong Su, Wenwen Liu, Xiaoguang Liu, Gang Wang, Zehua Wang, and Victor C. M. Leung

Networks

A Stream Data Service Framework for Real-Time Vehicle Companion Discovery 281
Zhongmei Zhang and Shuai Zhang

KS-Autoformer: An Autoformer-Based SOC Prediction Framework for Electric Vehicles 297
Yaoyidi Wang, Niansheng Chen, Lei Rao, Dingyu Yang, Guangyu Fan, Songlin Cheng, and Xiaoyong Song

Deep Reinforcement Learning-Based Multi-node Collaborative Task Offloading Optimization in 6G Space-Air-Ground Integrated Networks 313
Wang Li, Xin Chen, Libo Jiao, Wangzhong Ning, and Wenwu Zhu

Securing Wireless Communication in Critical Infrastructure: Challenges and Opportunities 333
Jörn Bodenhausen, Christian Sorgatz, Thomas Vogt, Kolja Grafflage, Sebastian Rötzel, Michael Rademacher, and Martin Henze

Activity Recognition

Cross-User Activity Recognition via Temporal Relation Optimal Transport 355
Xiaozhou Ye and Kevin I-Kai Wang

SelfAct: Personalized Activity Recognition Based on Self-Supervised and Active Learning 375
Luca Arrotta, Gabriele Civitarese, and Claudio Bettini

A Novel Method for Wearable Activity Recognition with Feature Evolvable Streams 392
Yixiao Wang, Chunyu Hu, Hong Liu, Lei Lyu, and Lin Yuan

Let's Vibrate with Vibration: Augmenting Structural Engineering with Low-Cost Vibration Sensing 408
Masfiqur Rahaman, Md. Nazmul Hasan Sakib, Nafisa Islam, Saiful Islam Salim, Uday Kamal, Raihan Rasheed, and A. B. M. Alim Al Islam

Security Management

Research on Data Drift and Class Imbalance in Android Malware Detection ... 429
Zhen Liu, Ruoyu Wang, Bitao Peng, Changji Wang, and Qingqing Gan

Reputation-Based Dissemination of Trustworthy Information in VANETs 445
Vincenzo Agate, Alessandra De Paola, Giuseppe Lo Re, and Antonio Virga

Reputation Systems for Supply Chains: The Challenge of Achieving Privacy Preservation 464
Lennart Bader, Jan Pennekamp, Emildeon Thevaraj, Maria Spiß, Salil S. Kanhere, and Klaus Wehrle

Data Management in Appendable-Block Blockchains: A Case Study for IT Life-Cycle Management 476
Rodrigo W. Silveira, Roben C. Lunardi, and Avelino F. Zorzo

Exploiting the Potential Anomaly Detection in Automobile Safety Data with Multi-type Neural Network 489
Quanlong Guan, Tian Zhang, Xiujie Huang, Yuansheng Zhong, Cuifeng Du, Changjiang Liu, Zhefu Li, Guanghui Zhang, Xiaofeng Wu, and Zhifei Duan

Urban/Mobile Crowdsensing

HAUM^3: A Height Aware Urban Map Matching Mechanism 505
Jie Tang, Sunjian Zheng, Bo Yu, and Shaoshan Liu

A Resource-Efficient Approach of GNSS Activation for Pedestrian Monitoring 526
Asiimwe Paddy Junior, Luis Enrique Díez, Alfonso Bahillo, and Odongo Steven Eyobu

Author Index 547

Urban/Mobile Crowdsensing

A Data-Driven Study on Pedestrian Walking Behaviour as Transitioning Different Spaces

Mengnan Cai(✉) and Xinling Shen

University of Melbourne, Melbourne, VIC 3010, Australia
{mengnanc,xinlings}@student.unimelb.edu.au

Abstract. Pedestrians' daily walking behaviours involve transitioning urban spaces with different functions in the pedestrian transportation. Current research has extensively investigated different pedestrian walking behaviours. However, beyond the influence of physical disturbances and structure on pedestrian behaviour, current research lacks an understanding of the impact and role of space itself on pedestrian behaviour from the perspective of spatial changes, and similarities in the behaviour of pedestrians as they transition different spaces. This study uses real-world datasets to conduct a data-driven study on the impact of spaces (open space, sidewalk, and crossing) on pedestrian walking speed. The results show that, pedestrian speed changes statistically significantly near the connection line between two spaces. The location of this significant change in speed depends on the type of connected space. Also, the magnitude of speed change in this area depends on the type of adjacent space and the movement direction. We also study the extent to which the turning behaviour of pedestrians as they transition in different directions of movement affects the speed. Furthermore, We investigate the correlation between the rate of speed change as pedestrians transition in different directions and through different spaces. We regard our results can benefit the realistic pedestrian walking simulations, behaviour prediction, and speed based knowledge.

Keywords: Walking Behaviour · Different Spaces for Urban Traffic · Pedestrian Walking Behaviour Knowledge

1 Introduction

Pedestrians exhibit a wide range of walking behaviours in complex urban traffic. Considering the relationship and influence between pedestrians and spaces has become an important topic in urban research, practice, and application [20], as researchers seek to better understand the complex interactions between pedestrians and space especially when influences change. Studying the link between real-world pedestrian behaviour and space is crucial since it enhances the realism in modeling pedestrian behaviour in scenarios such as Open-World Games [17], and refines predictive accuracy in pedestrian behaviour [8]. Additionally, it offers

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 3–14, 2024.
https://doi.org/10.1007/978-3-031-63992-0_1

valuable perspectives in social and behavioural sciences, aiding in understanding human responses to the built environment [5].

In daily walking, pedestrians often transition various urban outdoor spaces which have different functional uses or purposes for which land is designed to be used [7], such as crossings, sidewalks, and open spaces. Current literature has listed the factors influencing pedestrian movement in spaces [15] and general behaviours in different spaces [6]. However, there is a knowledge gap regarding how pedestrians spontaneously adapt their behaviours and habits in transitioning between connected spaces, as well as in understanding the differences and commonalities in pedestrian behaviour within mixed traffic and exclusively pedestrian spaces, which vary in safety awareness levels [14].

In particular, we ask: is there a significant difference in the speed of pedestrians when they transition different spaces? If so, where do these changes usually occur? Are there any similarities between these changes in different spaces and movement directions? Given the above context, we use two real-world datasets [13,22] which recorded pedestrian trajectories when transitioning spaces to investigate and analyze the walking speed, with speed being considered an important reference for most walking behaviour simulation models [18]. We elucidate the effects of urban space transitions on pedestrian walking speed, which helps to better understand pedestrian walking habits. The expected contributions are the understanding of spontaneous walking speed as pedestrians transition different spaces, and the understanding of the similarities and differences of speed trends when transitioning different connected spaces and in different movement direction.

2 Background Knowledge and Related Works

In this study, the urban spaces focus on are outdoor public lands and built environment on which they are designed for different transport purposes [2]. The spaces in this research are public outdoor spaces between buildings which are represented by transport infrastructures [4], and pedestrian-oriented with a pedestrian scale. In this research, We consider open space, crossing, and sidewalk which are generally considered common separable urban public spaces as our scope.

Pedestrian behaviour, influenced by environmental and crowd factors, is often studied through simulation models [1]. Various methods have been used to analyze walking behaviours in different settings, utilizing kinematic equations and behavioural rules [12]. Although behaviour-based investigations are emerging, the application of space-pedestrian relationship knowledge in modeling remains limited, often focusing on specific structures or environmental factors [3]. Further research is needed to understand pedestrian walking habits across different spaces and the impact of spatial variation on walking speed.

In addition to modelling, the pedestrian-space relationships are crucial for trajectory prediction models used in applications such as autonomous vehicles [21]. These models incorporate space as a key feature to predict navigation and pedestrian-space interactions [10]. Despite their potential, challenges of computational intensity and data scarcity make it hard to balance model complexity and

prediction accuracy [23]. Knowledge of spatially relevant spontaneous behaviour can enhance model training and improve accuracy. Further, while space significantly influences walking speed [19], more research is needed to understand speed variations during space transitions, particularly between mixed traffic and pedestrian spaces.

3 Methods

We investigate pedestrian speeds when transitioning through: (1) in an entirely pedestrian space consisting of sidewalk and open space orthogonally connected, and (2) in a mixed traffic space consisting of open space, crossing and sidewalk connections.

We emphasize the assumptions in this study, as follows:

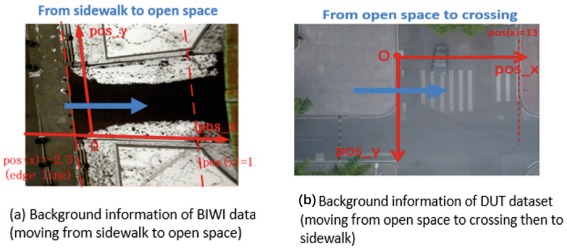
- All pedestrians in our study behave in similar way and have the same perception of the space. We do not consider specific individual behaviours.
- Possible interactive and conflicting behaviour between pedestrians does not have a significant impact on speed and have been ignored [9].
- Frictional resistance of pavement construction materials is similar. Possible ground landscaping and differences in pavement height do not have a significant impact on pedestrian speeds and have been ignored.
- The curb between a sidewalk and mixed traffic has a temporary and minor impact on speed, primarily due to the low curbstone height.
- Our study is on pedestrians who walk for commuting purposes, excluding those who walk to the scenario edge.

3.1 Dataset Description

To study how walking speed varies across spaces, datasets with human trajectory data and spatial information for at least two different spaces are needed. The BIWI Walking Pedestrians Dataset (BIWI) and Vehicle-Crowd Interaction - DUT Dataset (DUT) are two popular choices whose backgrounds are shown in Fig. 1, both featuring at least two types of spaces and reflecting low to medium pedestrian density scenarios [9].

BIWI Dataset: This dataset captures pedestrian movements in front of ETH Zurich's main building, providing a bird's eye view [13]. Although there is another scenario in this dataset (seq_hotel), it only features one space type and is thus excluded. The BIWI dataset includes an open space and a sidewalk, where pedestrians mainly commute between. Outside the sidewalk is a mixed-traffic space.

DUT Dataset: This dataset features everyday campus scenarios at Dalian University of Technology, focusing on an intersection scenario [22]. It includes open space, non-signal crossing, and a sidewalk, where the pedestrian flows mainly between. Some scenarios (1–3, 4, 6, 10–14, 16, and 17) are excluded due to low trajectory numbers, vehicle interference, group gatherings.



(a) Background information of BIWI data (moving from sidewalk to open space)

(b) Background information of DUT dataset (moving from open space to crossing then to sidewalk)

Fig. 1. Background information of BIWI dataset and DUT dataset (with moving directions).

3.2 Data Processing

Our data processing includes two main steps:

1. Data Cleaning: We pre-process the data to eliminate outliers, retaining only trajectories within three standard deviations of the mean speed, based on a normal distribution estimation.
2. Trajectory Filtering: To analyze speed variations across different spaces, we ensure the trajectories traverse at least two spaces:
   - Retain trajectories with endpoints on opposite sides of the edge line between two spaces.
   - For trajectories with endpoints on the same side of the edge line, split and retain them if they cross the edge line.
   - Exclude trajectories if the perpendicular lengths of both endpoints to the edge line are less than twice the average step length, ensuring only deliberate crossings with a slight crossing angle are retained.
   - In the DUT dataset, due to a parked car on the crossing in some scenarios, retain only trajectories below the parked vehicle.

The filtered trajectory data is summarized in Table 1. Acknowledging that turning affects speed due to human joint kinematics [11], we categorize BIWI dataset pedestrians into turning and non-turning groups.

Table 1. Trajectory information after filter

	Datasets		
	BIWI dataset		DUT Dataset
	From sidewalk to open space	From open space to sidewalk	From open space to crossing, sidewalk
Speed Standard Deviation (m/s)	0.29	0.29	0.25
Speed Mean Value (m/s)	1.46	1.46	1.21
Speed Range (m/s)	(0.59, 2.33)	(0.59, 2.33)	(0.45, 1.96)
Number of total pedestrian ids	184	140	405
Number of filtered pedestrian ids	82	43	105
Number of trajectories with turning behaviour	21	16	–
Number of trajectories without turning behaviour	61	27	–

4 Results and Discussion

4.1 BIWI Dataset

BIWI Dataset Results. We use a mixed-effects regression analysis to model the relationship between space and speed, while accounting for the presence of clustered or correlated data. Here, in the regression model, we include the pedestrian identifies as a random effect to account for the difference in initial speeds and comfortable speeds among pedestrians. The parameters in the mixed effect regression analysis are listed in Table 2.

We notice that the p-values are almost 0 for both moving directions in Table 2 which makes us confident that the differences in speed versus positions is not by chance, but significantly related.

Table 2. Mixed effect regression analysis results of BIWI Walking Pedestrian Dataset and DUT dataset (with moving directions)

Datasets	Direction	Stages	Mean speed change	Mean standard deviation change (m/s)	Estimated change of mean speed within SVA
BIWI dataset	Sidewalk to open space	Sidewalk to SVA	−1.86%	0.04	−5.77%
		SVA to open space	6.74%	0.03	14.08%
	Open space to sidewalk	Open space to SVA	−2.05%	0.06	−6.95%
		SVA to sidewalk	−2.37%	−0.06	10.35%
DUT dataset	Open space to crossing then to sidewalk	Open space to SVA1	−11.82%	0.05	−18.54%
		SVA1 to crossing	9.01%	−0.08	14.29%
		Crossing to SVA2	−2.47%	−0.00	−3.54%

Non-turning group We first analyse the data in the direction from sidewalk to open space in BIWI dataset.

To eliminate brief, unrepresentative fluctuations while preserving detail and capturing the real walking patterns of pedestrians, we calculate the mean and standard deviation of speed in intervals defined by an average stride length of 1 m [16]. We then plot upper and lower bounds of a 90% confidence interval around the mean speed, using a Z value of 1.645. Although our observations are made within this confidence interval, it is important to note that our statistical analyses are based directly on the pre-processed data, not within this interval.

Since there are only a limited four data records in the initial interval from −7.76 m to −6.24 m in Fig. 2(a), we exclude it to maintain data reliability. Aligning position(x) with real-world locations based on the dataset, where the edge line is at about −2.3, we distinguish the sidewalk (less than −2.3) from the open space (greater than −2.3), setting the stage to share our findings:

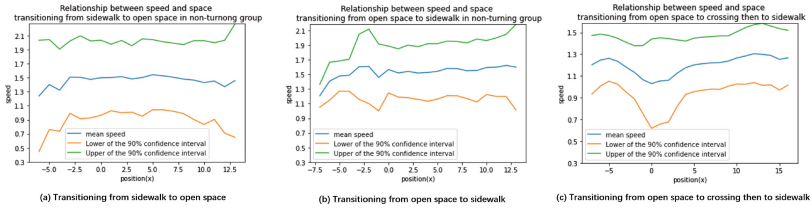


Fig. 2. Relationship between position(x) (meter) and speed (m/s) in filtered trajectories using mean value and standard deviation.

- As the pedestrians transition from the sidewalk to the open space, their originally rising speed changes before the edge line. In the area from approximate -5m to -3m, the mean speed curve and 90 percent confidence interval both show a near V-shaped change.
- When pedestrians approaching the exit of the open space on the right, the area after approximate 11m in 90 percent confidence interval is becoming wider indicating there is a high degree of variability or uncertainty in the overall speed estimates, meaning that the difference in speed between pedestrians becomes greater here, with some accelerating and others decelerating.

Analyzing the BIWI dataset for pedestrians moving from open space to sidewalk (Fig. 2(b), right to left), we observe a shift from near-stable speeds to a V-shaped change in the mean speed and 90% confidence interval between 0 m and −2.3 m. Specifically, the upper confidence interval stays constant from −1 m to 0 m, while the lower drops significantly, indicating a predominant deceleration with few maintaining or increasing their speed.

Our analysis of the mean value curve and 90% confidence interval curves in two reverse directions in the BIWI dataset leads to the following findings: As the pedestrians transition between open space and sidewalk, their speed shows a considerable change in short prior to the edge line between the two spaces.

We refer to the locations where the observed near V-shaped speed changes as the Speed Variation Area (SVA). Next, we explore the extent to which the speed within the SVA has changed from the speed in the area before it and whether there is a significant difference.

To understand speed fluctuations in the SVA, we categorize pedestrian data into stages—sidewalk, SVA, and open space—based on their coordinates. We compute and summarize the average speed changes and their variations across these stages, as shown in Table 3. This analysis highlights the average speed differences between the SVA and adjacent areas, as well as the speed increase and decrease ratios within the SVA, correlating with the key points in the V-shaped speed curve in Fig. 2.

In the BIWI dataset, pedestrians consistently slow down when entering the SVA, regardless of movement direction. However, the magnitude of speed change

Table 3. The extent to which the speed within the SVA varies for different spaces of pedestrian transition (in BIWI dataset and DUT dataset)

	Direction	Stages	Mean speed change	Mean standard deviation change (m/s)	Estimated change of mean speed within SVA
BIWI dataset	Sidewalk to open space	Sidewalk to SVA	−1.86%	0.04	−5.77%
		SVA to open space	6.74%	0.03	14.08%
	Open space to sidewalk	Open space to SVA	−2.05%	0.06	−6.95%
		SVA to sidewalk	−2.37%	−0.06	10.35%
DUT dataset	Open space to crossing then to sidewalk	Open space to SVA1	−11.82%	0.05	−18.54%
		SVA1 to crossing	9.01%	−0.08	14.29%
		Crossing to SVA2	−2.47%	−0.00	−3.54%

as they leave the SVA and transition to the next space varies significantly, highlighting a clear link between average speed change and movement direction. Variations in the 'Estimated change of mean speed within SVA' also underscore that speed changes within the SVA are dependent on the direction of movement.

In the case of ignoring the edge line between the open space and the building, the SVAs range include:

- From sidewalk to open space, approximate: $-5 < \text{position(x)} < -3$;
- From open space to sidewalk, approximate: $-2.3 < \text{position(x)} < 0$

To verify that the speed changes in the SVA are statistically significant, we perform Welch's t-tests to compare speeds within the SVA and the previous area for pedestrians transitioning between sidewalk and open space in both directions. Using a bootstrapping strategy with a sample size of 26 pedestrian identities-determined by a prior power analysis-and repeating the process ten times, we obtain average p-values of 0.045 for transitions from sidewalk to open space, and 0.018 for the reverse. These results validate the statistical significance of the observed speed changes in the SVA, affirming a consistent shift in pedestrian speed just before reaching the edge line, regardless of the direction of transition.

Turning Group. To analyze the impact of turning on pedestrian speed within right-angle spatial connections, we compared turning group speeds during turns to their steady speeds. We define turning thresholds using the standard deviation of heading changes in open space for each individual, identifying the start of a turn when position(x) is less than 0 and the heading change exceeds this threshold, and marking the end when the heading change fell below it. The average speeds and speed standard deviations for the pre-turn, during-turn, and post-turn stages are then calculated and presented in Table 4.

It can be seen that pedestrians generally slow down when turning due to the layout of the space. Notably, moving from sidewalk to open space, they hardly accelerate post-turn, showing only a 0.12% speed change- a stark contrast to the evident speed increase seen when transitioning from open space to sidewalk.

Table 4. The extent to which the speed of pedestrians changes pre, during and post turning in different directions in the BIWI dataset

Direction	Stage	Mean speed(m/s)	Standard deviation(m/s)	Mean speed change
Open space to sidewalk	pre-turn	1.53	0.23	–
	during turn	1.48	0.28	−3.26%
	post-turn	1.55	0.14	4.72%
Sidewalk to open space	pre-turn	1.67	0.20	–
	during turn	1.57	0.20	−5.98%
	post-turn	1.58	0.22	0.12%

BIWI Dataset Discussion. The mixed traffic space next to the sidewalk, described in the dataset's spatial background, might alter pedestrians' walking speeds, leading to patterns different from typical pavement walking. Despite this, our analysis in Fig. 2 shows a pronounced V-shaped change in speed within the Speed Variation Area (SVA), highlighting pedestrians' speed adjustments in response to the space transition, regardless of external influences from the mixed traffic space. This phenomenon is commonly observed in urban areas where sidewalks intersect orthogonally with open spaces and mixed traffic spaces.

We investigate the effects of external factors and space connectivity methods on pedestrian speed by comparing two groups of pedestrians. Particular attention is paid to the need to turn when walking from the sidewalk to an open space. The turning group exhibits an average speed of 1.67 m/s when non-turning on the sidewalk. Their average speed is slightly higher than the non-turning group. To rule out this potential effect, we calculate the relative difference between the average speeds of the two groups of pedestrians in the same open space area, which turned out to be 0.07. This indicates that the potential average speed of non-turning pedestrians not influenced by external factors is about 1.56 m/s. The average speed of the non-turning pedestrians is about 1.56 m/s. The average speed of the non-turning pedestrians is about 1.56 m/s. Compared to the average speed of pedestrians affected by external influences (1.45 m/s), we can conclude that the sidewalk external space could potentially slow down pedestrians by about 7%. Note that our analysis is limited to scenarios with sidewalks orthogonally connected to open spaces with mixed traffic areas outside the sidewalk. We didn't investigate speed variations near the edge line in scenarios with different sidewalk connections or stable pedestrian sources unaffected by external factors.

The potential reason for the obvious difference in the extent of speed change when pedestrians transition from the SVA to the next space in a non-turning manner is thought to be the influence of different spatial characteristics. When pedestrians move from a wide-open space to a narrow sidewalk, they exhibit less deceleration, indicating an early reduction in speed to minimize subsequent slowdowns. Conversely, pedestrians transitioning from mixed traffic spaces to

open spaces typically accelerate upon entering the open area, despite a prior deceleration before reaching the edge line between the sidewalk and open space.

Also, we find that turning slows down pedestrians, who tend to accelerate after turning onto a sidewalk. However, the increase in speed may not be as pronounced when turning into open space, possibly because pedestrians prefer to move at a more comfortable speed in open space. In general, pedestrians on sidewalks tend to be slightly faster than pedestrians in open spaces. We acknowledge that our method of detecting turning states may be relaxed for individuals who change direction significantly, which may affect the accuracy of the turning data and our analysis of turning speeds.

4.2 DUT Dataset

DUT Dataset Results and Discussion. We continue to use the same way to plot the relationship between speed and space in DUT dataset, as shown in Fig. 2(c). Then, we summarize the SVAs in Fig. 2(c):

- SVA1: From open space to crossing, approximate $-5 <$ position(x) < 4;
- SVA2: From crossing to sidewalk, approximate $13 <$ position(x) < 16

The statistical analysis results show the average p-values are 0.001 for SVA1 and 0.022 for SVA2, indicating the speed in both SVA1 and SVA2 is statistically different from the speed in the previous areas.

The potential reasons of the findings are due to the safety properties inherent in the crossing which is a mixed traffic space. That is: For safety reasons, pedestrians like to decrease their speed after they reach the next space (sidewalk) rather than on the crossing. It's important to recognize that the SVA's length can vary depending on the characteristics of the connected spaces, including their type, dimensions, and cultural factors. The SVA's length may change as pedestrians transition between different space types and can differ among individuals. This range is a general average, and individual variations are common.

4.3 Similarities and Differences in BIWI Dataset and DUT Dataset

Our findings highlight areas where pedestrians significantly adjust their speed during transitions between entirely pedestrian spaces (BIWI) and mixed traffic spaces (DUT), often characterized by higher safety awareness. To understand how pedestrian safety awareness affects speed changes during space transitions and the influence of movement direction, we aim to address two key questions: (1) Do speed changes differ when pedestrians exit pedestrian spaces to enter mixed traffic areas compared to the reverse scenario? (2) Does movement direction during transitions in pedestrian spaces impact the rate of speed change?

We use LOESS smoothing to model speed change rates within the SVAs and then create an interpolation function. To assess the quality of this interpolation, we calculate the Mean Absolute Error (MAE) between predicted and actual

values at the original data points. The low MAE values in Table 5 indicate that the interpolation function closely matches the original curve, demonstrating its quality as a representation.

Table 5. Quality of the LOESS curves (MAE) and the representative information of the rate of speed change curves in SVAs

Dataset	Moving Direction	SVA (m)	MAE (m/s^2)
BIWI dataset	(1) from sidewalk to open space (without turns)	$-5 <$ pos_x < -3	0.009
	(2) from open space to sidewalk (without turns)	$-2.3 <$ pos_x < 0	0.049
DUT dataset	(3) from open space to crossing	$-5 <$ pos_x < 4	0.001
	(4) from crossing to sidewalk	$13 <$ pos_x < 16	0.001

We describe the relationship between the curves by comparing the two interpolation functions and Pearson's Correlation Coefficient. Figure 3 shows the processed rate of speed change curves for two directions, which should be interpreted in a mirrored manner. In (1) and (2), pedestrians remain within pedestrian spaces, aware of their transition to or from mixed traffic areas. The Pearson value is 0.99, indicating a strong correlation between these two curves. In (3) and (4), we observe pedestrians in mixed traffic areas, with a Pearson value of -0.96, indicating a strong positive correlation between these curves.

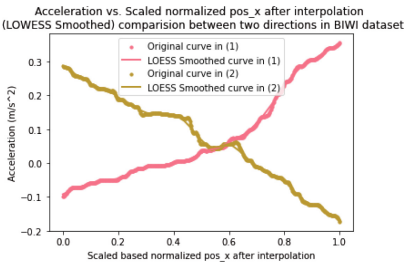


Fig. 3. Speed change rate vs. scaled-normalized pos_x after interpolation in SVAs.

The comparison between (1) and (2) shows that the rate of speed change remains consistent during transitions, regardless of the movement direction. Conversely, the comparison of (3) and (4) indicates that when transitioning from an open space to a crossing, pedestrians first slow down and then gradually accelerate and adapt to the mixed traffic. However, when transitioning from a crossing to a sidewalk, they initially maintain a higher speed and gradually slow down as their perceived threat decreases. Essentially, the trends in pedestrian speed change during different spatial transitions may be inversely related.

5 Conclusions

We carry out a data-driven study using two publicly pedestrian trajectory datasets collected in different countries, in spaces with different safety awareness, and at different times. We find that, pedestrian speeds significantly change near the edge line between mixed traffic and entirely pedestrian spaces, defining a SVA. The SVA's characteristics, including its location, magnitude of speed change, and acceleration/deceleration rates, depend on the space type, movement direction, and pedestrian behaviour, showcasing consistent patterns in non-turning pedestrians transitioning in opposite directions, and contrasting speed change rates in mixed traffic spaces. The potential reasons can be explained by the fact that pedestrians adjust their speed based on the perceived changes in space and their direction of movement, leading to distinct speed variations during spatial transitions. The identified speed characteristics around edge lines enhance our understanding and replication of real-world pedestrian behaviour across different spaces.

Furthermore, we believe that beyond speed, there are other motion characteristics worthy of consideration, and exploring how they can be applied to behaviour prediction is an intriguing research direction. Exploring pedestrian speed variations and rate of speed change in different spatial connections and factors influencing turning behaviour is also an intriguing research direction. Our findings on pedestrian speed changes during space transitions have practical applications in video games, pedestrian behaviour simulations, and data correction. They also shed light on how pedestrians perceive and adapt to various environments, opening avenues for further study.

References

1. Antonini, G., Bierlaire, M., Weber, M.: Discrete choice models of pedestrian walking behavior. Transp. Res. Part B: Methodol. **40**(8), 667–687 (2006)
2. Blomley, N.: Unsettling the City: Urban Land and the Politics of Property. Routledge (2004)
3. Franěk, M.: Environmental factors influencing pedestrian walking speed. Percept. Mot. Skills **116**(3), 992–1019 (2013)
4. Gössling, S., Schröder, M., Späth, P., Freytag, T.: Urban space distribution and sustainable transport. Transp. Rev. **36**(5), 659–679 (2016)
5. Granié, M.A., Brenac, T., Montel, M.C., Millot, M., Coquelet, C.: Influence of built environment on pedestrian's crossing decision. Accid. Anal. Prevent. **67**, 75–85 (2014)
6. Han, S., et al.: A systematic review of objective factors influencing behavior in public open spaces. Front. Public Health **10**(1), 1–20 (2022)
7. Handy, S.: Smart growth and the transportation-land use connection: what does the research tell us? Int. Reg. Sci. Rev. **28**(2), 146–167 (2005)
8. Hänseler, F.S., Lam, W.H., Bierlaire, M., Lederrey, G., Nikolić, M.: A dynamic network loading model for anisotropic and congested pedestrian flows. Transp. Res. Part B: Methodol. **95**, 149–168 (2017)

9. Luan, Q.Y., Liu, S.B., Fu, Z.J., Lyu, J.Y.: Experimental and modelling studies of collision avoidance strategy choices and behavioural characteristics in interweaving pedestrian flow. Roy. Soc. Open Sci. **9**(7), 1–25 (2022)
10. Nasr Esfahani, H., Song, Z., Christensen, K.: A deep neural network approach for pedestrian trajectory prediction considering flow heterogeneity. Transportmet. A: Transp. Sci. 1–24 (2022)
11. Orendurff, M.S., Segal, A.D., Berge, J.S., Flick, K.C., Spanier, D., Klute, G.K.: The kinematics and kinetics of turning: limb asymmetries associated with walking a circular path. Gait Posture **23**(1), 106–111 (2006)
12. Papadimitriou, E., Yannis, G., Golias, J.: A critical assessment of pedestrian behaviour models. Transport. Res. F: Traffic Psychol. Behav. **12**(3), 242–255 (2009)
13. Pellegrini, S., Ess, A., Schindler, K., van Gool, L.: You'll never walk alone: modeling social behavior for multi-target tracking. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 261–268 (2009)
14. Schönauer, R., Stubenschrott, M., Huang, W., Rudloff, C., Fellendorf, M.: Modeling concepts for mixed traffic: steps toward a microscopic simulation tool for shared space zones. Transp. Res. Rec. **2316**(1), 114–121 (2012)
15. Silvennoinen, H., Kuliga, S., Herthogs, P., Recchia, D.R., Tunçer, B.: Effects of Gehl's urban design guidelines on walkability: a virtual reality experiment in Singaporean public housing estates. Environ. Plann. B: Urban Anal. City Sci. **49**(9), 2409–2428 (2022)
16. Singh, P., Esposito, M., Barrons, Z., Clermont, C.A., Wannop, J., Stefanyshyn, D.: Measuring gait velocity and stride length with an ultrawide bandwidth local positioning system and an inertial measurement unit. Sensors **21**(9), 1–11 (2021)
17. Szymanezyk, O., Dickinson, P., Duckett, T., et al.: From individual characters to large crowds: augmenting the believability of open-world games through exploring social emotion in pedestrian groups. In: Proceedings of the 2011 DiGRA International Conference: Think Design Play, pp. 1–15 (2011)
18. Vanumu, L.D., Ramachandra Rao, K., Tiwari, G.: Fundamental diagrams of pedestrian flow characteristics: a review. Eur. Transp. Res. Rev. **9**, 1–13 (2017)
19. Willis, A., Gjersoe, N., Havard, C., Kerridge, J., Kukla, R.: Human movement behaviour in urban spaces: Implications for the design and modelling of effective pedestrian environments. Environ. Plann. B. Plann. Des. **31**(6), 805–828 (2004)
20. Winters, M., Buehler, R., Götschi, T.: Policies to promote active travel: evidence from reviews of the literature. Curr. Environ. Health Rep. **4**(3), 278–285 (2017)
21. Xin, X., Jia, N., Ling, S., He, Z.: Prediction of pedestrians' wait-or-go decision using trajectory data based on gradient boosting decision tree. Transportmet. B: Transp. Dyn. **10**(1), 693–717 (2022)
22. Yang, D., Li, L., Redmill, K., Özgüner, Ü.: Top-view trajectories: a pedestrian dataset of vehicle-crowd interaction from controlled experiments and crowded campus. In: 2019 IEEE Intelligent Vehicles Symposium (IV), pp. 899–904. IEEE (2019)
23. Zamboni, S., Kefato, Z.T., Girdzijauskas, S., Norén, C., Dal Col, L.: Pedestrian trajectory prediction with convolutional neural networks. Pattern Recogn. **121**, 1–12 (2022)

Standardizing Your Training Process for Human Activity Recognition Models – A Comprehensive Review in the Tunable Factors

Yiran Huang(✉), Haibin Zhao, Yexu Zhou, Till Riedel, and Michael Beigl

Telecooperation Office, Karlsruhe Institute of Technology, Karlsruhe, Germany
{yhuang,hzhao,zhou,riedel,beigl}@teco.edu

Abstract. In recent years, deep learning has emerged as a potent tool across a multitude of domains, leading to a surge in research pertaining to its application in the Wearable Human Activity Recognition (WHAR) domain. Despite the rapid development, concerns have been raised about the lack of standardization and consistency in the procedures used for experimental model training, which may affect the reproducibility and reliability of research results. In this paper, we provide an exhaustive review of contemporary deep learning research in the field of WHAR and collate information pertaining to the training procedure employed in various studies. Our findings suggest that a major trend is the lack of detail provided by model training protocols. Besides, to gain a clearer understanding of the impact of missing descriptions, we utilize a control variables approach to assess the impact of key tunable components (e.g., optimization techniques and early stopping criteria) on the inter-subject generalization capabilities of HAR models. With insights from the analyses, we define a novel integrated training procedure tailored to the WHAR model. Empirical results derived using five well-known WHAR benchmark datasets and three classical HAR model architectures demonstrate the effectiveness of our proposed methodology: in particular, there is a significant improvement in macro F1 Leave-One-Subject-Out (LOSO) Cross-Validation (CV) performance.

Keywords: human activity recognition · deep learning model

1 Introduction

Wearable Human Activity Recognition (WHAR) is a process that aims to classify human actions within a specific temporal boundary based on measurements,

Supported by the Ministry of Science, Research and the Arts Baden-Wuerttemberg as part of the SDSC-BW and by the Carl-Zeiss-Foundation as part of "stay young with robots" (JuBot) project.

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 15–27, 2024.
https://doi.org/10.1007/978-3-031-63992-0_2

e.g., acceleration, rotation speed, and geographical coordinates, gathered from personal digital devices [23]. It is central to numerous applications and services. For instance, in the smart homes [3], accurately detecting a user's activity, such as watching television, can enable the automation of derivative tasks like adjusting room light levels. Similarly, in industrial manufacturing [5], recognizing a worker's current task could aid in suggesting sequential actions for workflow optimization. In the context of road safety [29], identifying a driver's present driving behavior or mental states can catalyze measures to improve road safety. These applications place high-performance demands on a WHAR system, necessitating the guarantees of sufficient data and high-performance models.

Wearable technology has experienced significant growth in recent years, with nearly 500 million units shipped globally by the end of 2022, a 500% increase from the 82 million units shipped in 2016. Smartwatches are projected to be the most popular, accounting for 216 million units in 2022. Smart trackers and glasses follow, with 105 million and 32 million units, respectively. Equipped with various sensors, these devices offer advantages such as unobtrusive continuous data collection, direct user contact, and long battery life. Consequently, they provide detailed insights into human behavior, making wearables a key platform for developing applications that automatically recognize human activity.

According to Google Scholar[1], from 2021 to the present, a total of 1,660 academic articles have contained the keywords "human activity recognition" and "neural networks" in their titles.

Although these factors have contributed to the rapid growth of WHAR, they also pose challenges in terms of reproducing recent work and benchmarking research. There is a noticeable lack of attention given to explicating the process by which experimental models are trained. The training process of a model, encompassing the trajectory of model development from initialization to gaining expertise in a specific task, plays a pivotal role in determining the model's final performance.

A comprehensive description of the training process should include the following components: Dataset Description, Model Parameters Setting if present, Data Preprocessing, Batch Size and Maximum Epoch Setting, Optimizer Setting, Learning rate strategy, Early stopping strategy, Final model selection strategy, Validation&Test setting, Additional details.

By comprehensively articulating each stage of the training process, the intention of the training process description is to provide the reader with the ability to accurately replicate the experimental result described in the publication. Such documentation reinforces the transparency and integrity of the research, elements that are paramount in the field of deep learning science.

Nonetheless, our analysis points out that while nearly all publications deliver comprehensive details about dataset descriptions, model parameters setting, data preprocessing, and validation and test settings, there is a notable lack of exhaustive detail in other aspects.

[1] https://scholar.google.com.

Until now, the majority of review articles in the realm of WHAR have mainly concentrated on outlining various machine learning and deep learning architectures, with an emphasis on model design. Contrary to this trend, our research pivots its focus towards a comprehensive analysis of the model training process in the context of WHAR. To the best of our knowledge, this survey stands as a pioneering effort specifically targeting descriptions of the model training process. Through this exploration, we endeavor to shed light on the current state of model training process narratives in the relevant literature and proffer recommendations to guide future scholarly pursuits in this domain.

The key contributions of our work can be summarized below:

- To the best of our knowledge, we are the first in the WHAR field to write a review that focuses on the model training procedure in various research papers.
- This study delivers insights into how models have been trained in recent years. It identifies prevailing issues about model training in the field and highlights the potential consequences of these problems using control variables.
- In response to these identified issues, we propose a countermeasure and provide experimental evidence of its efficacy.
- We are releasing the proposed training procedure as an open-source package to aid in further research in this field.

Table 1. Summarize of the hyperparameters and the corresponding range in model training procedure.

Parameter	Range
Optimizer	{sGD [20], ADAM [13], Adadelta [27], Rmsprop}
Learning Rate	$[1e-1, 1e-5]$
Weight Decay	$[1e-1, 1e-8]$
Scheduler	{Step [25], LRPlateau [22], Cos [16], Cos+restart [16]}
Scheduler Base	{Train Loss, Validation Loss, Train Metric, Validation Metric}
Scheduler Patience (epoch)	[1, 100]
Batch Size	[16, 512]
Maximum Epoch	[10, 1500]
Early Stopping	{Yes, No}
Early Stopping Base	{Train Loss, Validation Loss, Train Metric, Validation Metric}
Early Stopping Patience (epoch)	[1, 100]
Model Selection Base	{Last, Train Loss, Validation Loss, Train Metric, Validation Metric}

2 Search Strategy

We followed Preferred Reporting Items for Systematic review and Meta-Analysis Protocols (PRISMA-P) to select relevant and important articles for our research target. We completed this search using two protocols: the search protocol and the three-level exclusion protocol.

Search Protocol: We utilized Google Scholar, a free academic search engine provided by Google, as our search platform. It provides an extensive coverage of academic literature across numerous disciplines worldwide, such as journal articles, theses, conference papers, and more. Its database is regularly updated, ensuring access to the most recent academic literature. Google Scholar supports keyword-based searches to retrieve related articles. In alignment with our research goals, we initially used the following three keyword sets for our exploration: ("Human Activity Recognition" OR "HAR"), ("Machine learning" OR "Deep learning" OR "Neural network"), and ("Wearable" OR "Smartphone" OR "Smartwatch" OR "Smartglasses" OR "Smartband"), combined with the platform's limited search statements. In light of our objective to analyze the latest developments, we limited the search time frame to 2021 through June 2023. Through this procedure, we sourced a total of 4,960 articles.

Three-Level Exclusion Protocol: We prioritized relevance and ranked papers by their impact and subsequently excluded duplicates, non-English articles, non-conference/journal papers, and those lacking references. We filtered out articles whose titles did not align with our research purpose. This initial inspection helped us narrow down to 274 articles. By reading the abstracts of these 274 articles, we removed survey articles and those that focused on other types of Human Activity Recognition (HAR) tasks. The second filtering phase yielded 208 papers. On downloading these papers and swiftly reviewing their contents, we discarded those primarily focusing on machine learning models, data collection, and data processing. The third and final filtering round culminated in a total of 102 papers.

3 Observation

In deep learning, beyond the inherent model parameters optimized using gradient descent, several hyperparameters critically influence a model's performance. These include the number of layers, kernel sizes, and training aspects like optimizer selection and batch size. These hyperparameters are not directly learned through training but profoundly affect the model's structural complexity, training speed, and overall efficacy.

The description of the training process in a research paper is essentially a record of the values taken for these parameters. It ensures that the entire training process of the model is unambiguous. Table 1 summarizes the hyperparameters that describe the training process, along with their common ranges within the WHAR domain as derived from 102 studied papers.

For a specific task, hyperparameter tuning aims to identify the optimal configuration of hyperparameters leading to the best model performance e.g. through Bayesian Optimization [26], reinforcement learning [9] or Monte Carlo [7,8]. However, as demonstrated in Table 1, the search space for training model hyperparameters is vast. It leads to substantial computational costs associated with hyperparameter tuning, especially for tasks like WHAR that involve large datasets and complex feature extraction using complex models. Individual model training on the classic oppo [2] dataset can take approximately one hour on Tesla P100[2]. Performing systematic hyperparameter searches incurs significant time and computational resource expenses. Furthermore, considering typical limitations in paper length and the focus of WHAR-related papers on introducing new model structures, algorithms, or theories, hyperparameter tuning is generally not the primary focus. Among the 102 papers we analyzed, only 16 of them (15.7%) involved hyperparameter tuning.

We observe that all these works only explored a subset of the hyperparameter space. Most concentrated on seeking for the learning rate and batch size, followed by the number of epochs and the choice of optimizer. There is a lack of exploration in other hyperparameter dimensions. Additionally, we found that the results of the search efforts were relatively consistent regarding optimizer and learning rate. All searches pertaining to the optimizer ultimately converged on selecting the Adam optimizer [13]. The majority of chosen learning rates were around $1e-3$, accounting for 64.3% of similar searches. Epoch values mostly centered around 60. Batch size results were polarized, with half of the studies selecting a size of 64, while the other half opted for batch sizes above 400. This apparent inconsistency in the search results further emphasizes the significance of hyperparameters. It underscores that the required hyperparameters can vary significantly based on differences in models and data.

Apart from the 16 articles that engaged in hyperparameter tuning, the remaining 86 articles addressed the potential adverse effects of neglecting hyperparameter tuning through various strategies. Some adopted parameter settings from similar works, while the majority rely on empirical choices [10,30]. Figure 1 illustrates the hyperparameter setting made by these articles in terms of the four mentioned parameters. Similar to the tuning results, these studies also predominantly favored the Adam optimizer [13]. Concerning learning rates, in addition to $1e-3$, $1e-4$ was also a popular choice. The setting of the epoch number deviated from the tuning results. Most of the works leaned towards larger epoch values. This trend could be attributed to the application of techniques like early stopping. In early stopping scenarios, even if a large number of epochs is set, training can be terminated ahead of time if specific conditions are met. For instance, if the validation loss ceases to decrease, training can halt prematurely. These strategies aim to mitigate the negative impact of neglecting exhaustive hyperparameter optimization. While they might not yield the optimal performance, they offer a practical approach given the resource constraints, limited paper space, and the specific focuses of the papers in the WHAR domain.

[2] The performance is tested with https://colab.research.google.com/.

Additionally, we observed in Fig. 1 that the "none" category has a high occurrence rate across all plots. This signifies that, using batch size as an example, in these papers, 45% of them did not provide a description of the batch size. This indicates an incompleteness in the descriptions of their model training process. This scenario is likewise seen in other hyperparameter plots.

Figure 2 presents a summary of the descriptions related to parameter schedulers across all reviewed papers. It was found that 81% of the articles lacked descriptions about schedulers. Moreover, only 6% of the articles described early stopping procedures, and only 9% outlined the final model selection process.

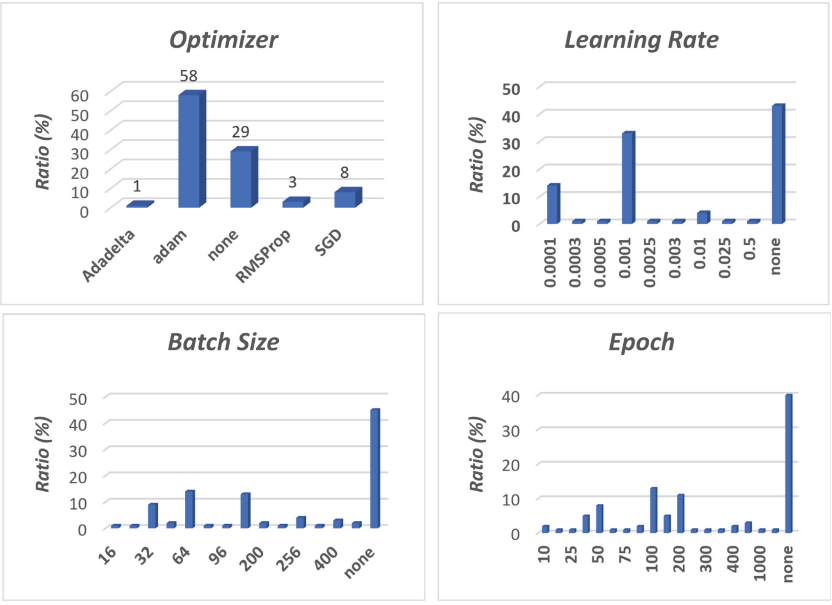


Fig. 1. Distribution on the four hyperparameters Optimizer, Learning Rate, Batch Size, and Epoch across all reviewed papers. 'none' indicates that there is no corresponding description in the paper.

Intuitively, it is clear that challenges due to descriptive deficiencies can significantly affect the reliability and reproducibility of the corresponding articles, underscoring the need for a standardized and comprehensive description of model training. However, the actual strength of the impact of the parameters describing the model training process, for instance, optimizer settings (type, learning rate, weight decay), early stopping strategies and model selection strategies, on the WHAR model end performance remains an open question, which is discussed in the next section.

4 Control-Variates Analysis

This section aims to investigate the independent impact of the aforementioned factors on the final model performance in WHAR context. To achieve this, we employ the control-variates method and run experiments with CNNLSTM [17] model on the HAPT [19] dataset. The baseline training procedure is derived from the collected papers by identifying the most frequently utilized hyperparameters (as described in Sect. 3) in the training procedure. Specifically, an Adam optimizer [13] is employed with an initial learning rate set at $1e-3$ and no weight decay. Cross-entropy is selected as the loss function, along with batch training set to a batch size of 64. The maximum number of epochs is capped at 60. A step scheduler is also utilized to reduce the learning rate by a factor of 0.1 every 10 epochs. For early stopping, we use a 10-epoch patience also based on validation loss. The test model selected runs on the least validation loss post-training. Figure 3 visualizes the validation loss during the training in control-variates experiment for hyperparameter: optimizer, learning rate, weight decay and batch size.

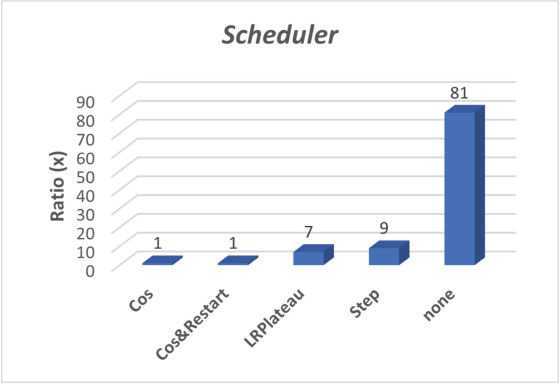


Fig. 2. Distribution of choices for scheduler hyperparameters across all reviewed papers.

Figure 3(a) depicts the impact of diverse optimizers (i.e., sGD [20], Adagrad [4], and Adam [13]) on model training. Our analysis finds notable variance in results across these optimizers. Of the three, Adam [13] yields the best performance. Conversely, sGD [20] does not converge due to its hyperparameter sensitivity and inappropriate initialization.

Figure 3(b) illustrates the effect of different initial training rates. Notably, although the baseline Adam optimizer [13] can adaptively adjust the step size, which is almost equivalent to adjusting the learning rate, the initial learning rate still plays a crucial role in the model training. In particular, setting an improper initial learning rate may hinder the model from converging (e.g., the

black curve), leading to suboptimal performance. In addition, the best model performance is achieved by setting the initial learning rate $1e-3$.

Despite the strong capability of weight decay for improving model generalization in other domains [6,15], Fig. 3(c) draws a different conclusion in WHAR domain. We can observe that the utilization of different weight decays does not significantly influence the validation loss. An excessive weight decay may even increase the validation loss, conversely, setting the weight to $1e-4$ seems to facilitate a more stable training process.

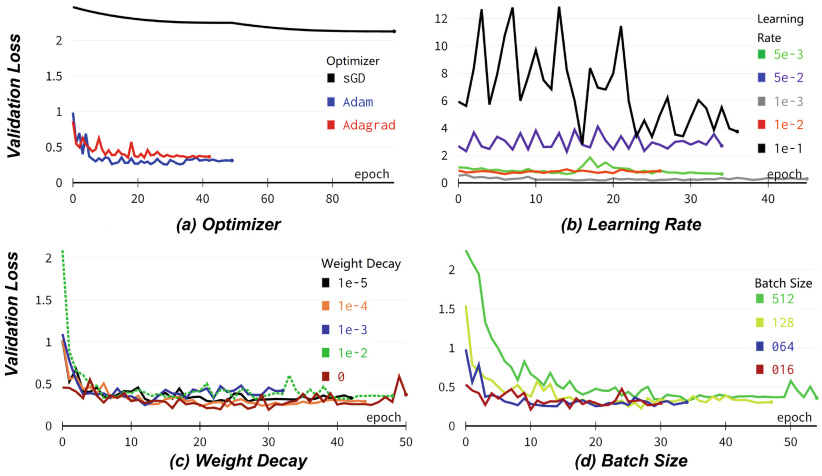


Fig. 3. Validation loss during model training in control-variates experiment.

Figure 3(d) summarizes the validation loss with different batch sizes. Consistent with the conclusion from another independent work [12], a smaller batch size leads to the better model generalization. However, as reported in Fig. 1, it is not a common option in recent WHAR articles.

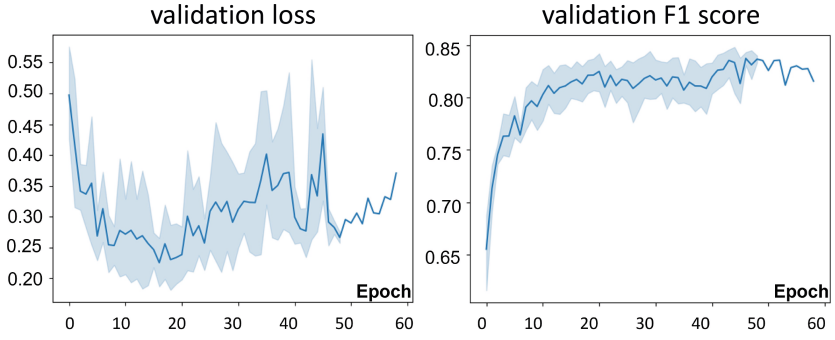


Fig. 4. Mean and standard deviation of the validation loss (left) and macro F1 score (right) of the models trained on HAPT dataset.

Figure 4 depicts the validation loss and macro F1 score over the epoch during the training on HAPT [19] dataset. The solid line represents the mean F1 score w.r.t. subjects, while the shading indicates the corresponding variance. Our analysis reveals that the validation loss decreases gradually and reaches its minimum at approximately the 15th epoch. However, the loss tends to increase progressively thereafter. In contrast, the F1 score continues to improve gradually even after the 15th epoch. This implies that employing validation loss for early-stopping may hinder the model from reaching a higher F1 score, which is the actual metric for evaluating the model performance. Additionally, the model with early-stopping does not perform as well as the model obtained by terminating the training at the maximal training epochs.

Based on the analysis of the control-variates approach, we can conclude that all the aforementioned factors have a discernible impact on the final performance of WHAR models. Therefore, their inclusion in academic papers is crucial to ensuring the reproducibility of experiments.

Table 2. Summary of the datasets used in the experiment. The abbreviations acc, gyro, mag denote 3d accelerometers, gyroscopes and magnetometers, respectively.

Name	#Subjects	#Channels	#Length	#Classes	Sensors type	Freq
DSADS	8	45	126	19	acc, gyro, mag	25 Hz
HAPT	30	6	128	12	acc, gyro	50 Hz
OPPO	4	77	30	18	acc, gyro, mag	30 Hz
PAMAP2	9	18	168	12	acc, gyro	100 Hz
RWHAR	15	21	128	8	acc	50 Hz

5 Experiment

It has been observed that frequently used training hyperparameters derived from reviewed articles do not yield an optimal WHAR model. Accordingly, without using hyperparameter tuning, how should we set the parameters of the training process to get relatively close to optimal results? To address this, we propose a hypothesis: by employing the optimal setting for each factor (determined during the control-variates analysis) for model training (denoted as *new*), the performance of the trained model could be enhanced, surpassing the effectiveness of the baseline used in Sect. 4 (denoted as *comm*). Specifically, our approach involves using an Adam optimizer [13] with an initial learning rate of $1e-3$ and a weight decay of $1e-4$. In addition, a Cosine Scheduler [16] is employed, along with the application of the cross-entropy loss function [21]. We utilize batch training with a batch size of 16 and apply the F1 score on the validation set for early stopping, with a patience of 30 epochs. After training, the optimal model is chosen based on the best F1 score on the validation set. In this section, we conduct an experiment to test this hypothesis.

Benchmark WHAR Datasets. To ensure experimental fairness, we conducted experiments on five most widely used benchmark datasets, i.e., DSADS [1], HAPT [19], OPPO [2], PAMAP2 [18], and RW [24]. As summarized in Table 2, these datasets differ significantly in terms of sensor types, sampling rates, number of subjects, number of channels, sequence length and application scenarios. As data preprocessing, we applied sliding windows to split data in segments, specifically, we employed a 50% overlap between adjacent windows for the training and validation sets. While for the test set, we used a 90% overlap between adjacent windows [11].

Model. To ensure the generalizability of our new training process, three models with different structures are utilized as our target models, namely, Convolution Neural Network (CNN)-based model (MCNN) [14], hybrid CNN-LSTM model (CNNLSTM) [17], and transformer-based model (Transformer) [28].

Evaluation and Metric. To evaluate the performance of the final obtained models and demonstrate their generalizability across subjects, we employ the LOSO-CV method on all datasets. In response to the category imbalance present within the WHAR data, we apply the F1 score as the evaluation metric. Moreover, to ensure the robustness of our results, we repeat the experiment five times using different random seeds and report the average results for all experiments.

Baseline. We identified the most frequently utilized training procedure in the collected academic papers and use them as our baseline (comm). Concretely, we use an Adam optimizer [13] with an initial learning rate of $1e-3$ without weight decay. We use the cross-entropy loss to guide the batch training with a batch size of 256. We employ an LRscheduler that reduces the learning rate by a factor of 0.1 based on the validation loss with patience of 10 epochs. We apply early-stopping with the patience of 30 epochs also on validation loss. Finally, we consider the model with the lowest validation loss as the optimal model.

5.1 Result

Table 3 presents the results of our experiments. We can observe that, the new training procedure achieved significantly better results compared to the baseline procedure on all datasets and across all models. This enhancement confirms the effectiveness of the proposed training procedure. In addition, it is noteworthy to mention that, this enhancement can be gained without any supplementary demands on the training resource and any increasing model inference time.

Table 3. LOSO-CV performance (mean F1 score) comparison.

Model	Method	DSADS	HAPT	OPPO	PAMAP2	RW
MCNN	comm	0.801	0.779	0.405	0.708	0.691
	new	**0.865**	**0.802**	**0.432**	**0.734**	**0.747**
CNNLSTM	comm	0.854	0.806	0.392	0.735	0.706
	new	**0.900**	**0.820**	**0.395**	**0.764**	**0.768**
Transformer	comm	0.812	0.767	0.402	0.654	0.598
	new	**0.872**	**0.805**	**0.440**	**0.758**	**0.731**

6 Conclusion and Future Work

In this study, we highlight the prevalent trend that, the majority of academic papers are deficient in providing a comprehensive description of the model training process. Employing the control-variables analysis, we affirm that, these overlooked factors exert a significant influence on the model performance. Therefore, the lack of these descriptions can markedly affect the reproducibility of the achievements reported in those academic papers. In addition, based on the control-variates analysis, we propose a new training procedure and test its effectiveness by conducting extensive experiments on five widely used benchmark datasets using three models with different architectures. The experiments indicate that, the new training procedure can achieve significantly better results compared to the most common training procedure on all datasets and all models.

References

1. Barshan, B., Altun, K.: Daily and sports activities. UCI Machine Learning Repository (2013). https://doi.org/10.24432/C5C59F
2. Chavarriaga, R., et al.: The opportunity challenge: a benchmark database for on-body sensor-based activity recognition. Pattern Recogn. Lett. **34**(15), 2033–2042 (2013)
3. Du, Y., Lim, Y., Tan, Y.: A novel human activity recognition and prediction in smart home based on interaction. Sensors **19**(20), 4474 (2019)
4. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. J. Mach. Learn. Res. **12**(7) (2011)
5. Günther, L.C., Kärcher, S., Bauernhansl, T.: Activity recognition in manual manufacturing: detecting screwing processes from sensor data. Procedia CIRP **81**, 1177–1182 (2019)
6. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science **313**(5786), 504–507 (2006)
7. Huang, Y., Schaal, N., Hefenbrock, M., Zhou, Y., Riedel, T., Beigl, M.: McXai: local model-agnostic explanation as two games. In: 2023 International Joint Conference on Neural Networks (IJCNN), pp. 01–08. IEEE (2023)

8. Huang, Y., Zhou, Y., Hefenbrock, M., Riedel, T., Fang, L., Beigl, M.: Automatic feature engineering through Monte Carlo tree search. In: Amini, M.R., Canu, S., Fischer, A., Guns, T., Kralj Novak, P., Tsoumakas, G. (eds.) Machine Learning and Knowledge Discovery in Databases. Lecture Notes in Computer Science, vol. 13715, pp. 581–598. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-26409-2_35
9. Huang, Y., Zhou, Y., Hefenbrock, M., Riedel, T., Fang, L., Beigl, M.: Universal distributional decision-based black-box adversarial attack with reinforcement learning. In: Tanveer, M., Agarwal, S., Ozawa, S., Ekbal, A., Jatowt, A. (eds.) ICONIP 2022. LNCS, vol. 13625, pp. 206–215. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-30111-7_18
10. Huang, Y., Zhou, Y., Riedel, T., Fang, L., Beigl, M.: randomHAR: improving ensemble deep learners for human activity recognition with sensor selection and reinforcement learning. arXiv preprint arXiv:2307.07770 (2023)
11. Jordao, A., Nazare Jr., A.C., Sena, J., Schwartz, W.R.: Human activity recognition based on wearable sensor data: a standardization of the state-of-the-art. arXiv preprint arXiv:1806.05226 (2018)
12. Keskar, N.S., Mudigere, D., Nocedal, J., Smelyanskiy, M., Tang, P.T.P.: On large-batch training for deep learning: generalization gap and sharp minima. arXiv preprint arXiv:1609.04836 (2016)
13. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
14. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. Commun. ACM **60**(6), 84–90 (2017)
15. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
16. Loshchilov, I., Hutter, F.: SGDR: stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983 (2016)
17. Mutegeki, R., Han, D.S.: A CNN-LSTM approach to human activity recognition. In: 2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), pp. 362–366. IEEE (2020)
18. Reiss, A., Stricker, D.: Introducing a new benchmarked dataset for activity monitoring. In: 2012 16th International Symposium on Wearable Computers, pp. 108–109. IEEE (2012)
19. Reyes-Ortiz, J.L., Oneto, L., Samà, A., Parra, X., Anguita, D.: Transition-aware human activity recognition using smartphones. Neurocomputing **171**, 754–767 (2016)
20. Robbins, H., Monro, S.: A stochastic approximation method. Ann. Math. Stat. 400–407 (1951)
21. Shannon, C.E.: A mathematical theory of communication. ACM SIGMOBILE Mob. Comput. Commun. Rev. **5**(1), 3–55 (2001)
22. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
23. Straczkiewicz, M., James, P., Onnela, J.P.: A systematic review of smartphone-based human activity recognition methods for health research. NPJ Digit. Med. **4**(1), 148 (2021)
24. Sztyler, T., Stuckenschmidt, H.: On-body localization of wearable devices: an investigation of position-aware activity recognition. In: 2016 IEEE International Conference on Pervasive Computing and Communications (PerCom), pp. 1–9. IEEE (2016)

25. Tang, Y., Zhang, L., Min, F., He, J.: Multiscale deep feature learning for human activity recognition using wearable sensors. IEEE Trans. Industr. Electron. **70**(2), 2106–2116 (2022)
26. Wu, J., Chen, X.Y., Zhang, H., Xiong, L.D., Lei, H., Deng, S.H.: Hyperparameter optimization for machine learning models based on Bayesian optimization. J. Electron. Sci. Technol. **17**(1), 26–40 (2019)
27. Zeiler, M.D.: ADADELTA: an adaptive learning rate method. arXiv preprint arXiv:1212.5701 (2012)
28. Zhang, Z., Wang, W., An, A., Qin, Y., Yang, F.: A human activity recognition method using wearable sensors based on convtransformer model. Evol. Syst. 1–17 (2023)
29. Zhao, H., Scholz, A., Beigl, M., Ni, S., Singaraju, S.A., Aghassi-Hagmann, J.: Printed electrodermal activity sensor with optimized filter for stress detection. In: Proceedings of the 2022 ACM International Symposium on Wearable Computers, pp. 112–114 (2022)
30. Zhao, H., Zhou, Y., Riedel, T., Hefenbrock, M., Beigl, M.: Improving human activity recognition models by learnable sparse wavelet layer. In: Proceedings of the 2022 ACM International Symposium on Wearable Computers, pp. 84–88 (2022)

MyPath: Accessible Route Generation Using Crowd-Sensed Surface Information

Thomas Nguyen[1], Md Fourkanul Islam[2], Rochishnu Banerjee[1(✉)], Hanna M. Noyce[3], Emily M. Olejniczak[3], Roger O. Smith[3], Md Osman Gani[2], and Vaskar Raychoudhury[1]

[1] Department of Computer Science and Software Engineering, Miami University, Oxford, OH, USA
{nguyenk7,banerjr2,raychov}@miamioh.edu
[2] Department of Information Systems, University of Maryland, Baltimore County, USA
{mislam7,mogani}@umbc.edu
[3] Department of Rehabilitation Sciences and Technology, University of Wisconsin, Milwaukee, USA
{hmnoyce,olejnic8,smithro}@uwm.edu

Abstract. Wheelchair users often find it difficult to navigate unfamiliar areas both built and natural. Although routing applications have been designed for cars and bikes, accessible routing for wheelchair users is a non-trivial problem that has been scarcely investigated. Wheelchair navigation is obstructed by several barriers such as broken or uneven path surfaces, steep inclines, absence of curb ramps and access slopes, etc. An accessible route enables wheelchair users to avoid such barriers while navigating familiar and unfamiliar environments. In this paper, we propose MyPath - an accessible routing system that leverages crowd-sourced surface vibration data captured through smartphones carried by wheelchair users. Data contributed by the users are then used to automatically recognize surface types and to populate the OpenStreetMap (OSM). Our efficient routing algorithms leverage rich OSM information to generate end-to-end accessible routes and present them through an interactive user interface. MyPath system has been extensively evaluated by a dedicated expert group of wheelchair users and other participants to ensure the seamless operation of the proposed system.

Keywords: Mobile app development · Tracking and localization · Accessible routing · Wheelchair users

1 Introduction

Traveling on sidewalks and paths, especially in unfamiliar areas, is challenging for wheelchair users because of unpredictable barriers, such as uneven surfaces, discontinuous sidewalks, insurmountable stairs, steep slopes, or missing curb ramps. These routes can become more challenging in varied weather conditions such as

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 28–39, 2024.
https://doi.org/10.1007/978-3-031-63992-0_3

heavy rains and snow. For someone using a wheelchair, an inaccessible route can quickly become an overwhelming impediment resulting in a failed outing, delayed arrival, and physical exhaustion. Additionally, attempts to analyze and identify suitable routes may become too difficult or time-consuming, which reduces a wheelchair user's desire to go on future excursions. Consequently, knowing the most optimal route to any destination is essential for a user's successful community participation. Due to the variable nature of the accessibility of paths, some regulations such as [1,2] have been introduced that lay down clear guidelines for the accessibility of surfaces. ADA mandates accessible surfaces to be stable (resist movement), firm (resist deformation by applied forces), slip-resistant, and should have enough integrity and resilience.

The research on accessible routing has developed in many different paths. The major challenge in the domain of accessible routing is tackling the problem of unpredictable barriers through mapping technologies. Crowd-sourcing is a popular method of easily identifying inaccessible barriers in public areas [3,4]. Satellite-based geographic information systems (GIS) [5,6] and GPS [7] are also utilized along with smartphones with embedded accelerometers and gyroscopes [8–11]. Path accessibility information databases such as OpenStreetMap (OSM) [12,13] have also leveraged crowd-sourcing to collate valuable information about accessibility barriers [14–19]. The actual routing process can be initiated only after these barriers have been categorically identified.

However, there are fewer attempts to develop an accessible routing algorithm that generates custom routes based on the accessibility of the path. In this paper, we introduce the MyPath accessible routing system for wheelchair users which works by crowd-sourced surface vibration data collected by wheelchair users in an unobtrusive manner through the MyPath app. Our MyPath app is an accessible app by itself and is thoroughly verified by a team of wheelchair users, blind and low vision (BLV) users, and rehabilitation and mobility experts. The app has both Android and iOS versions and is tested for many hours for correct functioning.

The MyPath system consists of three interconnected modules - sensor data collection, ML-based surface classification, and accessible route generation (see Fig. 1). We model the accessible routing problem in the built environment as a graph-theoretic problem on a weighted road network and propose a tree-based path routing algorithm for estimating the optimal path for wheelchair users. The path-finding algorithm uses the accessibility score calculated from the surface type and incline of a path to generate the optimal route. We have already validated the effectiveness of our ML models for surface classification through prior work [20–22] and in this paper, we only discuss the system design aspects.

In summary, this paper makes the following contributions.

- We designed and developed an accessible app for collecting surface vibration data from wheelchair users through smartphone sensors.
- We proposed two accessible routing algorithms. The first algorithm generates the optimal route for the highest accessibility. The second algorithm generates multiple accessible routes between a pair of source and destination

coordinates. We developed a web-based application to find accessible routes from point to point for wheelchair users.
– We conducted extensive testing involving stakeholders to make sure our MyPath app is accessible. The app and the routing modules are tested thoroughly for functional correctness using modular testing involving real users.

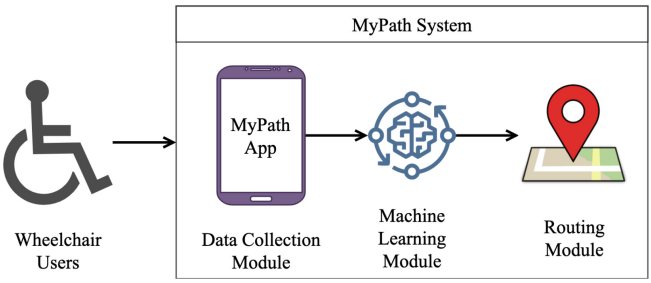


Fig. 1. Workflow of the MyPath system with three interconnected modules. In this paper we focus on the Data Collection and Routing Modules.

2 Related Works

In this section, we discuss the related research papers that deal with accessible routing and navigation techniques for wheelchair users. These include data recognition and analyzing methods for detecting real-time street obstacles and conditions, route generation, recommender systems, and routing algorithms.

2.1 Accessible Routing and Navigation Systems

Plenty of researchers have explored the socially and technically relevant problem of accessible routing and navigation in different contexts. We have identified three unique approaches that use mobility assistants, crowd-sourced knowledge, as well as well-defined accessibility criteria to determine accessible routes.

Mobility Assistants. Some researchers have focused on developing a map of the built environment using GPS and GIS technologies to properly identify various barriers and facilities in the built environment for wheeled mobility-aid users. Mobility assistants were software systems that created GIS models for the surrounding environment. Some well-known GIS-based mobility models are MAGUS [5,6] and U-Access [23]. MAGUS assigned different impedance scores to different routes based on the number of barriers, distance, slopes, surface type and the number of road crossings. U-Access [23], also considers different types of environmental objects as well as the physical abilities of the users while choosing the best routes.

Several systems [8–11] utilize embedded smartphone sensors such as gyroscope, accelerometer, and location information. Karimanzira et al. [24] applied machine learning algorithms to generate accessible routes for wheeled mobility.

Crowd-Sourced Recommendation Systems. Crowd-sourcing helps to populate information about the built environment on an accessibility map using the knowledge of many users. Extensive use of mobile phones fueled the use of crowd-sourcing and user-generated contents (UGC) (like geo-tagged pictures) for routing and navigation purposes [25]. The RouteCheckr system [26] also allows collaborative data annotation to generate personalized routes for its users. Mobile Pervasive Accessibility Social Sensing (mPASS) [27] also creates personalized routing paths but their crowd-sourced data also involves sensory data collected via geo-referencing. OpenStreetMap [12,13] has emerged as leading source of up-to-date mapping database, whose data was incorporated by intelligent navigation systems such as eNav [28]. A crowd-sourced mobility assistant for wheelchair users - WheelShare [29–31], was developed to generate end-to-end routes based on surface types identified by a robust machine learning system.

Defining Accessibility Criteria. The main challenge in modeling accessibility to generate optimal routes comes from the erratic and ever-changing features of routes due to man-made obstructions or inclement weather. Accessible routes can quickly become uncomfortable due to flooding, lack of maintenance, and not adhering to improving accessibility regulations such as having a slope beyond one-in-twenty or having a path narrower than 915 mm [1]. The authors of [32] proposed an algorithm for building a routing networks based on multiple accessible criteria collated from OpenStreetMap (OSM) data. Kawabata [33] proposed a context-adaptable pedestrian navigation system, which is capable of generating user-preferred routes. However, the author considered only blocking stairs and road surfaces but not existing slopes and ramps. Rahaman et al. [34] extended Kawabata's work by also considering the slope of the path, and implemented a multi-objective A* search algorithm (CAPRA) [35] that reports distance and elevation. Sahelgozin et al. [36] introduced a novel optimization method, Ubiquitous Pedestrian Way Finding Service (UPWFS) that defined length, safety, difficulty, and attraction as four criteria that must be met for a personalized path that suits users.

2.2 Routing Algorithms

There are many notable approaches taken to tackle the problem within some constraints. First of all, we need to note that all the shortest path finding algorithms cannot be utilized in this new problem because there's a lack of optimal substructure. Thus, many approaches go in either reducing the complexity of the problem with assumption or creating a heuristic for the estimated result. One approach by Quang Dung Pham and Yves Deville [37] tackles the problem

with a constraint-based technique. They re-define the problem within eight constraints about the graph's property and provide two approaches. One uses the idea of DFS to search for the exact solution while the another utilizes a local search path where the next optimal move is calculated given the previous one in a tabular manner. Also, Wong [38] mentioned the longest simple path on graphs within the domain of information retrieval on peer-to-peer networks where vertices are labeled with weights. A genetic algorithm is proposed in the paper to solve the problem. However, most papers here either proposed an approximate solution without proof of optimum or a restraint-ed exact solution applied on special graph. In this paper, we come up with a solution with high performance in time and correctness.

3 MyPath System

The MyPath project is currently divided into three modules: 1) MyPath Data Collection App, 2) MyPath Surface Classification, and 3) MyPath Accessible Routing. These three modules are connected to produce a crowd-sourced data collection and routing system to assist wheelchair users in navigating the built environment. MyPath system efficiently integrates both the data collection and routing algorithm to present a comprehensive and self-sufficient intelligent system capable of generating accurate accessible routing solutions. Figure 2 depicts a schematic diagram of the proposed system architecture with its internal components that are distributed between client and server-side operations.

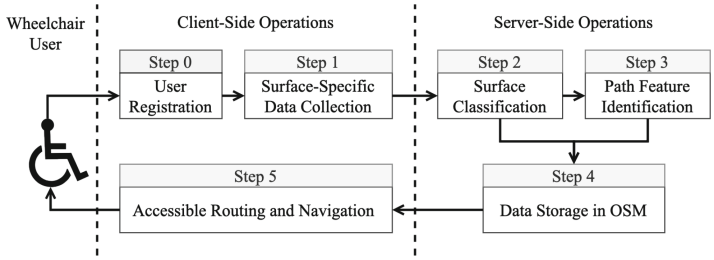


Fig. 2. Complete MyPath System Architecture. Client-side: User registration (step 0), Data collection (step 1), and Accessible route generation and navigation (step 5). Server-side: Surface classification (step 2), Path feature identification (step 3), and Data upload to OSM (step 4).

3.1 MyPath System Functionality

The MyPath system was built in two phases: an internal training phase, and an external live phase. In the training phase, path data from various indoor and outdoor surfaces were manually collected and used to train a robust machine-learning model for surface detection. The detected surfaces were verified and

further used as input data for the initial testing of the routing algorithms. The live phase consists of recruiting real wheelchair users who will collect vibrational data in their local community in an unobtrusive manner which will be stored and used to generate personalized end-to-end routes. Figure 2 illustrates the functional workflow of the MyPath system. Step 1: A wheelchair user registers an account with the system which is used to return personalized routes. Step 2: The user collects data using a smartphone which is automatically uploaded to our server. Step 3: The collected data is fed into the surface recognition model which predicts the surface type. Step 3–4: The surface type and inclination features of the paths are accumulated and uploaded to OSM. Step 5: The routing algorithm pulls the data from OSM to generate personalized routes which are displayed on the smartphone to assist the wheelchair user.

3.2 MyPath System Architecture

In this section, we describe the Mypath system architecture from an operational perspective by dividing the various functionalities discussed in Sect. 3 into *client-side* and *server-side* modules. The client-side operations of the MyPath system consist of two modules - a *data collection module* and a *routing and navigation module*. The Data Collection Module is developed as a cross-platform mobile (Android and iOS) application whereas the Routing and Navigation Module can be accessed through a web application. In the future, we aim to integrate all working modules into a seamless smartphone application with web functionalities. The MyPath server performs multiple tasks, 1) pre-processing and hosting the de-identified collected data, 2) classifying surfaces using deployed machine learning model, and 3) generating accessible routes using our algorithm. It provides several APIs to securely communicate with our smartphone application.

3.3 MyPath Smartphone Application

MyPath is a cross-platform Android and iOS mobile application that aims to improve accessibility and navigation for people with limited mobility. Its primary purpose is to collate data from diverse wheelchair users through crowd-sourcing which will facilitate accessible navigation for individuals with limited mobility. MyPath leverages the distributed data generated by a large number of wheelchair users from geographically diverse communities to create personalized routes for individual users' specific needs and requirements. The app collects accelerometer, gyroscope, and location data using device sensors, which are stored locally alongside session information to maintain data context. MyPath also provides navigation assistance and data visualization features that allow users to view and track their collected data. Figure 3 shows the welcome, data collection, and data visualization screen of the MyPath app.

3.4 MyPath Route Generation

We model the accessible route generation for wheelchair users as a graph problem where the graph G has a set of edges E connecting a set of vertices V. We formally define the accessible routing problem as follows. Given an undirected weighted graph $G(V, E, W)$, withf $E = \{\{x, y\}|x, y \in V\}$, and W is a set of edge weights in the form of $W = \{w(e)|e \in E\}$, we aim to find the simple path S_p such that the max edge weight in S_p is minimum. We developed two algorithms to generate end-to-end routes between a source and a destination point. Single-Route-Generation Algorithm generates a single optimal route from a weighted graph using a variation of Prim's Minimum Spanning Tree (PMST) algorithm. Multiple-Route-Generation Algorithm applies the Depth-First Search (DFS) approach to generate multiple routes between the source and the destination nodes. The first algorithm utilizes the accessibility score to generate an optimal path while the second one generates all possible paths within a boundary. Thus, the output of the second algorithm might or might not contain the output of the first algorithm.

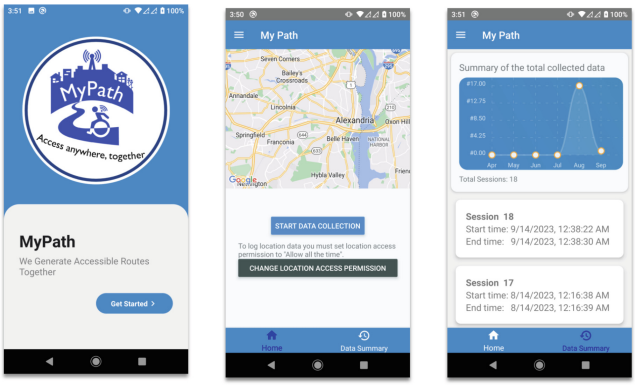


Fig. 3. MyPath App UI Flow

4 Performance Evaluation

Each module of the MyPath system has been rigorously evaluated to ensure seamless working of the MyPath data collection and routing modules. The testing has been performed in three different project sites across the United States (location hidden for anonymity) covering more than 150 paths and walkways. The evaluation of the systems was done as a two-part process.

4.1 Part 1: Evaluation of the MyPath Data Collection App

We have evaluated the MyPath smartphone app based on the accuracy and precision of the GPS localization and the accessibility of the App to wheelchair users and others with low vision, and mobility experts.

- The data collected by the GPS sensors was analyzed to ensure the correctness and high accuracy of real-time localization that is displayed in the app. The precision of the mapped coordinates was tested by collecting data from opposite sidewalks of the same road as shown in Fig. 4. The localization shows a high degree of accuracy in identifying the different sidewalks.
- To evaluate the accessibility of the MyPath smartphone app, a participatory action research (PAR) and advisory group was created consisting of wheelchair users, stakeholders, and technical experts. The progress of the research team was monitored and discussed every three months by the PAR group, and every 6 months by the Advisory group. These groups provided us with valuable insight into enhancing the usability, accessibility, and overall performance of the MyPath system. Figure 5 shows the changes made to the app display after consulting with the PAR and advisory groups.

Fig. 4. Accuracy of localization for identifying left and right sidewalks of the same road

4.2 Part 2: Evaluation of the MyPath Routing Module

To evaluate the overall routing module on a real map, we collected surface type and incline data covering an area of 0.3 km^2 and over 100 paths. To facilitate the testing of the routing application, inclination data was collected manually by measuring the incline level on all pathways for every 10–20 ft. of distance. These inclines and the surface types of each path were uploaded to Open Street Maps (OSM). The routing application pulls the data from OSM to generate the routes rendered on the map. Not all wheelchairs can access steep inclines and consequently, each wheelchair has a maximum threshold of inclination it can handle without slipping, tipping over, and the user losing control. This accessibility (or

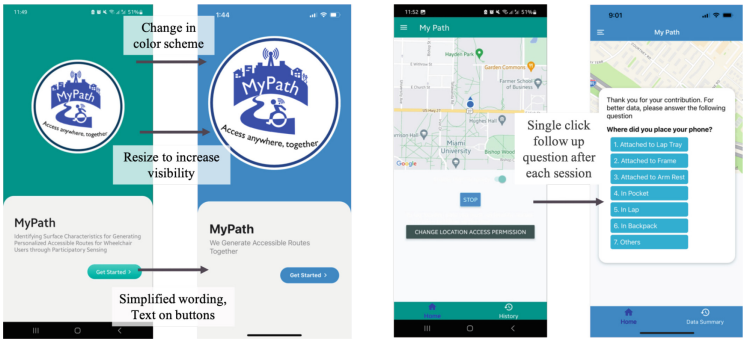


Fig. 5. UI changes in the MyPath App based on PAR recommendations

inaccessibility) of the paths based on the inclination of the path generates a heatmap which is shown in Fig. 6. Figure 7 shows two examples of the routing application.

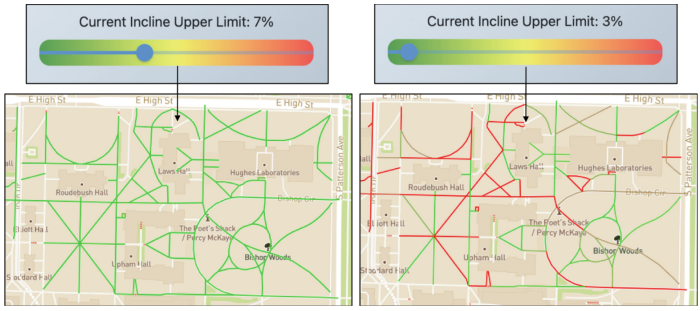


Fig. 6. Heatmap of incline accessibility

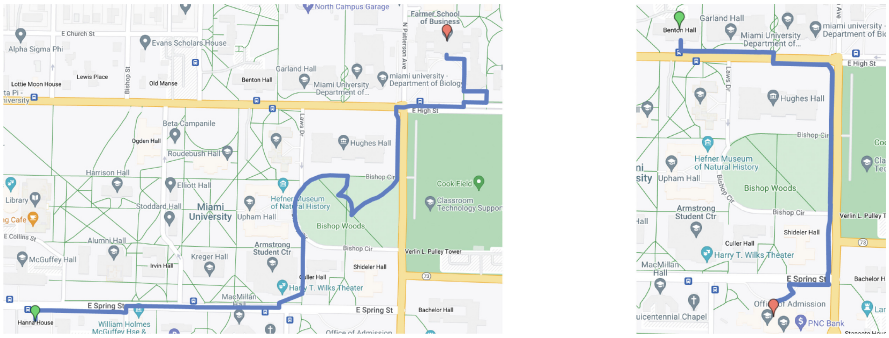


Fig. 7. Optimal route generation from surface type and incline data

5 Conclusion and Future Works

In this paper, we introduce MyPath which is an accessible route generation system for wheelchair users based on surface vibration information captured through their smartphones. MyPath system has a data collection app and a routing web service. The data collection app uses built-in sensors in smartphone to capture surface vibration and user location. This information is then used to automatically classify surfaces and to determine their accessibility through a scoring mechanism. Surface accessibility is then used to generate end-to-end accessible routes for wheelchair users through the built environment. Evaluation with real wheelchair users establishes the accessibility and effectiveness of the MyPath system. In the future, we plan to test the MyPath system with a wider audience and include step-by-step navigation information using textual and voice-based interaction.

Acknowledgement. This work was developed in part under a grant from the National Institute on Disability, Independent Living, and Rehabilitation Research (NIDILRR grant number 90IFDV0024). NIDILRR is a Center within the Administration for Community Living (ACL), Department of Health and Human Services (HHS). The content of this work does not necessarily represent the policy of NIDILRR, ACL, or HHS, and does not assume endorsement by the Federal Government.

References

1. ADA. Americans with Disabilities Act (ADA) Standards for Accessible Design. http://www.ada.gov/2010ADAstandards_index.htm
2. Barrierefreies bauen - teil 1: Straßen, plätze, wege, öffentliche verkehrs- und grünanlagen sowie spielplätze; planungsgrundlagen (1998)
3. Zambonelli, F.: Pervasive urban crowdsourcing: visions and challenges. In: 2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), pp. 578–583. IEEE (2011)
4. Bicocchi, N., Cecaj, A., Fontana, D., Mamei, M., Sassi, A., Zambonelli, F.: Collective awareness for human-ICT collaboration in smart cities. In: 2013 22nd IEEE International WETICE Conference (WETICE), pp. 3–8 (2013)
5. Beale, L., Field, K., Briggs, D., Picton, P., Matthews, H.: Mapping for wheelchair users: route navigation in urban spaces. Cartogr. J. **43**(1), 68–81 (2006)
6. Matthews, H., Beale, L., Picton, P., Briggs, D.: Modelling access with GIS in urban systems (MAGUS): capturing the experiences of wheelchair users. Area **35**(1), 34–45 (2003)
7. Kurihara, M., Nonaka, H., Yoshikawa, T.: Use of highly accurate GPS in network-based barrier-free street map creation system. In: 2004 IEEE International Conference on Systems, Man and Cybernetics, vol. 2, pp. 1169–1173. IEEE (2004)
8. Bujari, A., Licar, B., Palazzi, C.E.: Movement pattern recognition through smartphone's accelerometer. In: 2012 IEEE Consumer Communications and Networking Conference (CCNC), pp. 502–506. IEEE (2012)
9. Sinkonde, D., Mselle, L., Shidende, N., Comai, S., Matteucci, M.: Developing an intelligent PostGIS database to support accessibility tools for urban pedestrians. Urban Sci. **2**(3), 52 (2018)

10. Harle, R.: A survey of indoor inertial positioning systems for pedestrians. IEEE Commun. Surv. Tutor. **15**(3), 1281–1293 (2013)
11. Renaudin, V., Combettes, C.: Magnetic, acceleration fields and gyroscope quaternion (MAGYQ)-based attitude estimation with smartphone sensors for indoor pedestrian navigation. Sensors **14**(12), 22 864–22 890 (2014)
12. OpenStreetMap. Planet OSM Files. http://planet.openstreetmap.org
13. Ramm, F., Topf, J., Chilton, S.: OpenStreetMap: Using and Enhancing the Free Map of the World. UIT, Cambridge (2010)
14. Holone, H., Misund, G., Holmstedt, H.: Users are doing it for themselves: pedestrian navigation with user generated content. In: The 2007 International Conference on Next Generation Mobile Applications, Services and Technologies, NGMAST 2007, pp. 91–99. IEEE (2007)
15. Neis, P., Zipf, A.: OpenRouteService.org is three times "open": combining OpenSource, OpenLS and OpenStreetMaps. In: GIS Research UK (GISRUK 2008), Manchester (2008)
16. Neis, P., Goetz, M., Zipf, A.: Towards automatic vandalism detection in OpenStreetMap. ISPRS Int. J. Geo Inf. **1**(3), 315–332 (2012)
17. Neis, P., Zielstra, D., Zipf, A.: The street network evolution of crowdsourced maps: OpenStreetMap in Germany 2007–2011. Future Internet **4**(1), 1–21 (2011)
18. Zielstra, D., Hochmair, H.: Using free and proprietary data to compare shortest-path lengths for effective pedestrian routing in street networks. Transp. Res. Rec.: J. Transp. Res. Board **2299**, 41–47 (2012)
19. Neis, P., Zielstra, D.: Recent developments and future trends in volunteered geographic information research: the case of OpenStreetMap. Future Internet **6**(1), 76–106 (2014)
20. Mokrenko, V., Yu, H., Raychoudhury, V., Edinger, J., Smith, R.O., Gani, M.O.: A transfer learning approach to surface detection for accessible routing for wheelchair users. In: 2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC), pp. 794–803. IEEE (2021)
21. Yu, H., Raychoudhury, V., Saha, S., Edinger, J., Smith, R.O., Gani, M.O.: Automated surface classification system using vibration patterns-a case study with wheelchairs. IEEE Trans. Artif. Intell. (2022)
22. Banerjee, R., Islam, M.F., Saha, S., Raychoudhury, V., Gani, M.O.: Surface recognition from wheelchair-induced noisy vibration data: a tale of many cities. In: 2022 18th International Conference on Mobility, Sensing and Networking (MSN), pp. 619–626. IEEE Computer Society (2022)
23. Sobek, A.D., Miller, H.J.: U-access: a web-based system for routing pedestrians of differing abilities. J. Geograph. Syst. **8**(3), 269–287 (2006). https://doi.org/10.1007/s10109-006-0021-1
24. Karimanzira, D., Otto, P., Wernstedt, J.: Application of machine learning methods to route planning and navigation for disabled people. In: MIC 2006: Proceedings of the 25th IASTED international conference on Modeling, Indentification, and control, pp. 366–371 (2006)
25. Beeharee, A.K., Steed, A.: A natural wayfinding exploiting photos in pedestrian navigation systems. In: Proceedings of the 8th Conference on Human-Computer Interaction with Mobile Devices and Services, pp. 81–88. ACM (2006)
26. Völkel, T., Weber, G.: RouteCheckr: personalized multicriteria routing for mobility impaired pedestrians, pp. 185–192 (2008)
27. Prandi, C., Salomoni, P., Mirri, S.: mPASS: integrating people sensing and crowdsourcing to map urban accessibility. In: 2014 IEEE 11th Consumer Communications and Networking Conference (CCNC), pp. 591–595 (2014)

28. Džafić, D., Schoonbrood, P., Franke, D., Kowalewski, S.: eNav: a suitable navigation system for the disabled. In: Wichert, R., Mand, B. (eds.) Ambient Assisted Living. ATSC, pp. 133–150. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-52322-4_9
29. Gani, M.O., Raychoudhury, V., Edinger, J., Mokrenko, V., Cao, Z., Zhang, C.: Smart surface classification for accessible routing through built environment: a crowd-sourced approach. In: Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation, pp. 11–20 (2019)
30. Hata, J., Gani, O., Raychoudhury, V.: CARE: campus-wide accessible route estimation through surface analysis. In: Proceedings of the 20th International Workshop on Mobile Computing Systems and Applications, p. 173 (2019)
31. Edinger, J., Hofmann, A., Wachner, A., Becker, C., Raychoudhury, V., Krupitzer, C.: WheelShare: crowd-sensed surface classification for accessible routing. In: 2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), pp. 584–589. IEEE (2019)
32. Neis, P., Zielstra, D.: Generation of a tailored routing network for disabled people based on collaboratively collected geodata. Appl. Geogr. **47**, 70–77 (2014). http://www.sciencedirect.com/science/article/pii/S0143622813002804
33. Kawabata, M., Nishide, R., Ueda, M., Ueshima, S.: Graph-based approach to context-adaptable PNS and its application scenarios. In: 21st International Conference on Data Engineering Workshops (ICDEW 2005), p. 1250 (2005)
34. Rahaman, M.S.: Context-aware mobility analytics and trip planning. Ph.D. dissertation, RMIT University Melbourne (2018)
35. Rahaman, M.S., Mei, Y., Hamilton, M., Salim, F.D.: CAPRA: a contour-based accessible path routing algorithm. Inf. Sci. **385**, 157–173 (2017)
36. Sahelgozin, M., Sadeghi-Niaraki, A., Dareshiri, S.: Proposing a multi-criteria path optimization method in order to provide a ubiquitous pedestrian wayfinding service. Int. Arch. Photogram. Remote Sens. Spat. Inf. Sci. **40**(1), 639 (2015)
37. Pham, Q.D., Deville, Y.: Solving the longest simple path problem with constraint-based techniques. In: Beldiceanu, N., Jussien, N., Pinson, É. (eds.) CPAIOR 2012. LNCS, vol. 7298, pp. 292–306. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29828-8_19
38. Wong, W.Y., Lau, T.P., King, I.: Information retrieval in P2P networks using genetic algorithm. In: Special Interest Tracks and Posters of the 14th International Conference on World Wide Web, pp. 922–923 (2005)

Edge Computing

Shepard: Dynamic Placement of Microservices in the Edge-Cloud Continuum

Farhan Asghar, Tehreem Fatima, Junaid Haroon Siddiqui, Naveed Anwar Bhatti(✉), and Muhammad Hamad Alizai

Lahore University of Management and Sciences LUMS, Lahore, Pakistan
{18030017,18030009,junaid.siddiqui,naveed.bhatti,hamad.alizai}@lums.edu.pk

Abstract. We present Shepard, an innovative microservice placement approach tailored for edge-assisted cloud infrastructures. Shepard dynamically migrates application services between the edge and cloud to harness optimal performance gains. This approach is structured around three core components: (1) a *resource manager* for monitoring available edge resources and the evolving demands of applications, (2) an *optimization module* that transposes the service placement dilemma into a labeled-graph cut challenge, aiming to identify the most advantageous cut given a set of parameters, and (3) a *deployment module* tasked with adjusting service placement in response to shifts in the optimal graph cut's position.

Our implementation of Shepard underwent rigorous testing in two distinct case studies. In the inaugural study, Shepard managed energy for a solar-driven edge within an agricultural IoT framework, resulting in a striking 79% elevation in service reliability and availability compared to a conventional static service placement strategy. For our subsequent study focusing on cost-effectiveness within a ride-hailing application, Shepard facilitated a substantial 45% slash in application deployment expenses, all the while maintaining comparable performance levels to a standard dynamic service placement technique.

Keywords: Edge Computing · Microservices-based Architecture · Dynamic Resource Management

1 Introduction

The evolution of edge computing has created the need for new ways of structuring and deploying application services. An edge is not just a homogeneous extension of cloud resources that can scale up application services on demand, but a precious compute resource near the data source. Intelligently managing this limited yet expensive compute resource is imperative to enhance application performance or extend the advantages of edge to multiple applications (or tenants). As such, deciding what part of the application logic goes into the edge and what stays in the cloud has become a key DevOps challenge.

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 43–62, 2024.
https://doi.org/10.1007/978-3-031-63992-0_4

A static decomposition and placement of services is not feasible because the factors influencing this decision are dynamic, such as the current load of the service, the available resources on the edge and their cost, and the service's performance requirements [43]. Although solutions exist that dynamically scale up or scale down services across data centers or cloud platforms, they are not cognizant of the specific features of edge-assisted clouds. Therefore, developing a practical solution to the service placement problem in this domain holds a much greater significance to garner the unique benefits of edge computing.

Challenges. An edge corresponds to the compute, storage, and network resources available in the request's geographical proximity. Augmenting a cloud with an edge resource is becoming essential to meet the growing latency, availability, bandwidth, and privacy requirements of modern applications, such as artificial intelligence models, augmented and virtual reality, the internet of things, robotics, and video analytics. However, service orchestration and placement strategies in an edge-assisted cloud infrastructure need to address challenges that go well beyond the typical scaling of services within a cloud for the following reasons.

- *Finite Resources:* Resources on the edge cannot anticipate unlimited capacities, and they may vary depending on the characteristics of the edge platform, which could be an extremely resource-constrained smart object, a cellular gateway, or a reasonably well-provisioned small-scale data center (cloudlet). Thus, resources on the edge must be carefully managed and intelligently utilized.
- *Dynamic Requirements:* Application requirements may change over time and space due to fluctuating loads and mobility of requests. A service with sharp latency requirements at one time of the day may become mostly irrelevant during other times when another service comes to the fore [24]. Therefore, service provisioning on the edge must be amenable to such dynamic application requirements.
- *Differential Costs:* Unlike clouds, the cost of the edge may vary based on the current load [23] or ownership. The edge may be owned by the same or different cloud service providers, which may employ surge pricing to achieve supply-demand equilibrium [39], or it may be wholly free but severely resource-constrained if owned by the end-user. Thus, a cost-aware strategy is needed for affordable service orchestration in the edge-assisted cloud.

Scope. The key to addressing this challenge is to determine *when* and *which* portion of the application logic should reside on the edge. Scaling or migrating one service to the edge may require bringing another one back to the cloud to create space for the former or reduce total cost. This may entail continuous monitoring of edge resources and assessing application requirements over time. For instance, in the context of IoT, an occupant of a smart home uses different appliances at different times of the day [31], providing room for dynamically reconfiguring related services on the edge for real-time response. Changing vehicular traffic conditions in a city may provide an opportunity for smart city applications to

migrate the adaptive signal-control service [40] back from the edge to the cloud to reduce deployment costs. A solar-powered edge may accept different loads depending on the predicted energy provisioning patterns [44].

Solution. We present Shepard, a microservices-based service orchestration and placement architecture for edge-assisted cloud infrastructure. The use of microservices provides a fundamental underpinning for decomposing applications into loosely coupled, independent services. The Shepard architecture comprises three components: First, a *resource manager* that (i) monitors available resources on the edge and (ii) gathers statistics about each service's demands and performance. Second, the *optimization module* that maps the services and their interconnections into a graph, thus transforming the service placement problem into a graph cut problem to find the best cut (placement decision for a set of services) given a set of constraints (resource availability on the edge). This module populates the vertices and edges of the graph with information accrued by the resource manager. Third, a *deployment module* that migrates services between the edge and the cloud and reconfigures them whenever the location of the best cut in the graph changes.

Thus, we make the following concrete contributions.

- We propose and implement a novel end-to-end service orchestration architecture for edge-assisted cloud infrastructure.
- We extensively evaluate this architecture in two case-studies and compare its performance with a baseline dynamic placement approach.

Benefits. Our first case-study demonstrates how Shepard improves the availability of services on a solar-powered edge deployed at a remote agricultural farm. We observed up to a 79% decrease in application downtime compared to the original static placement of services, resulting in fewer node deaths, reduced application latency, and increased availability and reliability of services. Our second case-study shows how Shepard reduces application deployment costs on the edge by dynamically migrating unused or overloaded services to the cloud while delivering almost edge-native performance to the end-user. Our results show a 45% reduction in application deployment costs in an edge-assisted cloud infrastructure while ensuring cheaper computation and lower latency than a baseline dynamic service placement strategy.

2 Background and Motivation

Edge Computing. The importance of edge computing in modern cloud infrastructures is undisputed for a wide range of applications. In edge-assisted clouds, the edge serves as the first computing point for data. This allows computations to be performed closer to the source of data, resulting in immediate response times, better learning, and caching, rather than transferring the data to the cloud. Our definition of edge is not limited to a specific platform, as in existing literature [9], but includes any computing unit near the last mile network where

a subset of application services can be offloaded. An edge may be an IoT device, a gateway to the internet, or a slim datacenter nearby.

As a result, service placement challenges may vary depending on the type and capabilities of the edge platform.

Service Placement. Service placement is the process of mapping services (or virtual machines) to available physical hosts within a datacenter or a remote datacenter in a cloud federation. Various efficiency goals dictate these placement strategies. For example, load balancing [11,38] can improve application performance, while minimizing the number of physical machines needed to run the application(s) and shutting down unused ones can enhance energy efficiency [1,3,16].

In contrast to datacenter-based clouds, edge computing does not have the luxury of infinite resources [18,25,45]. The edge may have limited energy, computation, memory, storage, and networking capabilities, as it is designed to serve a subset of users in close geographic proximity. As a result, service placement becomes a challenge. The edge may only be able to host a subset of application services, which must be identified based on a well-defined set of metrics that ensure performance gains. For instance, a low-latency service may be better placed in the cloud if the compute delays on the edge exceed the network delays to the cloud [33]. Additionally, the cost of these services and the availability of resources on the edge may change over time due to varying workloads [32]. There may also be further restrictions on the mobility of services, such as a privacy-preserving service that cannot be moved to the cloud [28].

A service placement strategy in edge-assisted clouds must consider these unique challenges.

Microservices. Microservices-based cloud application designs are a growing trend nowadays [7,27]. Most cloud service providers have successfully transitioned from monolithic architecture to microservices [2,8,26]. This is primarily because microservices resolve traditional limitations of monolithic designs in handling huge codebases, adopting new technology, scaling up deployments, implementing more recent functionality, and more.

Microservices enable applications to be developed as a suite of small services that run in their own processes and communicate using lightweight mechanisms. They are typically packaged in a container that encapsulates the code and all its dependencies, allowing for reliable migration of a service from one computing environment to another and quick resumption.

The introduction of edge computing further emphasizes the need for microservices. Monolithic architectures do not meet the increased flexibility requirements in scaling, profiling, placement, and reconfiguration of application modules. The idea of edge-assisted cloud infrastructure seamlessly integrates with microservices-based application design. The latter enables applications to behave as a single entity, yet have several dissociated parts. These containerized parts can then perform their designated functions irrespective of their location in the system, be it inside the cloud datacenter or at a resource-constrained edge in a different continent.

Consequently, our proposed solution considers microservice-based application designs to keep up with existing trends and unveil the great potential of edge computing.

3 Shepard

We first provide a brief overview of the design space and the associated challenges involved in developing a dynamic service placement solution. We then describe, in detail, the design of each architectural component of Shepard. Finally, we conclude this section by providing the most relevant implementation details.

3.1 Overview and Challenges

Dynamic service placement techniques are commonly employed in data-center based clouds. However, existing approaches are not tailored to the specific needs of an edge-assisted cloud architecture, where an edge is not necessarily a homogeneous extension of existing cloud resources. To achieve the desired performance benefits and the vision of edge computing, the resources on the edge must be intelligently utilized.

Resource Monitoring. In this context, a service placement strategy must account for the dynamic demands of heterogeneous services constrained by resources' availability on the edge. It should identify the most suitable services in the cost-benefit spectrum at any given time and accordingly reconfigure their placement on the fly. To achieve this, we first need to identify candidate services that can be moved from the edge to the cloud and vice versa.

Shepard collects different metrics that define the performance of a service in a given application. These metrics are collected at predefined intervals and recorded into a data store, where they can be queried to assess performance and potentially migrate services if the placement strategy determines a more profitable configuration than the current one. For this purpose, each host runs a lightweight monitoring service that gathers statistics such as the number of calls across pairs of services, the amount of data transferred, CPU usage, average memory consumption, etc. Shepard periodically analyzes service performance and resource utilization on the edge to make placement decisions.

Placement Strategy. A greedy approach to service placement would either place all the services on the cloud or edge, for example, to reduce network bandwidth requirements. Similarly, a load-balancing approach to service placement may proportionally distribute services across the edge and cloud. However, this is not the goal in an edge-assisted cloud infrastructure. We need to find a placement layout that maximizes the desired performance metrics of a given application.

Shepard maps the services onto a labeled graph such that the vertices of the graph represent services and the edges represent the data transfer between different services. The labels on vertices capture the resource demands of a service, while the labels on edges capture the bandwidth requirements, including

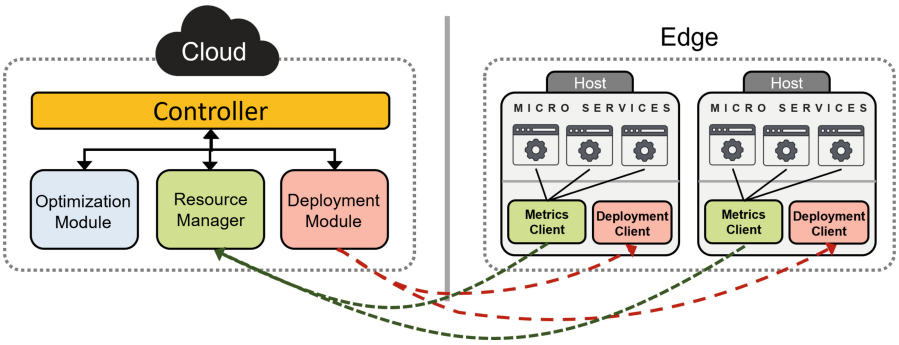


Fig. 1. System Architecture

the number of calls and the amount of data transferred between pairs of services. These labels are periodically updated.

If the application dictates certain latency requirements while minimizing bandwidth requirements, `Shepard` can partition (or cut) the graph in a way that achieves this optimization goal. The latency-critical services will be grouped in a single partition and dispatched to the edge. The best partition (min-cut) in the graph, however, may change over time as the labels are updated. Moreover, the partitioning process must also take into account the resource constraints of edge nodes.

A brute force technique that exhausts the entire search space is a trivial solution to finding the best cut in a graph, but its complexity increases exponentially with the size of the graph. Appropriate heuristics must be employed by `Shepard` to reduce this complexity without compromising the accuracy of finding the best cut. To this end, `Shepard` reduces the size of the graph by aggregating edges, for example, by combining vertices with the highest weights between their edges. The effect of such aggregation naturally places two related services either on the edge or in the cloud while reducing the search space size for finding the best cut in linear time.

Deployment. After the graph has been partitioned, `Shepard` deploys the services according to the determined placement strategy. This may involve scaling up services due to increased demand or moving services between the cloud and edge. There are different migration strategies that can be employed, such as moving services entirely, but this may result in significant network bandwidth consumption. Alternatively, services can be stored locally and activated by `Shepard` to minimize migration overhead, although this may not be feasible for stateful services. `Shepard` uses a combination of these methods to maximize their benefits whenever possible.

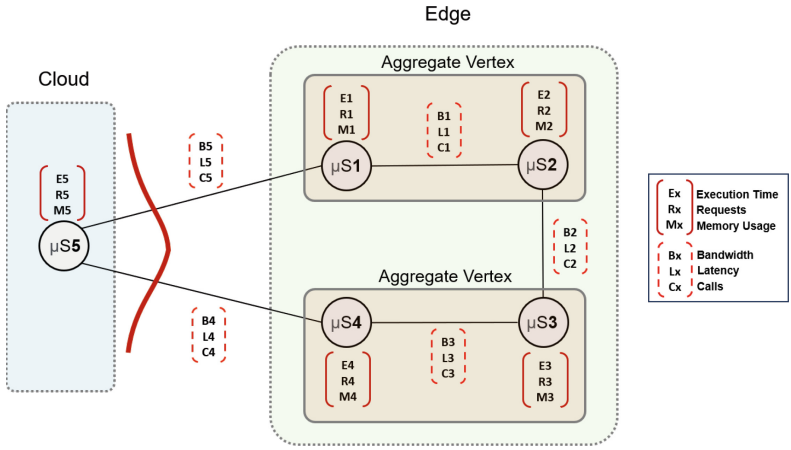


Fig. 2. Capacity-aware multi-weighted min-cut: Visualization of vertex aggregation based on high-weight edges to optimize service placement.

3.2 Architecture

The high-level architecture of `Shepard` is illustrated in Fig. 1. `Shepard` consists of three modules responsible for monitoring resource requirements, optimizing placement decisions, and deploying microservices. These modules are integrated through the `Shepard` controller, which configures them on an application-specific basis and facilitates information exchange among them.

Resource Manager. The Resource Manager module maintains a record of the available resources on the edge and the dynamic resource demands of microservices. It relies on a resource manager client installed on each edge to gather these statistics and relay them back to the main module in the cloud. The metrics of interest for this module include available resources on the edge, such as computing power, memory, storage, and network bandwidth, as well as application performance metrics, such as execution time, network latency, and the number of service calls and data exchanged between service pairs. This information is essential for the optimization module to generate a comprehensive view of the current service deployment configuration and suggest changes to accomplish specific optimization objectives, such as reducing deployment cost or latency.

Optimization Module. To devise an efficient placement plan, the optimization module first constructs a labeled graph using the resource manager's metrics through the following approach.

Graph Construction: Each microservice is depicted as a vertex in the graph, while the edges signify the coupling between services. In Fig. 2, we provide a representative example with six vertices: two on the cloud and four on the edge. The vertices in this figure are marked with the resource demands of the corresponding service, which include memory consumption, execution time, and the

average number of requests to be served. The labels on the edges indicate the communication cost between services. This representation provides a comprehensive view of the application requirements and the extent to which they can be fulfilled on resource-constrained edges. While our illustration centers on some primary constraints, it's worth noting that specific application scenarios might introduce additional requirements. For instance, in settings where edge devices operate on harvested energy, battery levels become a paramount constraint. Similarly, when the cost associated with edge usage fluctuates over time, temporal factors emerge as crucial constraints. The optimization module then finds the optimal cut in the graph, while adhering to the edge constraints, and returns the set of services to be deployed on the edge. The remaining services are deployed on the cloud. A deeper dive into these variable constraints and their implications will be elaborated upon in the evaluation Sect. 4.

Brute Force: A brute force technique for finding the optimal service placement layout involves exhaustively searching the solution space. This method generates all possible partitions in the graph and selects the one that allocates the maximum number of services to the edge while satisfying the resource and cost constraints. However, this straightforward approach does not scale well because it requires exploring a potentially large solution space before identifying the optimal partition.

Algorithm 1. contractEdge

Require: $G = (V, E)$
Ensure: Graph G'
  while Number of nodes in graph $G' >$ THRESHOLD **do**
    $e \leftarrow$ edge from G with the highest weight
    $\{v1, v2\} \leftarrow$ the two vertices associated with edge e
    create a new vertex v
    assign v a weight equal to the sum of weights of $v1$ and $v2$
    $G' \leftarrow G$ with e removed
    Remove $v1$ and $v2$ from G'
    Add vertex v to G'
    for each edge in G' connected to $v1$ or $v2$ **do**
      Redirect the edge to v
    end for
  end while
  return $G' = 0$

Heuristic Algorithm: In light of the scalability challenge posed by the brute force method, we introduce a heuristic approach. This heuristic is rooted in the observation that pairs of services, which frequently communicate or have a high data exchange rate, placing them close (either both on the edge or both on the cloud) can be optimal for performance. Thus, our heuristic aggregates those vertex pairs in the graph with exceedingly weighty edges, considering these unlikely to form a cut. Figure 2 not only demonstrates the graph construction but

also showcases the vertex aggregation of the heuristic method. In this figure, the four vertices on the edge are consolidated into two, highlighting the compactness achieved by our heuristic approach.

Algorithm 1 details the process of aggregating vertices. It iteratively contracts edges with high weights and combines their vertices into a single virtual node until a pre-defined size of the graph is achieved. The compact, aggregated graph is then used as input to the brute force search, which is described in Algorithm 2.

To reinforce the effectiveness of our heuristic, Fig. 3 offers a performance comparison between the brute force method and the heuristic-based by showing the search times for the optimal min-cut value of both algorithms. This figure specifically illustrates the execution times of both algorithms with varying numbers of microservices. For the initial testing and performance evaluations illustrated in Fig. 3, both algorithms made use of synthetic microservices. These synthetic services were designed to replicate typical microservice scenarios and the varying parameters as previously discussed.

It is important to note that the brute force search time rises exponentially with the number of microservices, while the heuristic method exhibits a more favorable linear growth. Remarkably, despite the differences in computational complexity and time, both the brute force and heuristic algorithms converged to the same min-cut value, affirming the heuristic's efficacy.

Algorithm 2. Find Min-Cut

Input: A weighted graph $G = (V, E)$ with node weights W and edge constraints (EC)
Output: Set of services to be deployed on edge (S)

***Initialization*:**
S = {}
***Procedure*:**
G' = contractEdge(G, e)
Find the weights W of all possible cuts $C = (A, A')$ in G' based on edge weights E' where $A'$is deployed on edge
Sort the cuts C in ascending order of weights W
for *cut* in C **do**
    if (Sum of Vertex weights of services $A' \leq$ Edge Node Capability (EC)) **then**
        $S = S \cup A'$
    end if
end for
return S =0

Deployment Module. The deployment module merely updates the service placement layout based on the output of the optimization module. Depending on the underlying platform, this may require migration of services between cloud and edge or only instigating a service stored locally. Further constraints may also apply, such as migration of the state of a state-full service even if the code is stored locally.

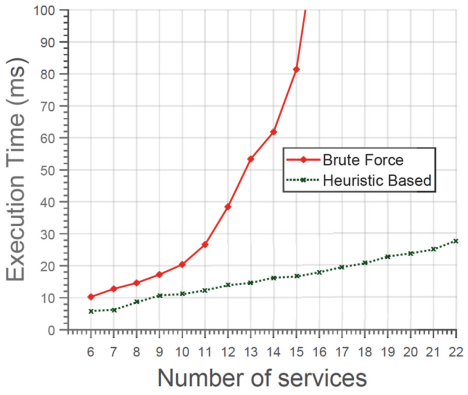


Fig. 3. Performance comparison of a brute force strategy with a heuristic based approach

3.3 Implementation

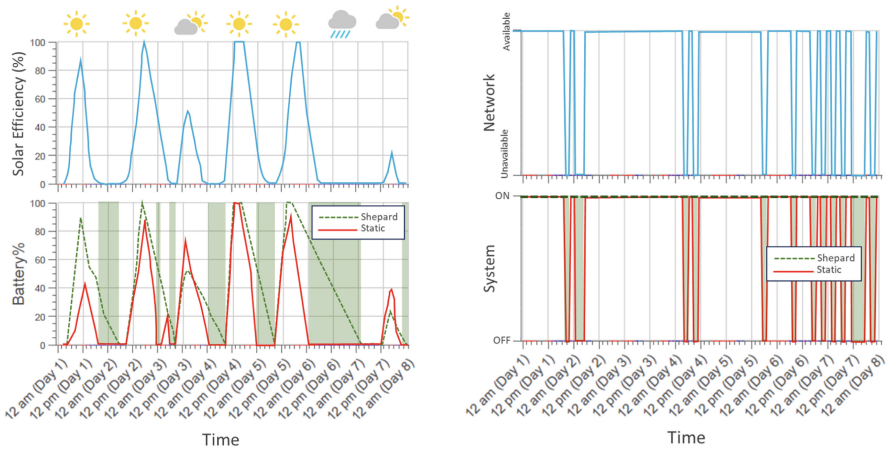
We implemented `Shepard` using Nodejs [10] and utilized Docker containers as microservices, given their popularity in both industry and academia. A container is a standardized software unit that packages code and its dependencies, including the runtime, system tools, system libraries, and settings, to ensure reliable operation on diverse computing environments [5]. Different services communicate with each other using sockets [36], and these Docker containers run on a Docker engine that provides a secure and scalable isolated environment [5,14]. Our implementation assumes that each microservice is running in a separate Docker container [5].

We implemented our *deployment module* using Docker Swarm [4], which is a widely used service orchestration platform. The Swarm Manager represents the cloud, while the Worker represents the edge. When the optimization module proposes a different service layout, Docker Swarm dynamically migrates the services based on the suggested layout.

We implemented our *resource manager module* using the Prometheus [30] monitoring system, which incorporates a time-series database to collect and store metrics from different services. Prometheus can collect metrics from multiple sources, such as those produced by cAdvisor (Container Advisor), which analyzes the resource usage and performance characteristics of running containers, and those generated by the services themselves using a Prometheus exporter. Prometheus can be asynchronously queried for metrics stored in the database.

4 Use-Case Driven Evaluation

We have evaluated the performance of `Shepard` by conducting two case studies for microservice deployment on the edge and have reported on two key performance metrics. Our results demonstrate that the dynamic placement of services,



(a) Battery percentage (b) Network outage

Fig. 4. Shepard notably enhances application availability: The graphs display node deaths and network connectivity for two distinct use-cases: energy and network-outage based deployments as seen in state-of-the-art methods [44]. Green regions mark times when the Heuristic-based node remains ACTIVE, contrasting with periods the static-based node is DEAD. Through dynamic service placement updates, Shepard effectively optimizes the energy budget.

facilitated by Shepard, allows for edge-assisted cloud infrastructure to achieve the optimal balance between cost, energy consumption, and latency while serving user requests from edge devices. Specifically, Shepard can achieve:

- up to 79% reduction in system downtime in a case study of solar-powered edge devices. This shows the effectiveness of Shepard in leveraging the underlying resources available. In this particular use-case, the energy forecast played a significant role in guiding the min-cut algorithm, among other parameters.
- up to 45% reduction in the total cost of deployment when compared to static deployment. For this use-case, the cost was an integral consideration for the min-cut algorithm, enabling cheaper computations at low latency.

4.1 Improving Service Availability

We conceive two case studies to show the improvement in micro-service based application's availability when equipped with Shepard.

Application. Similar to Farmbeats [44], we consider a scenario where microservices are deployed on an edge device for performing precision agriculture. This deployment comprises multiple services performing heterogeneous tasks such as collecting readings from sensors, creating summaries of sensor readings, and controlling a drone to get aerial imagery for creating precision maps from sparse

Table 1. Moveable/Non-moveable Micro-services

Services	Moveable
Orthomosaic creation	✓
Cow monitoring	✓
Storage monitoring	✓
Drone path planning	✓
Drone control	x
Aerial imagery compression	✓
Sensor reading	x
Web interface	✓
Crop suggestions	✓
Cross-farm analytics	x
Precision map creation	✓
Long-term storage	x
Weather forecast	✓

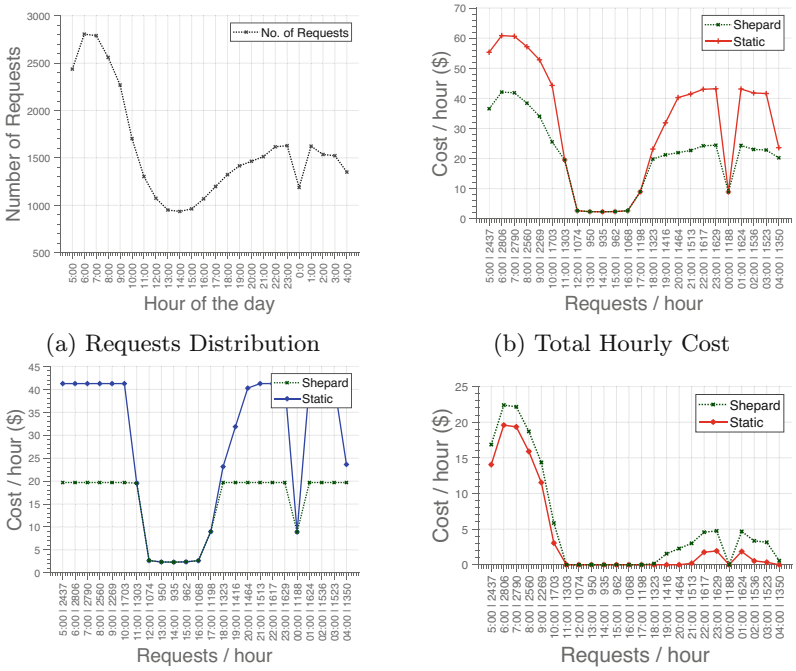
sensors deployment. Table 1 summarizes the services that can or cannot be moved between the edge device and the cloud.

Energy-Aware Dynamic Deployment. We first consider a scenario that takes into account the energy available to the device to perform its tasks.

Use-case. In this scenario, we assume that all services are hosted on a solar-powered edge gateway. Our system uses existing weather data to create a forecast model that predicts node failure based on the current deployment strategy. To do this, we count the consecutive number of hours where hourly energy consumption exceeds the energy gained using solar power. We use the same model for both deployment strategies, and Fig. 4a shows the comparison of system availability in all cases.

Results. As shown in Fig. 4a, a static deployment consumes energy at a very high rate compared to the dynamic deployment. This high energy consumption rate stems from the inability to move services dynamically from edge to cloud when the system can predict a power failure. In contrast, `Shepard` conserves energy by adapting its placement strategy and relocating services to the cloud, ensuring the node remains operational for a longer duration on the given battery charge.

Note that solar efficiency depends on the weather's cloudy nature and varies significantly throughout the day. This affects the strategy employed by the dynamic model, resulting in a different rate of energy consumption. Whenever the input energy is sufficient for the device to charge its battery, the number of services hosted on the edge device increases, increasing the device's energy consumption, as shown in Fig. 4a.



(a) Requests Distribution

(b) Total Hourly Cost

(c) Edge Hourly

(d) Cloud Hourly Cost

Fig. 5. Shepard reduces cost of deployment: By enabling energy- and resource-aware deployment, Shepard allows the requests to be serviced at the edge for as long as the cost of computing on the edge is less than that of the cloud. This protects edge devices from over utilization and reduces deployment cost.

Network Outage. We employ the same application [44] in a different use-case scenario to assess system downtime in the event of network failure caused by harsh weather conditions. It's essential to note that the weather conditions and parameters for this evaluation vary from the earlier energy-aware dynamic deployment assessment, offering insights into the system's adaptability under different challenges.

Use-Case. Harsh weather conditions can cause the network to become unavailable for extended periods [13,17,20]. This demands dynamic reassignment of services on the edge device to maintain network availability. Therefore, we develop a network outage model to predict network failure in harsh weather environments. This model forecasts future network availability, which helps Shepard determine when and which service to deploy on the edge. Figure 4b illustrates a comparison of system availability for a given network outage prediction.

Results. Our results demonstrate that whenever there is a network outage, Shepard can sustain it due to its dynamic nature. It reassigns all services to the edge devices if a predicted network failure occurs due to harsh weather

conditions. This dynamic reassignment enables all requests to be processed on the edge, thereby reducing application latency and increasing availability, as shown in Fig. 4b.

Conclusion. The dynamic reassignment of services in the precision agriculture scenario enables the device to operate during network outages and brings energy efficiency, which plays a critical role in maintaining the availability of an edge-assisted cloud infrastructure. Increased availability reduces application latency, a significant issue for cyber-physical systems with real-time constraints. Therefore, the findings presented emphasize the significance of our dynamic service deployment system in enhancing system efficiency and reliability.

4.2 Reducing Deployment Cost

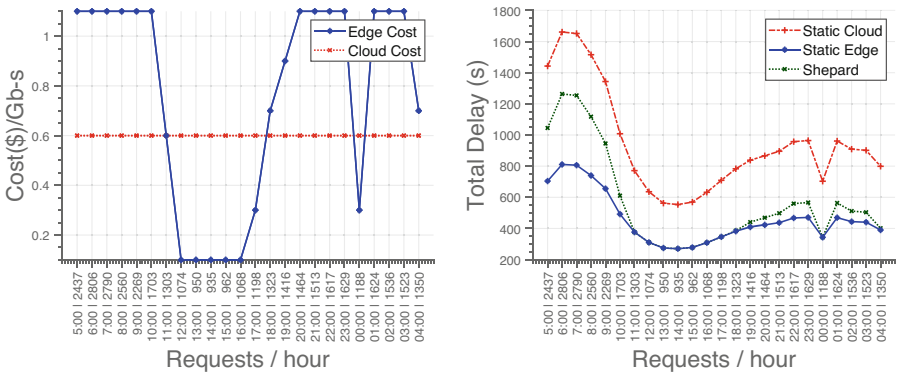
To analyze the impact of `Shepard` on the overall computation cost, we consider data from ride-hailing services such as Uber [41] and Lyft [22].

Application. When a user searches for a suitable ride nearby, multiple services are invoked before returning the final result. These tasks include finding the shortest path between the source and destination, calculating the fare, and more. As this is a distributed application, all these tasks must be performed in a central remote location. With a large user base, this application can put a significant load on the cloud for computation. Additionally, the distance between users and the cloud can introduce communication latency, which can negatively impact the user experience. Thus, an edge device near the user capable of performing the same computation as the cloud can help reduce the cloud's latency and load. However, the introduction of such a device increases the overall cost of the cloud.

Setup. We use 24 h of Uber movement data [42] to determine the number of ride requests sent every hour. The extracted data is shown in Fig. 5a. We consider the computation cost of the cloud and edge as used in existing literature [46]. Moreover, we compare `Shepard`'s dynamic service placement model against a *static* baseline service placement approach where all requests are sent to the edge until it reaches 100% utilization, after which every request is directed towards the cloud.

Impact on Cost. To better understand the cost impact of `Shepard`, we compare the cost of computing on a system equipped with `Shepard` against a static approach where a request is redirected to the cloud only when the edge is being utilized to its full potential.

Edge vs. Cloud Cost. Each request serviced at the cloud costs the same, regardless of the rate at which requests are received. However, the cost of computation changes with the number of requests being serviced by the edge. The higher the utilization, the higher the cost of performing computations at the edge, as shown in Fig. 6a.



(a) Edge vs Cloud Cost Distribution (b) Total Delay

Fig. 6. Deployment cost on edge vs cloud. Shepard can find the sweet spot: For higher number of requests, the cost of computing on the edge is greater than of the edge. Shepard reduces this cost by shifting the load to the cloud while minimal increase in latency.

Edge Cost per Hour. Edge devices are charged an extra amount if their utilization crosses a certain threshold. Static deployment techniques fail to cater to such requirements and utilize the edge to its full potential, moving service to the cloud only when the edge device's utilization reaches 100.

Figure 5c shows the cost of performing computations on the edge device for every hour of the day. When compared with Shepard's deployment, we can observe that the cost of computing on the edge is very high for static deployment whenever the number of requests is high. Unlike clouds, the cost of computing on an edge device depends on the current load; the higher the load, the higher the price. Static deployments utilize the edge device to its full potential, thereby performing computations on the edge during peak hours for a longer duration of time.

Cloud Cost per Hour. The cost of computation of a static deployment is meager at the cloud end of the edge-assisted cloud infrastructure. Since most requests are handled at the edge end, the number of requests serviced by the cloud is very low compared to Shepard. As a result, the cloud's computational cost is low compared to dynamic deployment, as shown in Fig. 5d.

A comparison of the total cost incurred in both scenarios is given in Fig. 5b. Although Shepard's dynamic placement requires more cost for computations, the overall cost of computations for edge-assisted cloud infrastructure is still 20% to 30% lower compared to static placement due to the savings in cost at the edge, as shown in Fig. 5c.

Impact on Latency. While Shepard's dynamic placement reduces cost, it is important to note that placing services on the cloud incurs an additional trans-

mission and propagation delay on every request, resulting in increased application latency. To measure the effect of `Shepard`'s placement strategy on latency, we used data for all trips lasting 30 min in the Uber movement dataset [42].

We calculated the total delay of all requests in every hour and performed these calculations for three different scenarios:

1. Static Cloud: when all requests are serviced from the cloud.
2. Static Edge: when all requests are serviced from the edge.
3. Dynamic Delay: when requests are serviced using `Shepard`'s service placement.

Figure 6b shows the total delay for all three approaches. We can see that `Shepard`'s delay lies between the two static methods, as the static approach cannot adapt to the changing request patterns. By adapting to user requests, `Shepard` always deploys some services on the cloud to find the right balance between cost and latency. Scenarios 1 and 3 are two extreme cases where computations are performed either on the edge or the cloud. While performing computations on edge devices decreases latency, it significantly increases cost, rendering edge-assisted cloud infrastructure infeasible. On the other hand, servicing all requests on the cloud significantly increases the latency at a minimal cost reduction. `Shepard` finds the sweet spot between these two extremes to combine the best of both worlds.

Conclusion. Unlike static approaches, `Shepard` dynamically places services on the edge and cloud according to the changing needs of the application. This allows for minimization of edge utilization during peak hours, resulting in cost reduction. At the same time, `Shepard` carefully chooses the set of services to be deployed on the edge so that application latency remains as low as possible.

5 Related Work

A large body of work investigates service placement in the cloud computing domain. However, to scale down our discussion on related works, we focus on dynamic service placement in edge-assisted cloud infrastructure. We broadly divide related literature into two categories: computational offloading in edges and service orchestration strategies.

5.1 Computational Offloading in Edges

Du et al. [6] have formulated the computational offloading and content caching problem as a joint optimization problem to enhance the operator's profit in queuing cellular network. Samanta et al. [34] have proposed an adaptive service offloading scheme for the mobile edge computing platform in the presence of multiple edge devices to maximize profit and reduce latency. Liu et al. [21] have proposed a game-based approach to model the interaction between the edge cloud and users. The edge cloud sets prices to maximize its revenue while

ensuring that its finite computation capacity is not exceeded. For the given prices, each user locally makes offloading decisions to minimize its own cost, which is defined as latency plus payment.

Zhang et al. [47] have used unmanned air vehicles (UAVs) to assist vehicle-carried terminals in performing complex computational tasks. These tasks can be offloaded onto these UAVs, which would otherwise have been offloaded to the cloud, thus causing long delays. Zhang et al. have formulated an energy-aware optimal resource allocation problem to maximize the utility under an edge-assisted cloud computing environment. Zhu et al. [48] have proposed a cooperative mobile edge computing model to reduce both energy consumption and task execution latency, and have formulated a convex optimization problem to utilize energy efficiently. Li et al. [19] have proposed a similar approach that reduces energy consumption and access delays while deciding the service placement scheme.

The primary focus of these works is on minimizing price and energy consumption. However, focusing only on these two factors is of little benefit if we cannot fit the selected service on a resource-constrained edge device. In contrast, `Shepard` not only minimizes price and energy but also considers the resource constraints of the embedded edge devices, such as memory and energy buffer. These constraints play an essential role in deciding the set of services to be hosted on the cloud and edge.

5.2 Service Orchestration Strategies

Hu et al. [12] propose a binary partitioning algorithm based on a randomized contraction algorithm to reduce inter-service traffic in cloud computing environments. Kiss et al. [15] propose a framework that automatically adjusts the supply of cloud services based on application demands. This framework enables developers to integrate cost and performance optimization mechanisms into their application code using APIs, allowing for dynamic orchestration of services in the cloud computing environment. Selimi et al. [37] utilize the state information of the underlying network to devise a service placement strategy for micro-cloud infrastructure that reduces bandwidth usage and latency for end-users.

Farhadi et al. [9] propose a two-time-scale solution for joint service placement and request scheduling in edge clouds under communication, computation, and storage constraints. Petri et al. [29] propose an orchestration engine that can be located at the network edge and supports the allocation of tasks to edge and/or cloud resources for coordinating complex industrial processes. Using the proposed orchestrator, they have achieved lower task completion times while reducing data transfer times. Sampaio et al. [35] propose a platform-independent runtime adaptation mechanism to reconfigure the placement of microservices based on their communication and resource usage.

While many of these works share similarities with ours in spirit, our approach is unique. We focus on edge-assisted cloud infrastructures, where edges are resource-constrained devices that represent a valuable computational resource. Our system provides an efficient way of finding the optimal service placement

strategy while considering the resource-constrained nature of edge devices. In contrast, these approaches focus on either the cloud computing paradigm or propose adaptations applicable to service orchestration in general.

6 Conclusion

In conclusion, `Shepard` offers a novel approach to microservice placement in edge-assisted cloud infrastructures, leveraging a labeled-graph representation to efficiently find the best graph-cut while considering network bandwidth and edge device constraints. By using heuristics to aggregate edges and vertices of the graph, `Shepard` can effectively scale down the problem of finding the cut, resulting in a significant impact on performance. As demonstrated, `Shepard` can increase service availability in solar-powered edge deployments by leveraging future weather prediction to relocate services, and also reduce the overall cost of service deployment in ride-hailing applications without affecting performance. This highlights the versatility and potential of `Shepard` as a solution for microservice placement in resource-constrained edge-assisted cloud infrastructures. Future work could focus on exploring the effectiveness of `Shepard` in other use cases and evaluating its performance in more extensive deployments.

References

1. Al Shayeji, M.H., Samrajesh, M.: An energy-aware virtual machine migration algorithm. In: 2012 International Conference on Advances in Computing and Communications, pp. 242–246. IEEE (2012)
2. Amazon: Microservices (2019). https://aws.amazon.com/microservices/
3. Beloglazov, A., Buyya, R.: Energy efficient allocation of virtual machines in cloud data centers. In: 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, pp. 577–578. IEEE (2010)
4. Docker: Swarm mode overview (2019). https://docs.docker.com/engine/swarm/
5. Docker: What is a container? (2019). https://www.docker.com/resources/what-container
6. Du, J., Zhao, L., Feng, J., Chu, X., Yu, F.R.: Economical revenue maximization in cache enhanced mobile edge computing. In: 2018 IEEE International Conference on Communications (ICC), pp. 1–6. IEEE (2018)
7. DZone: Enterprises are adopting microservices architectures (2019). https://dzone.com/articles/new-research-shows-63-percent-of-enterprises-are-a
8. DZone: Lessons from the birth of microservices at google (2019). https://dzone.com/articles/lessons-from-the-birth-of-microservices-at-google
9. Farhadi, V., et al.: Service placement and request scheduling for data-intensive applications in edge clouds. In: IEEE INFOCOM 2019-IEEE Conference on Computer Communications, pp. 1279–1287. IEEE (2019)
10. Foundation, O.: Node.js (2019). https://nodejs.org/en/
11. Ghorbani, S., Godfrey, B., Ganjali, Y., Firoozshahian, A.: Micro load balancing in data centers with drill. In: Proceedings of the 14th ACM Workshop on Hot Topics in Networks, p. 17. ACM (2015)

12. Hu, Y., de Laat, C., Zhao, Z.: Optimizing service placement for microservice architecture in clouds. Appl. Sci. **9**(21), 4663 (2019)
13. Ismail, B.I., et al.: Evaluation of docker as edge computing platform. In: 2015 IEEE Conference on Open Systems (ICOS), pp. 130–135. IEEE (2015)
14. Jaramillo, D., Nguyen, D.V., Smart, R.: Leveraging microservices architecture by using docker technology. In: SoutheastCon 2016, pp. 1–5. IEEE (2016)
15. Kiss, T., et al.: Micado-microservice-based cloud application-level dynamic orchestrator. Future Gener. Comput. Syst. **94**, 937–946 (2017)
16. Kord, N., Haghighi, H.: An energy-efficient approach for virtual machine placement in cloud based data centers. In: The 5th Conference on Information and Knowledge Technology, pp. 44–49. IEEE (2013)
17. LeClair, D.: The edge of computing: it's not all about the cloud. Inov. Insights (2014)
18. Lewis, G.A., Echeverría, S., Simanta, S., Bradshaw, B., Root, J.: Cloudlet-based cyber-foraging for mobile systems in resource-constrained edge environments. In: Companion Proceedings of the 36th International Conference on Software Engineering, pp. 412–415. ACM (2014)
19. Li, Y., Wang, S.: An energy-aware edge server placement algorithm in mobile edge computing. In: 2018 IEEE International Conference on Edge Computing (EDGE), pp. 66–73. IEEE (2018)
20. Liu, K., Manangi Ravindrarao, N., Gurudutt, A., Kamaal, T., Divakara, C., Prabhakaran, P.: Software-defined edge cloud framework for resilient multitenant applications. Wirel. Commun. Mob. Comput. **2019** (2019)
21. Liu, M., Liu, Y.: Price-based distributed offloading for mobile-edge computing with computation capacity constraints. IEEE Wirel. Commun. Lett. **7**(3), 420–423 (2017)
22. Lyft: Lyft (2019). https://www.lyft.com/
23. Mazrekaj, A., Shabani, I., Sejdiu, B.: Pricing schemes in cloud computing: an overview. Int. J. Adv. Comput. Sci. Appl. **7**(2), 80–86 (2016)
24. Meiklejohn, C., Miller, H., Lakhani, Z.: Towards a solution to the red wedding problem. In: {USENIX} Workshop on Hot Topics in Edge Computing (HotEdge 2018) (2018)
25. Nastic, S., Truong, H.L., Dustdar, S.: Data and control points: a programming model for resource-constrained IoT cloud edge devices. In: 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 3535–3540. IEEE (2017)
26. Nginx: Adopting microservices at Netflix: Lessons for architectural design (2019). https://www.nginx.com/blog/microservices-at-netflix-architectural-best-practices/
27. Nginx: The future of application development and delivery is now (2019). https://www.nginx.com/resources/library/app-dev-survey/
28. Oberheide, J., Veeraraghavan, K., Cooke, E., Flinn, J., Jahanian, F.: Virtualized in-cloud security services for mobile devices. In: Proceedings of the First Workshop on Virtualization in Mobile Computing, pp. 31–35. ACM (2008)
29. Petri, I., Rana, O., Zamani, A.R., Rezgui, Y.: Edge-cloud orchestration: strategies for service placement and enactment. In: 2019 IEEE International Conference on Cloud Engineering (IC2E), pp. 67–75. IEEE (2019)
30. Prometheus: Prometheus (2019). https://prometheus.io/
31. Qayyum, F., Naeem, M., Khwaja, A.S., Anpalagan, A., Guan, L., Venkatesh, B.: Appliance scheduling optimization in smart home networks. IEEE Access **3**, 2176–2190 (2015)

32. Ramachandran, G.S., Contreras, S.L., Krishnamachari, B., Kozat, U.C., Ye, Y.: Publish-pay-subscribe protocol for payment-driven edge computing. In: 2nd {USENIX} Workshop on Hot Topics in Edge Computing (HotEdge 2019) (2019)
33. Ren, J., Yu, G., Cai, Y., He, Y.: Latency optimization for resource allocation in mobile-edge computation offloading. IEEE Trans. Wireless Commun. **17**(8), 5506–5519 (2018)
34. Samanta, A., Chang, Z.: Adaptive service offloading for revenue maximization in mobile edge computing with delay-constraint. IEEE Internet Things J. **6**(2), 3864–3872 (2019)
35. Sampaio, A.R., Rubin, J., Beschastnikh, I., Rosa, N.S.: Improving microservice-based applications with runtime placement adaptation. J. Internet Serv. Appl. **10**(1), 4 (2019)
36. Scott, J.A.: A Practical Guide to Microservices and Containers. Addison-Wesley, Reading (1972)
37. Selimi, M., Cerdà-Alabern, L., Sánchez-Artigas, M., Freitag, F., Veiga, L.: Practical service placement approach for microservices architecture. In: 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), pp. 401–410. IEEE (2017)
38. Singh, A., Korupolu, M., Mohapatra, D.: Server-storage virtualization: integration and load balancing in data centers. In: Proceedings of the 2008 ACM/IEEE Conference on Supercomputing, p. 53. IEEE Press (2008)
39. Singh, V.K., Dutta, K.: Dynamic price prediction for amazon spot instances. In: 2015 48th Hawaii International Conference on System Sciences, pp. 1513–1520. IEEE (2015)
40. Sorkhoh, I., Ebrahimi, D., Atallah, R., Assi, C.: Workload scheduling in vehicular networks with edge cloud capabilities. IEEE Trans. Veh. Technol. **68**(9), 8472–8486 (2019)
41. Uber: Uber (2019). https://www.uber.com/
42. Uber: Ubermovement (2019). https://movement.uber.com/
43. Vaquero, L.M., Rodero-Merino, L., Buyya, R.: Dynamically scaling applications in the cloud. ACM SIGCOMM Comput. Commun. Rev. **41**(1), 45–52 (2011)
44. Vasisht, D., et al.: FarmBeats: an IoT platform for data-driven agriculture. In: 14th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 2017), pp. 515–529 (2017)
45. Vögler, M., Schleicher, J., Inzinger, C., Nastic, S., Sehic, S., Dustdar, S.: LEONORE–large-scale provisioning of resource-constrained IoT deployments. In: 2015 IEEE Symposium on Service-Oriented System Engineering, pp. 78–87. IEEE (2015)
46. Xu, H., Li, B.: Dynamic cloud pricing for revenue maximization. IEEE Trans. Cloud Comput. **1**(2), 158–171 (2013)
47. Zhang, L., Zhao, Z., Wu, Q., Zhao, H., Xu, H., Wu, X.: Energy-aware dynamic resource allocation in UAV assisted mobile edge computing over social internet of vehicles. IEEE Access **6**, 56700–56715 (2018)
48. Zhu, S., Gui, L., Chen, J., Zhang, Q., Zhang, N.: Cooperative computation offloading for UAVs: a joint radio and computing resource allocation approach. In: 2018 IEEE International Conference on Edge Computing (EDGE), pp. 74–79. IEEE (2018)

Minimizing Data Retrieval Delay in Edge Computing

Kolichala Rajashekar[1(✉)], Souradyuti Paul[1], Sushanta Karmakar[2], and Subhajit Sidhanta[1]

[1] Indian Institute of Technology Bhilai, Bhilai, India
kolichalar@iitbhilai.ac.in
[2] Indian Institute of Technology Guwahati, Guwahati, India

Abstract. For real-time mission-critical applications such as forest fire detection, oil refinery monitoring, etc., the edge computing paradigm is heavily used to process data fetched from IoT devices spread over a considerably large geographical region. For such real-time edge computing applications working under stringent deadlines, the overall *retrieval delay*, i.e., the delay in fetching the data from the IoT devices to the edge servers, needs to be minimized; Otherwise, the retrieval delay in fetching the data from IoT devices distributed over such a large geographical region can be prohibitively large. To achieve the above goal, each IoT device must be assigned to a particular edge server while considering the relative positioning as per the topology of the edge cluster. We prove that the above assignment of IoT devices to an edge cluster, which we denote as the Edge Assignment Problem (EAP), is NP-Hard. Therefore, obtaining a polynomial time solution is infeasible. For the above EAP problem, instead of performing both exploration and exploitation on the search space, state-of-the-art heuristic algorithms will only exploit the search space. As a result, these algorithms are unable to achieve an appreciably large reduction in the overall retrieval delay.

To that end, we propose a Deep Reinforcement Learning-based algorithm that is able to produce a near-optimal assignment of IoT devices to the edge cluster while ensuring that none of the edge servers is overloaded. We motivate and demonstrate our proposed algorithm with the use case of federated learning (FL) - a popular distributed machine learning paradigm that is based on the principle of edge computing such that the clients, i.e., edge servers, train local models from the data obtained from local IoT devices. These local models are further aggregated into a global model at an aggregator (the cloud/fog) by exchanging the model parameters instead of raw data. In that case, an optimal assignment of the IoT devices to each edge device is necessary for reducing the training time for the local models, which in turn, results in reducing the overall delay of federated learning. Using experiments that emulate real-world deployment scenarios, we demonstrate that our algorithm outperforms the state of the art in reducing the retrieval delay in the general edge computing scenario, while also minimizing the local training time in federated learning.

Keywords: IoT · edge computing · reinforcement learning

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 63–85, 2024.
https://doi.org/10.1007/978-3-031-63992-0_5

1 Introduction

We consider real-time edge computing applications, such as traffic monitoring (i.e., the detection of traffic violations), autonomous driving, oil refinery monitoring (i.e., an inspection of fluid levels in oil tanks), forest fire monitoring, etc. [23,37,43]. These applications are usually executed under stringent deadlines specified in the service level agreement (SLAs), for example, roughly 3 s in certain oil refinery monitoring use cases [53]. Typically these applications accept input data generated by IoT devices, such as wireless sensors. The above data is then processed by a cluster of edge servers performing some distributed real-time analytics tasks. From the above group of IoT devices, one or more devices are *assigned* to an edge server. Each edge server in the cluster collects and preprocesses a portion of the data [34]. Minimizing the *retrieval delay*[1] in fetching the data from the IoT devices results in minimizing the overall execution time of a real-time edge computing application. This minimization of retrieval delay will enable the application to satisfy its real-time requirement. Depending on the relative positions of IoT devices in a large geographical region and the topology of the edge cluster, the retrieval delay involved may vary. We consider a distributed deployment where the data from the IoT devices are retrieved to the edge servers in the cluster parallel. Hence, the overall retrieval delay for fetching the data is given as the maximum retrieval delay for fetching data from individual IoT devices. Our goal is to minimize this overall retrieval delay experienced by the end user, which we hereon denote as the retrieval delay. The growing number of IoT devices, in turn, will cause a rapid rise in the overall retrieval delay [10]. Hence, to enable real-time performance, the IoT devices must be assigned to individual edge servers in a topology-aware manner. This, in turn, minimizes the retrieval delay.

The above goal of minimizing the retrieval delay is particularly important for federated learning since the local training time, i.e., the time taken for training the local models, is significantly impacted by the retrieval delay. We empirically prove this later in the Evaluation Section of the paper. Federated learning has steadily risen to the ranks of one of the most widely adopted ML approaches because of its unique ability to circumvent data privacy issues by training on local datasets, as well as its obvious performance scalability due to inherent distributed and decentralized architecture. Federated learning is now becoming a "go-to" choice for running distributed machine learning workloads on an edge computing infrastructure, where the edge servers act as clients that train the local models using the data collected from the IoT devices in its proximity [50]. Though client (i.e., edge device) selection is a widely studied topic, the prior literature has not addressed the issue of appropriate assignment of IoT devices to each edge server, which would affect the local training time significantly.

[1] We consider the retrieval delay as the delay in fetching the data at an edge server from its corresponding IoT device.

To that end, this paper addresses the problem of an *edge assignment*[2], i.e., configuration of a given edge cluster in such a way that the IoT devices are assigned to individual edge servers such that the retrieval delay is minimized. At the same time, our proposed algorithm must not overload any edge server in the cluster by assigning too many IoT devices to it. This is because the edge server is resource-constrained and can be easily overloaded by workload from multiple IoT devices [44]. In this paper, we define an *optimal* edge assignment, which results in the minimization of the retrieval delay while ensuring that none of the edge servers is overloaded.

We formulate the above problem as a version of the bottleneck assignment problem which we denote as the *Edge Assignment Problem* (EAP). We prove that the *EAP* is NP-Hard in nature. Though, to the best of our knowledge, this exact problem has not been addressed in the prior literature, researchers have proposed various heuristics to obtain a near-optimal solution for similar optimization problems [14].

For a smaller topology, i.e., a network topology comprising a smaller set of IoT and edge servers, state-of-the-art tools such as Google OR-Tool [9] can solve the problem in a reasonable amount of time. If the number of devices is considerably large to the extent thousands of devices spread out across a very large region in a realistic scenario such as an oil refinery monitoring, we demonstrate that such solvers take a much longer time to reach a solution [33], Further, several heuristic algorithms have been proposed to solve different variations of the assignment problem, but some of the algorithms get stuck at local optima [16,27]. There are also several prior works which have focused on the edge server selection with respect to the computational capability of an edge server [22].

In that case, *reinforcement learning* (RL) is used to obtain a near-optimal solution to the assignment problem [32,36]. Following their lead, we also leverage reinforcement learning approaches, specifically *Deep−Q learning* [46] to provide a near-optimal solution to EAP. The contributions of the paper are as follows.

- We propose an algorithm that provides a near-optimal solution to the edge assignment problem, assigning a group of IoT devices to the edge cluster while ensuring that none of the edge servers is overloaded.
- Using experiments that emulate real-world deployment scenarios, we demonstrate that our algorithm outperforms the baseline approaches in minimizing the retrieval delay.
- We also show that the assignment results can help achieve a significant reduction of the local training time of federated learning through evaluation.

The rest of the paper is organized as follows. In Sect. 3 and Sect. 4, we present the problem definition and system model. In Sect. 5, we present the proposed algorithm in detail. In Sect. 6, we present the experimental evaluation and insights obtained. Section 7 summarizes the paper and outlines future work.

[2] We denote an assignment as a possible connection of the IoT devices to the edge servers in the cluster in a specific fashion.

2 Related Work

In this section, we review the previous works which are related to minimizing the retrieval delay in edge computing and federated learning scenarios.

Yousefpour et al. [51] proposed a policy for minimizing service latency in fog computing networks. Sun et al. [42] proposed a network architecture based on Cloudlet, which minimizes the average response time of mobile users by offloading the workloads. Kherraf et al. [15] proposed workload assignment of IoT devices to the edge server using Tabu search (TS). Nijimbere et al. [32] show that TS will get stuck at a local optimum. Perikan et al. [34] presented an approach for the assignment of an IoT device to the edge server without any discussion about minimizing communication latency issues between IoT and the edge server. Qiang et al. [10] proposed an algorithm called Application Aware Workload Allocation ($AREA$) for assigning the IoT users' requests to the cloudlet to reduce the response time from the cloudlet. Song et al. [42] presented an approach for the periodical distribution of incoming tasks at an edge computing network and satisfying the Quality of Service (QoS) requirements like completion time. However, they did not focus on real-time edge computing applications where the response time is crucial.

Numerous studies on workload offloading in a MEC infrastructure have emphasized end-to-end latency. Sun and Ansari [45] examined a network in which numerous users outsource their tasks to geographically dispersed MEC nodes. They provided a system for latency-aware offloading that lowers the overall response time caused by the workload of the users. In [47], for instance, the authors evaluated the trade-off between energy usage and latency by solving the IoT workload offloading problem. Their suggested approach improves system utility and energy usage while meeting latency requirements.

By jointly optimizing offloading decisions and resource allocation, Sheng et al. [39] suggested a computation offloading system to reduce the average offloading latency of users. Lin et al. [20] presented a double offloading approach in which neighboring edges shared their resources and jointly offloaded user queries in order to more efficiently utilize their communication and computing resources. Nevertheless, this research predominantly employed custom-built heuristic techniques that cannot be generalized. Recent research has demonstrated that deep reinforcement learning has considerable potential for handling complicated decision-making challenges in edge/cloud computing systems. Hua et al. [11] surveyed machine learning integration with edge computing. In this survey, they also reviewed the deep reinforcement learning solutions for optimizing edge computing scenarios with respect to reducing latency. Huang et al. [12] proposed a deep reinforcement learning-based offloading algorithm to evaluate whether tasks for a set of mobile devices should be locally processed or offloaded, and how to allocate wireless bandwidth to optimal performance. Lie et al. [22] proposed a deep reinforcement learning solution for edge server selection. In this selection of edge servers, they consider the aspect of computation offloading such as which edge server to be selected in order to perform computation. However,

they do not consider the edge server selection based on the delay incurred in fetching the data from IoT devices.

Nonetheless, these investigations consider minimizing the workload delay and focusing on reliability without considering the delay incurred in fetching the data from IoT devices. None of the mentioned work talks about the assignment of IoT devices to the edge servers based on the fetching delay incurred. In our work, we focused on minimizing overall retrieval delay incurred from the IoT devices to the edge servers.

Minimizing Data Retrieval Time for Federated Learning: The rate at which data reaches local devices, in our case edge servers, can have an effect on the training time of federated learning. Edge servers train the model using their own local data and then communicate the updated model parameters to a centralized server for aggregation [17,19].

In numerous ways, the training performance of federated learning can be improved if the data comes fast to the local device. Firstly, it can shorten the time required to train the model locally, enabling the device to contribute more frequently to the training process. Secondly, it can enhance the quality of local training since the model can learn from more recent data that more accurately reflects the current state of the system [13,17]. Overall, minimizing retrieval data is a crucial performance aspect of federated learning [19].

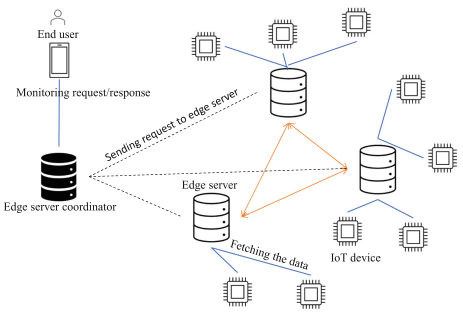


Fig. 1. Edge computing environment

3 Problem Statement: The *EAP* is NP-Hard

In this section, we formally define *the Edge Assignment Problem* (EAP) which we mentioned already in Sect. 1.

Let there be m edge servers and n IoT devices in the network. We denote $I = \{1, 2, \ldots, m\}$ and $J = \{1, 2, \ldots, n\}$; also, $i \in I$ and $j \in J$ denote the indices of an edge server and an IoT device. Let:

– p_{ij}, r_{ij} denote the *retrieval delay* and the *load* on the ith edge server, when the jth IoT device is connected to the ith edge server.

- b_i denote the *maximum load* the ith edge server can sustain.
- x_{ij} be a binary variable to denote if the jth IoT device is connected to the ith edge server.

Now the problem is: given $\{p_{ij}\}$, $\{r_{ij}\}$ and $\{b_i\}$, determine:

$$\min\{\max_{i\in I, j\in J}\{p_{ij}\cdot x_{ij}\}\} \tag{1}$$

subject to:

$$\sum_{j\in J} r_{ij}\cdot x_{ij} \leq b_i, \forall i \in I, \tag{2}$$

$$\sum_{i\in I} x_{ij} = 1, \forall j \in J, \tag{3}$$

$$x_{ij} \in \{0,1\}. \tag{4}$$

Constraint Eq. (2) says the sum of the loads on each edge server should not exceed its maximum capacity. Constraint Eq. (3) says each IoT device must be connected to exactly one edge server.

The EAP as described above is a well-studied problem under the name of the BGAP; the BGAP stands for the *Bottleneck (version of the) Generalized Assignment Problem (GAP)*. Both the GAP and the BGAP belong in the complexity class NP-Hard [25,26,28]. They are NP-Hard in the strong sense, which means that – even if we ignore the objective function (i.e., (1)) in both the problems – it is even hard to determine the existence of an assignment for $\{x_{ij}\}$ that may satisfy (2). For the sake of completeness, we reproduce the proof below.

Theorem 1. *The EAP is NP-Hard.*

Proof. We reduce the NP-Complete 2-set-partition problem to the EAP. The input to the algorithm that solves 2-set-partition is a set S containing n non-negative real numbers: $\{A_1, A_2, ..., A_n\}$; the output is 1, iff there exists a subset S_1, such that $sum(S_1) = sum(S \setminus S_1)$. In the EAP we set $m = 2$, $A_j = r_{1j} = r_{2j}, \forall j \in J$ and $b_1 = b_2 = \frac{\sum_{j\in J} A_j}{2}$. While mapping the instances of 2-set-partition to those of the EAP, we ignore b_1, b_2 and $\{p_{ij}\}$. Suppose, the algorithm $\mathcal{A}$ solves the EAP; it does so by outputting the optimal value, or by outputting $\perp$ denoting that no solution exists. It is easy to see that $\mathcal{A}$ also solves the 2-set-partition as follows:

1. If it outputs $A \neq \perp$ then set partition exists.
2. If it outputs $A = \perp$ then the set partition does not exist.

The above proof also establishes that even if we relax the EAP with $r_{1j=r2j}$ (call this problem EAP′), then EAP′ is also NP-Hard.

Let us concentrate on the following special cases of EAP′: $r_{ij} = 1$, $b_i = \mathcal{T}$, and $n = \mathcal{T}m, \forall i, j$. Interestingly, when $\mathcal{T} = 1$ (call this problem EAP″) then it is poly-time solvable [26]; but when $\mathcal{T} > 1$ (call this problem EAP‴), the

problem seems hard (although the proof is missing). In our case: we assumed that $\mathcal{T} > 1$.[3]

In summary, 2-set-partition $\leq_P$ EAP$'$ $\leq_P$ EAP; EAP$''$ $\leq_P$ EAP$'''$ $\leq_P$ EAP$'$ $\leq_P$ EAP. The EAP$''$ is poly-time solvable; the EAP$'$ and EAP are both NP-Hard; but the status of EAP$'''$ is not yet known. We have done extensive analysis of EAP$'''$, and have conjectured that it is also NP-Hard. Due to space constraints, here we omit the full discussion which will appear in the extended version.[4]. In the remainder of the paper, we solve EAP$'''$ using heuristics and machine learning techniques.

4 System Model

We consider an intelligent edge computing environment comprising IoT devices and edge servers as shown in Fig. 1. We assume that the IoT devices are equipped to do the following jobs: filtering, preprocessing, and transmitting data to the edge server for analysis purposes [29]. Edge servers host containerized applications or services intended to process the data from the respective IoT devices [2]. Each IoT device is connected to an edge server via a wired or wireless medium.

An end user submits a service request to the edge cluster. The request is handled by the *coordinator*, i.e., a designated edge server, which subsequently schedules the request to one or more edge servers. The above edge servers then proceed to retrieve the required data from one or more IoT devices. We consider the effect of the above retrieval delay, defined earlier, often referred to as the *communication time* [18], which has a significant effect on the overall turnaround time of the service [6]. Retrieval delay can be determined in a practical IoT deployment by employing a network monitoring tool in the edge server.[5]

We give special consideration to federated learning, which is a special case of edge computing and distributed machine learning, in which each edge server (i.e., local client) trains a local model using training data collected from IoT devices located close to said edge server. The observed local training time, i.e., the end-to-end delay in learning the local models, is directly proportional to the retrieval delay, i.e., the delay experienced by the edge servers in collecting the training data from the IoT devices assigned to it. Hence, an optimal assignment would minimize the above retrieval delay, which in turn, would result in a reduction of the overall local training time. In this paper, we work with the following assumptions.

- A-1: All edge servers and IoT devices are stationary, i.e., they rarely change their geographical locations. This assumption is justified by a host of edge

[3] For example, $\mathcal{T} = 2$ means that the maximum number of IoT devices that can be assigned to an edge server is 2.

[4] The conjecture is based on the fact that $\binom{\mathcal{T}n}{m}$ is constant when $\mathcal{T} = 1$, but exponential when $\mathcal{T} > 1$.

[5] https://www.cisco.com/c/en/us/products/cloud-systems-management/iox/index.html.

computing applications, such as oil refinery monitoring [38], traffic monitoring [5] etc., where edge servers are stationary in nature.[6]

– A-2: We consider a threshold to the number of IoT devices that can be connected to an edge server. We assume that this threshold is identical for all the edge servers. The edge servers perform certain analytics workloads on the data retrieved from the IoT devices, and in doing so they require some computing and storage resources. Such resources are typically limited within a pre-defined threshold for edge servers. Hence, the above resource limitations, i.e., the thresholds provided by the hardware manufacturers or the end users themselves, must be considered as an additional constraint in the given optimization problem of assigning the IoT devices to the edge servers [6].

5 Reinforcement Learning (RL) for EAP

In this section, we elaborate on the reasoning behind our decision to use *RL*-based algorithms to solve the given *EAP*. Though in the prior literature, traditional heuristic algorithms have been proposed to solve the *assignment problem*, which is similar to Eq. (1), most of these algorithms have been unable to show improvements beyond local optima [16,27,32]. Further, according to *No Free Lunch* (*NFL*) theorem, a single heuristic is not suitable for solving multiple optimization problems [49]. To that end, many researchers have proposed machine learning-based approaches to solve different variations of the assignment problem [21,33]. However, classical machine learning techniques are generally used to solve a problem where it is safe to assume that data are independent and identically distributed (*i.i.d*). On the other hand, in our case, prior information about the assignments of the IoT devices to the edge servers is not readily available. Hence, there is a lack of availability of training data to build a classical machine learning model. Because of the above reason coupled with the absence of any prior information about the distribution of the data, the classical machine learning algorithms are not suitable for our problem. Hence according to [46], the defined problem requires a trial and error-based approach to learning. Specifically, researchers have used Reinforcement Learning (*RL*) algorithms in the past to solve complex optimization problems without relying on training data [4,22]. Following [16,27,32], we express the *EAP* as a *Reinforcement Learning* (*RL*) problem.

5.1 The MDP Model for EAP

Here, we model *EAP* as *Markov Decision Process (MDP)* as follows.

– **State space** (S)**:** A state $s \in S$ is an n-dimensional vector, $s = [s_1, s_2, \ldots, s_n]$ where $\forall j \in J$, $s_j \in I$, represents the edge server which the jth IoT device is assigned to. If $s_j = 0$, the jth IoT device has not yet been

[6] In future work, we consider the mobility of IoT devices and edge servers which makes the problem more challenging.

assigned to any edge server. If $s_j = i$, where $i \in I$, the jth IoT device has been assigned to the ith edge server. We define s^T as a *terminal state* where all the IoT devices have been assigned, i.e., $\forall j \in J, s_j \neq 0$
For example, let us consider a problem instance with n = 4 IoT devices and m = 2 edge servers. A possible state vector s could be: $s = [0, 1, 0, 2]$. In this example, the first and third IoT devices have not yet been assigned to any edge server ($s_1 = 0$ and $s_3 = 0$). The second IoT device has been assigned to the first edge server ($s_2 = 1$), and the fourth IoT device has been assigned to the second edge server ($s_4 = 2$).

- **Action space (A):** An action $a \in A$ corresponds to assigning an IoT device to an edge server. Action space of the system A is defined as $A = \{a | a \in \{x_{11}, x_{12}, x_{13} \cdots, x_{nm}\}\}$. Generally, an agent interacts with the environment and takes appropriate action. Here, an action space of the system is the set of all *possible* assignments of IoT devices that an agent is going to take on the basis of the observed state. In each state, the agent can choose from a subset of actions in A, considering only the unassigned IoT devices and ensuring the threshold of the edge server given in Eq. (2).
  For example, let us consider a problem instance with n = 3 IoT devices and m = 2 edge servers. The complete action space A can be defined as $A = \{a | a \in \{x_{11}, x_{12}, x_{21}, x_{22}, x31, x_{32}\}\}$. For instance, if IoT device 1 has already been assigned to an edge server and the threshold of edge server 2 is reached, then the available actions for the RL agent in this state would be a subset of A, i.e., $a \in \{x_{21}, x_{31}\}$. The action space at the terminal state s^T is null because no further actions are required as all the IoT devices have been assigned to the edge servers.
- **Transition probabilities** P**:** The transition probability function $P(s'|s, a)$ defines the probability of transitioning from state s to state s' when action a is performed. In this problem, transitions between states are deterministic. If action a is valid i.e., assigning the IoT device to the edge server doesn't violate the threshold constraint, then the probability of transitioning to state s' is 1, and 0 otherwise.
- **Reward (R):** A reward is a scalar indication of an agent's performance towards achieving the goal. The goal here is to minimize the maximum retrieval delay such that none of the edge servers is overloaded as per (1). In our MDP, the reward function R(s, a, s') returns the immediate reward for transitioning from state s to state s' by taking action a which is defined as follows
  - $R(s, a, s') = -M$, where M arbitrarily large constant. If the action is invalid, i.e., the IoT device j has already been assigned, or the capacity constraint of the edge server i is violated, the reward function returns a large negative reward $-M$ to discourage the agent from taking invalid actions.
  - If the action is valid and all IoT devices have been assigned to edge servers after taking the action, the reward function returns the negative maximum retrieval delay in the new state s'. In this case, the reward function is defined as $R(s, a, s') = -max_delay(s')$, where $max_delay(s')$

is the maximum retrieval delay observed among all IoT devices for a particular assignment in state s'.

- $R(s, a, s') = 0$, If the action is valid, but not all IoT devices have been assigned to edge servers after taking the action.

The reason for choosing the above reward function follows from our original goal to minimize the maximum retrieval delay incurred in retrieving the data from the IoT devices to the edge servers. At the same time, the *RL* agent is discouraged to perform invalid actions.

5.2 Preliminaries

Here, we introduce the various concepts and terminology we have used in RL algorithms.

- **Policy (π):** A policy is a function that maps the state space to the action space. In our case, the policy is deterministic and represented as a function $\pi : S \rightarrow A$. In the case of EAP, the deterministic policy will provide the best action, i.e., the particular assignment to perform in each state on the basis of the current assignment and the threshold on each edge server. The aim of agent learning is to determine an optimal edge assignment defined earlier. An agent uses the policy to learn the optimal actions to reach a solution to the given problem. A policy should consider the exploration and exploitation trade-off in training the agent [46]. For the given problem, we consider the ϵ-*greedy* policy since it ensures that there is a healthy balance of exploitation[7] and exploration of the state-action pairs by the agent. We can configure the ϵ-*greedy* policy to perform *exploration* and *exploitation* with a probability $\frac{\epsilon}{|A(S)|}$ and $1 - \epsilon + \frac{\epsilon}{|A(S)|}$, respectively.
- **Action-Value (Q):** In RL, we use a value function to define the expected future rewards in terms of the current state or state-action pair. Since, we do not know the full dynamics of the environment, i.e., the probability of the action taken in a given state, dynamic programming solutions cannot be applied. Therefore, action-value function (Q) is used instead of the state-value function (V) [3,46]. As per the standard practice in RL [41], the action-value function $Q_\pi(s, a)$ is expressed in terms of the following Bellman equation.

$$Q_\pi(s, a) = Q_\pi(s, a) + \alpha \left[R_{t+1} + \gamma Q_\pi(s', a') - Q_\pi(s, a)\right] \tag{5}$$

where π denotes the policy, $a \in A$ denotes the action, $s \in S$ denotes the state, and γ denotes the discount factor, which quantifies the relative importance assigned to the future action rewards with respect to the current action reward. Therefore, Eq. (5) estimates the action-value based on the immediate reward obtained based on action a, and discounts the estimated action value

[7] Here exploitation means acting greedily with respect to the action-value function [31].

by *discount factor* (γ). If we use max on discounted action-value, action-value $Q(s,a)$ is defined as

$$Q_\pi(s,a) = Q_\pi(s,a) + \alpha \left[R_{t+1} + \gamma \max_{a'} Q_\pi(s',a') - Q_\pi(s,a) \right] \tag{6}$$

In Eq. 6, the expression $R_{t+1} + \gamma \max_{a'} Q_\pi(s',a')$ is called as *temporal difference target*. The optimal policy is defined as

$$\pi^*(a|s) = \arg\max_{a \in A} Q^*(s,a) \tag{7}$$

where $Q^*(s,a)$ is an optimal action value. This optimal policy $\pi^*(a|s)$ gives an action $a \in A$ where the estimated action-value $Q^(s,a)$ is maximum. As the agent does not know about the environment, we make use of the time difference of learning without any prior information about the environment [46]. The value of the state (V) under an optimal policy is expressed as *Bellman optimality equation* as follows.

$$v_*(s) = \max_{a \in A} Q_{\pi*}(s,a) \tag{8}$$

– **Episode:** Owing to the exploratory nature of RL algorithms, the agent continues the learning in the form of a series of episodes - each of which represents a particular sequence of tuples $(state, action, reward)$ generated on the basis of the policy followed. This sequence will continue till the terminal state s^T.

As elaborated with examples in Sect. 5.1, the state space in our case is a collection of state vectors, where each state vector is given as a finite set of integers. Similarly, the action space is given as a set of binary variables x_{ij} where each $x_{ij} \in \{0,1\}$. Hence, both state space and action space are discrete in nature[8]. We consider real-world large-scale edge deployments, where the number of edge servers and IoT devices is typically very large. On the other hand, the popularly used reward-based learning algorithms such as tabular methods are able to produce an optimal solution only for smaller discrete state space and action space [46]. Among reward-based learning methods, *Temporal Difference* (TD) learning is widely used in practice [8,40]. Popular tabular TD algorithms, such as $SARSA$ and *Q-learning*, have been frequently applied by researchers in the systems and networking domain [24]. However, with an increasing number of IoT devices and edge servers commonly encountered in our case, state and action space becomes too large to handle according to the above techniques.

As per [30], the above issue is tackled commonly by applying function approximation, and among the existing function approximation techniques, researchers have commonly used *neural networks*. Particularly for very large and discrete state space and action space, Deep Q network (DQN) is a popular choice to approximate the value function for each possible action of a state. In the next subsection, we elaborate on our DQN-based RL algorithm for EAP.

[8] In future work, we consider continuous state and action spaces since the edge servers and IoT devices are going to be non-stationary.

5.3 Deep Reinforcement Learning (DQN) for EAP

In DQN, the neural network receives the current state as input and generates a Q-value for each possible action. The action with the highest Q-value is selected, and the agent interacts with the environment by performing that action and observing the subsequent reward and state. The following Q function is used by the agent to update the Q-values.

$$Q(s, a, \theta) \approx Q^*(s, a), \tag{9}$$

where $Q(s, a, \theta)$ denotes the Q value of taking the action in a given state s, and is estimated with respect to the weights of the neural network θ. As a result of the above interactions with the environment, the agent learns from experience by iteratively changing its Q-function based on the reward assigned as per Subsect. 5.1.

Nevertheless, in most RL algorithms, the training data samples are frequently strongly correlated, resulting in instability and unsatisfactory performance [46]. To tackle the issue of instability, a *replay buffer* is typically used to store the agent's past experiences and perform random sampling randomly on it during training. Each experience is saved as a tuple (s, a, r, s') comprising the current state s, the action performed a, the resultant reward r, and the future state s'. As an agent interacts with its surroundings, these tuples are added to the replay buffer. During training, a random sample of events from the replay buffer is used to update the Q-function. By sampling at random from the replay buffer, the training data become less correlated and more representative of the actual distribution of events.

Deep Q-learning uses two networks to stabilize the training process from overestimation bias of action-value values, i.e., the policy network and the target network. Throughout the training process, the policy network is used to determine actions, and its weights are adjusted with each iteration to increase its performance, such as training stability. In contrast, the target network is used to evaluate the Q-values of the policy network, and its weights θ_t are fixed for a specified number of iterations before the agent starts to adjust θ_t to match it with the weights of the policy network θ. In the absence of the target network, the policy network would be modified depending on the estimated Q-values at each iteration, resulting in a feedback loop that can lead to overestimation bias and instability in the training process. During the training, the following standard loss function $L(\theta)$ is being used.

$$L(\theta) = E[(r + \gamma max_{a'} Q(s', a'; \theta_t) - Q(s, a; \theta))^2] \tag{10}$$

The DQN algorithm for agent training is shown in Algorithm 1. In lines 2–4, we initialize replay memory D with maximum capacity C. Replay memory stores the transitions (s, a, r, s') that the agent observes during its interactions with the environment. It is used to sample batches of transitions to train the Q-network - a neural network with random weights θ. The Q-network accepts as input a state s and outputs a Q-value for each possible action a. The action a taken in state s

Algorithm 1. Deep Q-Learning for EAP

Input: $\{p_{i,j}\}, n$ and $\mathcal{T}$ ▷ Inputs to the algorithm
Initialization : ▷ Initializing Reply buffer, Q network, Target network, and hyper parameters
Initialize Replay Buffer D with maximum capacity C
Initialize Q-network with random weights θ, target network with $\theta_t = \theta$
Initialize Parameters : exploration rate ϵ, exploration decay ϵ_{decay}, discount factor γ, learning rate α, number of episodes E, update frequency of target network U
Procedure:
for episode 1 to E **do**
    reset the state ▷ Here all the IoT devices are unassigned with edge servers
    set the episode reward $\mathcal{R} = 0$
    while episode not done **do**
        With probability ϵ select random action a ▷ assign IoT to any available edge server randomly
        Otherwise select action $a = \arg\max_a Q(s, a; \theta)$ ▷ assign IoT device to an edge server which results in maximum action value estimate
        Take action a, and observe R(s, a, s') and s'
        Store transition $(s, a, R(s, a, s'), s')$ in replay memory D
        Sample minibatch of transitions from D
        if episode not done **then**
            $Q_{target} = R(s, a) + \gamma * max'_a Q(s', a'; \theta_t)$
        else:
            $Q_{target} = R(s, a)$
        end if
        Update Q-network weights θ by minimizing loss: $L(\theta) = (Q_{target} - Q(s, a; \theta))^2$
        Every U steps, update $\theta_t = \theta$
        $\epsilon = \epsilon * \epsilon_{decay}$
        s = s'
        R = R + r
    end while
end for

is based on the estimated Q-value, and this estimation is performed by a second neural network called the target Q-network since it is used to generate the target Q-values for training the Q-network. The weights of the target Q-network are updated less frequently than the Q-network's weights to make the training more stable.

Line 5 sets the hyperparameters used for training the DQN agent. The discount factor γ determines the importance of future rewards. The learning rate α controls how much the weights of the Q-network are updated in response to the training data. The exploration probability ϵ determines the probability that the agent selects a random action instead of the action with the highest Q-value. The exploration decay rate ϵ_{decay} determines how quickly the exploration prob-

ability decreases over time. The update frequency U determines how often the weights of the target Q-network are updated.

Lines 6–22 explain the steps undertaken in training the DQN agent. For each episode, we reset the state of an environment and set the reward of the episode to 0. For each step in an episode, we select a random action a based on the exploration probability ϵ. Otherwise, the action with the highest Q-value is selected. Here, each action corresponds to a particular assignment of an IoT device to a given edge server. The agent performs the action a and observes a reward r and the next state s'. The transition tuple (s, a, r, s') is stored in replay memory. Now, we sample a minibatch of transitions from the replay buffer to train the Q-network, and subsequently, the target Q values for training the Q-network are computed. If an episode is unfinished, the target Q-value is expressed in terms of the immediate reward plus the discounted maximum Q-value, otherwise, the immediate reward itself is assigned to the target Q-value. Next, we update the Q-network weights θ while minimizing the loss given in 10. In lines 22–25, for every step U, we update the weights of the target network θ_t with the weights of the Q network θ, followed by an update of the exploration probability, the state, and the reward.

6 Experimental Evaluation

6.1 Experimental Setup

We run the simulation on a six-core processor (3.6 GHz), with a memory of 32 GB and an Ubuntu 20.04 LTS. To demonstrate the performance of our proposed *RL* algorithms, we created a custom environment using Gymnasium[9] - Python toolkit widely used for this purpose. For realistic analysis, the values of retrieval delays used in the experiments are estimated based on measurements done on public network platforms such as the PlanetLab data set [52] which comprises of estimates for the end-to-end delays between devices, usually falling in a range of 3 to 352 ms.[10] We randomly assign these estimated delays to our experimental cluster configurations comprising IoT devices and edge servers. This is because we randomly assign the retrieval delays during the training process so that the trained model would adapt to the variances in the retrieval delays incurred. We run our experiments for 500 to 1500 episodes, while the rest of the parameters are defined in Table 1.

6.2 Evaluation Methodology

Following the standard practice in prior papers in this area [7,22], we run simulation experiments of our proposed algorithms by varying the following hyperparameters. The first hyperparameter is the *configuration* of a cluster which denotes

[9] https://gymnasium.farama.org/.

[10] In most near real-time applications, the upper latency limit is 300 ms. Further, in the PlanetLab dataset used, the latency actually falls within the chosen range of 3–352 ms. We assume that the streaming latency does not vary much with the nature of the data.

Table 1. Table of Parameters

Number of IoT devices	2000
Number of edge servers	100
$\mathcal{T}$	20
Exploration rate (ϵ)	1 Decayed to 0.1
Learning rate (α)	0.5 Decayed to 0.0001
Discount factor (γ)	0.99
replay buffer size	10000
mini batch size	64
update frequency U	50

the number of edge servers and IoT devices, along with the relative positioning of the devices due to their geographical distribution. Due to changes in the geographical distribution of IoT devices and edge servers, retrieval delay between each pair of IoT devices and edge servers may vary. Therefore, we choose random configurations in our experiments, where each configuration has different associated retrieval delays. The other hyperparameter is the number of episodes for the RL algorithm. We record variations in the observed retrieval delay and mean reward obtained with respect to the changes in the above hyperparameters. We evaluate our proposed algorithms against the following baseline algorithms.

- First, we evaluate with respect to the nearest neighbor search (NNS) which is a popular and efficient greedy algorithm used in similar problems in systems and networking [48]. NNS assigns each IoT device to an edge server based on a *greedy property*, where an IoT device is assigned to an edge server in a greedy manner such that the retrieval delay is minimized. If an edge server exceeds its threshold $\mathcal{T}$ with the above assignment, then the concerned IoT devices are assigned to the next available edge server. The greedy property of NNS fails to find the global optimal solution since it considers only the locally optimal assignments.
- Next, we consider the *Tabu Search* (TS) - meta-heuristic optimization technique for search-based problems which was used by Kherraf et al. for assigning the IoT workloads to the edge servers while reducing the completion time [15]. While evaluating TS, we initialize the TS instance with the solution obtained by applying the NNS baseline described above. TS stores metadata of *tabu moves*[11] and the recent list of solutions found in the search procedure. We consider a tabu move as an assignment that results in a maximum retrieval delay. While maintaining the solution list, we avoid considering the tabu moves up to certain iterations to find an improvement in the currently obtained solution. In each iteration, TS will try to improve the solution by swapping the

[11] A tabu move is a forbidden move that cannot be included in the recent solution list obtained.

tabu move with an assignment that results in a minimum retrieval delay The search procedure halts if the specified iterations have finished, or if the solution gets stuck at one point in the search space. The drawback of this baseline is that TS will get stuck at some point, i.e., further tabu moves cannot be swapped since the metadata contains partial knowledge of the assignments performed [32].

– Next, we evaluate our proposed algorithm against Google OR-Tools solver [35] - a popular optimization problem solver. Specifically, we run the SCIP solver[12] package - a mixed integer programming solver from Google OR-Tools. For that purpose, we transform the CAP$'''$ to a minimization problem as follows [1].

$$\begin{aligned} \min \quad & \mathcal{Z} \\ \text{subject to} \quad & \mathcal{Z} \geq p_{i,j} x_{i,j}, i = 1, \ldots, n, j = 1, \ldots, m, \\ & \sum_{i \in I} x_{i,j} = 1 \; j \in J \\ & \lfloor \frac{n}{m} \rfloor \leq \mathcal{T}. \end{aligned}$$

With the above experiment we get the optimal solution.

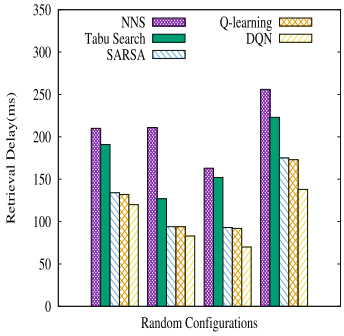
– As a last baseline, we compare with *Q-learning* and *SARSA*.

6.3 Experimental Results

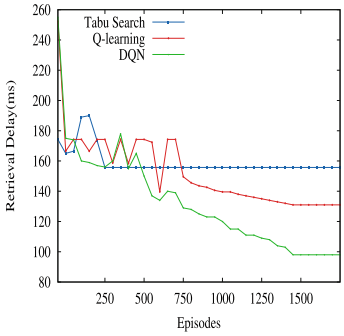
In Fig. 2(a), we illustrate the retrieval delay for different random configurations. We observe that the NNS approach results in a higher observed retrieval delay than TS, whereas tabular RL algorithms *Q-learning* and $SARSA$ perform betters than TS and NNS. In contrast, DQN is able to reduce the overall retrieval delay much more than all the baselines. Figure 2(b) shows that during the initial episodes, DQN results in a higher retrieval delay than TS. This is because TS employs a greedy NNS-based technique as its initial solution. In contrast, *Q-learning* first performs random assignment. Hence, the assigned IoT devices are not always nearest to the edge servers causing an increased retrieval delay. While the number of episodes increases, TS may become fixated on a single solution. On the other hand, *Q-learning* maintain a trade-off between exploration and exploitation - exploring, i.e., assigning an IoT device to an edge server that it has never seen before, and exploiting, i.e., assigning an IoT device to an edge server that it has seen before. It keeps learning various possible assignments from the environment and then exploits based on these discoveries. Note that one of the configurations from Fig. 2(a) was used for the rest of the experiments. The trends shown in Fig. 2(c) to Fig. 2(f) are consistent for all the configurations.

In Fig. 2(c), we evaluate $SARSA$, *Q-learning*, and DQN based on the mean reward obtained. During the exploration process of RL algorithms, mean reward

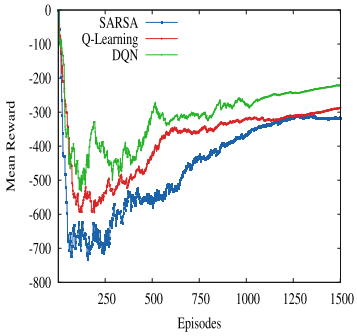
[12] https://www.scipopt.org/.



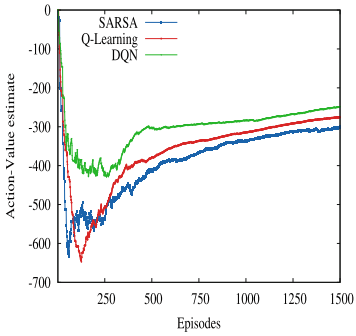
(a) Retrieval delay observed with different assignments of IoT devices to edge servers.



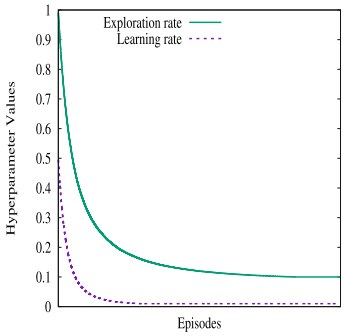
(b) Retrieval Delay observed with increase in the number of episodes where TS fixates on a one sub optimal solution whereas DQN provides much-reduced retrieval delay.



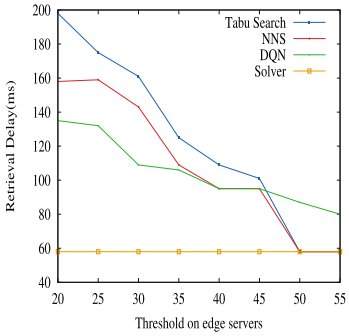
(c) Mean reward obtained during different episodes.



(d) Value Estimates in different episodes.



(e) Epsilon and Alpha Decayed values different epsiodes.



(f) Impact of increasing $\mathcal{T}$ at the edge servers on retrieval delay.

Fig. 2. Experimental Results with observed retrieval delay and mean reward.

includes the delayed rewards[13], i.e., a current action taken by the agent must take into account the immediate and future rewards. In Fig. 2(c), 2(d), we plot the mean reward obtained and estimated value of a state with respect to an increasing number of episodes. The mean reward obtained and value estimate will fluctuate in the initial episodes. Later, we observe that DQN gets a higher mean reward and value estimate than *Q-learning* and $SARSA$. Therefore, DQN shows better convergence to an optimal policy than tabular RL algorithms.

In Fig. 2(e), we demonstrate the decay of the exploration rate with the increase of the decay ratio. Following the standard practice [31], we set the exploration rate initially to 1, and the decay ratio degraded from 0.9 to 0.1. The initial higher values of exploration rate result in a higher degree of exploration and lesser exploitation. The learning rate also decayed from 0.5 to 0.01.

Table 2. Comparison of retrieval delays with varying sizes of IoT devices and edge servers

IoT devices	edge servers	OR-Tools Solver		Deep *Q-learning*	
		Iterations	Retrieval Delay (ms)	Episodes	Retrieval Delay (ms)
1000	50	1025	129	500	136.8
1500	100	2045	165.4	500	174.5
2000	100	3195	175.4	1000	162.5
2500	200	5195	198.40	1500	172.5

Following [33], we conduct similar experiments on Google or-tools with an increasing number of IoT devices and edge servers *solver*. We compared it with the DQN algorithm and recorded the observed retrieval delay in Table 2. We limited the solver iterations according to the episodes performed in DQN. Also, we conducted this experiment at approximately the same wall clock time for the solver and DQN. For smaller instances, the solver performs better than DQN. As the number of IoT devices and edge servers increases, DQN outperforms the solver solution. Hence, the RL approach scales better than the solver with an amortized cost of training.

Figure 2(f) illustrates the impact of increasing $\mathcal{T}$ at the edge servers on the observed retrieval delay for NNS, TS, solver, and DQN. Initially, DQN performs better than the baseline approaches. After increasing the $\mathcal{T}$ to a certain extent, NNS and TS match the solution obtained with the solver, whereas DQN gives a higher retrieval delay. This is because of unexplored assignments of the IoT devices to the edge servers in contrast to the greedy approach followed by NNS and TS. Here, we observe that the DQN is not converging with the solver solution whereas NNS and TS converged on the solver solution [6]. Though we increased $\mathcal{T}$ in our experimental setting, such variations are actually impractical in real scenarios since the edge server are typically resource constrained and are

[13] Delayed reward is a reward obtained after multiple episodes.

able to process data from multiple sources up to a certain threshold only. This is because edge servers are resource constrained and able to process a limited number of IoT devices only [6].

6.4 Assisting Federated Learning

As elaborated in Sect. 1, our proposed algorithm EAP is useful in the case of federated learning (FL), particularly in reducing the training time incurred at the edge servers (FL clients) in learning local models. Here, the edge servers and the IoT devices act as clients and data sources, respectively. The retrieval delay, i.e., the time in fetching the training data from IoT devices, has a direct impact on the local training time at the edge servers. Hence, in reducing the retrieval delay, our proposed algorithm is actually able to minimize the local training time [17]. For this experiment, we use cifar100 data set[14] and train an image classification model on different edge servers in a federate learning setting. Figure 3 illustrates that our approach results in considerably reducing the local training time compared to the baseline algorithms. Also, the figure illustrates that the retrieval delay consists of a large fraction of the local training time. Hence, reducing the retrieval delay is quite important in the case of FL. Note that we have considered that local models are trained only after the local training data has been retrieved. In federated learning systems, relatively large retrieval delays can result in excessively outdated local models in comparison with the global model. As a result, it would take the model aggregation at the client to take a much larger time in achieving convergence [17]. Thus, our approach is able to reduce the overall delay in federated learning.

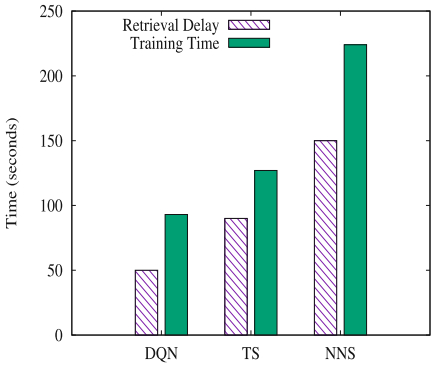


Fig. 3. Reduced local training time in federated learning.

[14] https://www.cs.toronto.edu/~kriz/cifar.html.

7 Conclusion and Future Work

In this work, we studied an intelligent assignment of the IoT devices to the edge servers to minimise the retrieval delay while ensuring that none of the edge servers is overloaded. We prove the problem is NP-hard even when the IoT devices and edge servers are stationary. To that end, we modeled the problem as an MDP framework and leveraged neural network-based Deep Q-learning to determine an optimal assignment of the IoT devices to the edge servers in a given cluster. In future work, we plan to consider IoT devices and edge servers to be non-stationary, which increases the problem's difficulty. Further, we would perform a much more in-depth analysis of the impact of EAP on different performance aspects of federated learning.

References

1. Design optimization. http://apmonitor.com/me575/index.php/Main/MiniMax
2. Whitepapers. https://docs.aws.amazon.com/whitepapers/latest/develop-deploy-dotnet-apps-on-aws/running-applications-in-containers.html
3. Abramson, M., Wechsler, H.: Tabu search exploration for on-policy reinforcement learning. In: 2003 Proceedings of the International Joint Conference on Neural Networks, vol. 4, pp. 2910–2915. IEEE (2003)
4. Ahsan, W., Yi, W., Liu, Y., Qin, Z., Nallanathan, A.: Reinforcement learning for user clustering in NOMA-enabled uplink IoT. In: 2020 IEEE International Conference on Communications Workshops (ICC Workshops), pp. 1–6. IEEE (2020)
5. Barthélemy, J., Verstaevel, N., Forehead, H., Perez, P.: Edge-computing video analytics for real-time traffic monitoring in a smart city. Sensors **19**(9) (2019). https://doi.org/10.3390/s19092048, https://www.mdpi.com/1424-8220/19/9/2048
6. Bonomi, F., Milito, R., Zhu, J., Addepalli, S.: Fog computing and its role in the internet of things. In: Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, MCC 2012, pp. 13–16. Association for Computing Machinery, New York (2012). https://doi.org/10.1145/2342509.2342513
7. Brockman, G., et al.: OpenAI gym. arXiv preprint arXiv:1606.01540 (2016)
8. Chien, W.C., Weng, H.Y., Lai, C.F.: Q-learning based collaborative cache allocation in mobile edge computing. Futur. Gener. Comput. Syst. **102**, 603–610 (2020)
9. Desikan, K.S., Kotagi, V.J., Murthy, C.S.R.: Topology control in fog computing enabled IoT networks for smart cities. Comput. Netw. **176**, 107270 (2020)
10. Fan, Q., Ansari, N.: Application aware workload allocation for edge computing-based IoT. IEEE Internet Things J. **5**(3), 2146–2153 (2018)
11. Hua, H., Li, Y., Wang, T., Dong, N., Li, W., Cao, J.: Edge computing with artificial intelligence: a machine learning perspective. ACM Comput. Surv. **55**(9), 1–35 (2023)
12. Huang, L., Bi, S., Zhang, Y.J.A.: Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks. IEEE Trans. Mob. Comput. **19**(11), 2581–2593 (2020). https://doi.org/10.1109/TMC.2019.2928811
13. Kairouz, P., et al.: Advances and open problems in federated learning. Found. Trends® Mach. Learn. **14**(1–2), 1–210 (2021)

14. Kherraf, N., Alameddine, H.A., Sharafeddine, S., Assi, C.M., Ghrayeb, A.: Optimized provisioning of edge computing resources with heterogeneous workload in IoT networks. IEEE Trans. Netw. Serv. Manage. **16**(2), 459–474 (2019)
15. Kherraf, N., Sharafeddine, S., Assi, C.M., Ghrayeb, A.: Latency and reliability-aware workload assignment in IoT networks with mobile edge clouds. IEEE Trans. Netw. Serv. Manage. **16**(4), 1435–1449 (2019). https://doi.org/10.1109/TNSM.2019.2946467
16. Khosravanian, R., Mansouri, V., Wood, D.A., Alipour, M.R.: A comparative study of several metaheuristic algorithms for optimizing complex 3-D well-path designs. J. Pet. Explor. Prod. Technol. **8**(4), 1487–1503 (2018). https://doi.org/10.1007/s13202-018-0447-2
17. Konečnỳ, J., McMahan, H.B., Yu, F.X., Richtárik, P., Suresh, A.T., Bacon, D.: Federated learning: strategies for improving communication efficiency. arXiv preprint arXiv:1610.05492 (2016)
18. Lai, F., You, J., Zhu, X., Madhyastha, H.V., Chowdhury, M.: Sol: fast distributed computation over slow networks. In: NSDI, vol. 20, pp. 273–288 (2020)
19. Li, T., Sahu, A.K., Talwalkar, A., Smith, V.: Federated learning: challenges, methods, and future directions. IEEE Signal Process. Mag. **37**(3), 50–60 (2020). https://doi.org/10.1109/MSP.2020.2975749
20. Lin, K.C.J., Wang, H.C., Lai, Y.C., Lin, Y.D.: Communication and computation offloading for multi-rat mobile edge computing. IEEE Wirel. Commun. **26**(6), 180–186 (2019). https://doi.org/10.1109/MWC.001.1800603
21. Liu, D., Kong, H., Luo, X., Liu, W., Subramaniam, R.: Bringing AI to edge: from deep learning's perspective. Neurocomputing (2021)
22. Liu, H., Cao, G.: Deep reinforcement learning-based server selection for mobile edge computing. IEEE Trans. Veh. Technol. **70**(12), 13351–13363 (2021). https://doi.org/10.1109/TVT.2021.3124127
23. Liu, S., Liu, L., Tang, J., Yu, B., Wang, Y., Shi, W.: Edge computing for autonomous driving: opportunities and challenges. Proc. IEEE **107**(8), 1697–1716 (2019). https://doi.org/10.1109/JPROC.2019.2915983
24. Liu, X., Yu, J., Wang, J., Gao, Y.: Resource allocation with edge computing in IoT networks via machine learning. IEEE Internet Things J. **7**(4), 3415–3426 (2020). https://doi.org/10.1109/JIOT.2020.2970110
25. Martello, S., Toth, P.: Knapsack Problems: Algorithms and Computer Implementations. Wiley, Hoboken (1990)
26. Martello, S., Toth, P.: The bottleneck generalized assignment problem. Eur. J. Oper. Res. **83**(3), 621–638 (1995)
27. Mazyavkina, N., Sviridov, S., Ivanov, S., Burnaev, E.: Reinforcement learning for combinatorial optimization: a survey. Comput. Oper. Res. **134**, 105400 (2021). https://doi.org/10.1016/j.cor.2021.105400, https://www.sciencedirect.com/science/article/pii/S0305054821001660
28. Mazzola, J., Neebe, A.: Bottleneck generalized assignment problems. Eng. Costs Prod. Econ. **14**(1), 61–65 (1988)
29. Min, M., Xiao, L., Chen, Y., Cheng, P., Wu, D., Zhuang, W.: Learning-based computation offloading for IoT devices with energy harvesting. IEEE Trans. Veh. Technol. **68**(2), 1930–1941 (2019). https://doi.org/10.1109/TVT.2018.2890685
30. Mnih, V., et al.: Playing Atari with deep reinforcement learning. CoRR abs/1312.5602 (2013). http://arxiv.org/abs/1312.5602
31. Morales, M., Isbell, C.: Grokking deep reinforcement learning. Manning (2020)

32. Nijimbere, D., Zhao, S., Gu, X., Esangbedo, M.O., Dominique, N.: Tabu search guided by reinforcement learning for the max-mean dispersion problem. J. Industr. Manage. Optim. **17**(6), 3223–3246 (2021)
33. Pathan, S., Shrivastava, V.: Reinforcement learning for assignment problem with time constraints (2021)
34. Perkin, T.M., Mini, S.: Assignment of IoT nodes to edge computing devices in internet of things. In: 2019 European Conference on Networks and Communications (EuCNC), pp. 528–532 (2019). https://doi.org/10.1109/EuCNC.2019.8802058
35. Perron, L., Furnon, V.: Or-tools. https://developers.google.com/optimization/
36. Rajashekar, K.: Reinforcement learning for minimizing communication delay in edge computing. In: 2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS), pp. 1270–1271. IEEE (2022)
37. Rajashekar, K., Paul, S., Karmakar, S., Sidhanta, S.: Topology aware cluster configuration for minimizing communication delay in edge computing. In: 2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS), pp. 1310–1311. IEEE (2022)
38. Schempp, P., Preuß, K., Tröger, M.: About the correlation between crude oil corrosiveness and results from corrosion monitoring in an oil refinery. Corrosion **72**(6), 843–855 (2016)
39. Sheng, M., Dai, Y., Liu, J., Cheng, N., Shen, X., Yang, Q.: Delay-aware computation offloading in NOMA MEC under differentiated uploading delay. IEEE Trans. Wireless Commun. **19**(4), 2813–2826 (2020). https://doi.org/10.1109/TWC.2020.2968426
40. Shoham, Y., Powers, R., Grenager, T.: Multi-agent reinforcement learning: a critical survey. Technical report (2003)
41. Silver, D.: Lectures on reinforcement learning (2015). https://www.davidsilver.uk/teaching/
42. Song, Y., Yau, S.S., Yu, R., Zhang, X., Xue, G.: An approach to QoS-based task distribution in edge computing networks for IoT applications, pp. 32–39. Institute of Electrical and Electronics Engineers Inc. (2017). https://doi.org/10.1109/IEEE.EDGE.2017.50
43. Spatharakis, D., et al.: A scalable edge computing architecture enabling smart offloading for location based services. Pervasive Mob. Comput. **67**, 101217 (2020). https://doi.org/10.1016/j.pmcj.2020.101217, https://www.sciencedirect.com/science/article/pii/S1574119220300778
44. Sudharsan, B., Breslin, J.G., Ali, M.I.: Edge2Train: a framework to train machine learning models (SVMs) on resource-constrained IoT edge devices. In: Proceedings of the 10th International Conference on the Internet of Things, pp. 1–8 (2020)
45. Sun, X., Ansari, N.: Latency aware workload offloading in the cloudlet network. IEEE Commun. Lett. **21**(7), 1481–1484 (2017). https://doi.org/10.1109/LCOMM.2017.2690678
46. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. A Bradford Book, Cambridge (2018)
47. Wei, Z., Jiang, H.: Optimal offloading in fog computing systems with non-orthogonal multiple access. IEEE Access **6**, 49767–49778 (2018). https://doi.org/10.1109/ACCESS.2018.2868894
48. Wikipedia: Nearest neighbor search—Wikipedia, the free encyclopedia (2022)
49. Wolpert, D.H., Macready, W.G.: No free lunch theorems for optimization. IEEE Trans. Evol. Comput. **1**(1), 67–82 (1997)

50. Xia, Q., Ye, W., Tao, Z., Wu, J., Li, Q.: A survey of federated learning for edge computing: Research problems and solutions. High-Confidence Comput. **1**(1), 100008 (2021). https://doi.org/10.1016/j.hcc.2021.100008, https://www.sciencedirect.com/science/article/pii/S266729522100009X
51. Yousefpour, A., Ishigaki, G., Jue, J.P.: Fog computing: towards minimizing delay in the internet of things. In: 2017 IEEE International Conference on Edge Computing (EDGE), pp. 17–24 (2017). https://doi.org/10.1109/IEEE.EDGE.2017.12
52. Zhu, R., Liu, B., Niu, D., Li, Z., Zhao, H.V.: Network latency estimation for personal devices: a matrix completion approach. IEEE/ACM Trans. Netw. **25**(2), 724–737 (2017). https://doi.org/10.1109/TNET.2016.2612695
53. Zinonos, Z., Vassiliou, V., Ioannou, C., Koutroullos, M.: Dynamic topology control for WSNs in critical environments. In: 2011 4th IFIP International Conference on New Technologies, Mobility and Security, pp. 1–5 (2011). https://doi.org/10.1109/NTMS.2011.5720652

Bringing the Edge Home: Edge Computing in the Era of Emerging WLANs

Sampath Edirisinghe[1(✉)] and Chathurika Ranaweera[2]

[1] Department of Computer Engineering, University of Sri Jayewardenepura, Nugegoda, Sri Lanka
essedirisinghe@sjp.ac.lk

[2] School of Information Technology, Deakin University, Geelong 3220, Australia
chathu.ranaweera@deakin.edu.au

Abstract. With the massive number of devices being added to Internet every day, the cloud infrastructure is struggling to serve them on timely manner with the required quality of service. In addition, the sheer volume of data that needs to be transported between the user devices and the cloud servers is massive and poses bottlenecks on the network infrastructure. In light of these challenges, Edge Computing is seen as a solution that can serve the user devices with less latency and ease the network congestion. Multi-Access Edge Computing (MEC) is one of the prominent solutions proposed recently with the primary focus of mobile cellular networks. However, a closer look at the user applications that require Edge Computing shows that having the Edge Computing facility at houses, offices or factories enable more effective operation and utilization of the capabilities of Edge Computing. Furthermore, the emerging WLANs such as Wi-Fi 6/6E/7, Li-Fi, and Optical Wireless Communication are addressing the indoor network capacity barriers that prevented Edge Computing being deployed closer to the user. Therefore, in this article, we introduce an Edge Computing architecture for wireless local area networks (WLAN) which resides in user premises. The viability and the performance of the architecture is evaluated with the use of Wi-Fi 6 WLAN network.

Keywords: Edge Computing · Wi-Fi · WLAN · MEC

1 Introduction

Edge Computing (EC) is being developed with a predominant focus on the mobile cellular networks. These developments were called mobile edge computing, however, it was later standardized by ETSI as multi-access edge computing (MEC) [39]. MEC is expected to provide low latency, highly reliable, and low energy computational facilities for users who are connected to 5G/6G and beyond

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 86–102, 2024.
https://doi.org/10.1007/978-3-031-63992-0_6

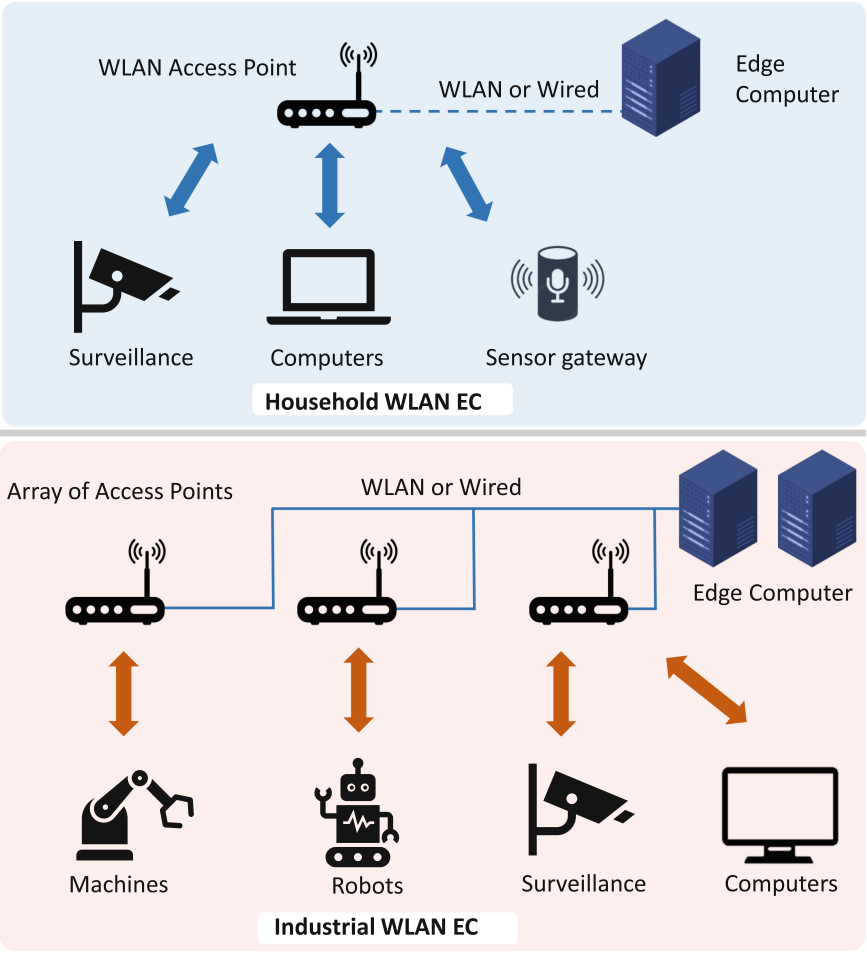


Fig. 1. Deployment of WLAN Edge Computing in household and industrial scenarios.

networks [34,35]. Furthermore, computing at the network edge reduces the congestion in the backhaul networks and brings the cost for network operators down [17].

There are a number of studies that explore the use of MEC in different directions such as smart cities, virtual/augmented reality (VR/AR), Edge Intelligence, unmanned aerial vehicles, vehicular networks, and Internet of Things (IoT) [10,18,26,27,36,43–45]. Nevertheless, in order to reap the benefits of MEC, a user needs to be connected to a cellular base-station, which can be a power consuming and costly task for certain user devices such as IoT devices [11]. Moreover, there might be devices that do not have the capability to connect to a cellular network or cannot be connected due to the network failures [42].

Therefore, many devices that we find in houses, offices and factories that can benefit from Edge Computing facilities do not get the opportunity to use it. These devices are usually connected to a wireless local area network (WLAN) such as Wi-Fi or Li-Fi.

A few Edge Computing architectures were proposed for indoor spaces using WLANs. For instance, a 3-tier architecture called Home Edge Computing (HEC) was introduced in [2] to decrease the latency of accessing the cloud resources in applications such as AR/VR and Tactile Internet. This architecture is proposed as an extension to MEC with an additional layer called the local data center or home server. However, the proposed 3-tier architecture have to maintain a constant flow of information with the cloud servers and intermediate nodes which hinders the low latency operation of the HEC. Moreover, the home server is transparent to the users and is completely managed by the Internet Service Provide (ISP). Hence, the users do not get to add custom features, apps, and capabilities to the home server.

Further, authors in [37] also introduced access edge cloud, where a server in attached to the WLAN access points. However, the implementation gives more emphasis on the metro edge cloud and the results are transparent of the particular WLAN technology. Further, the authors introduce a new TDMA-based (time division multiple access) polling for the effective resource management in the metro edge cloud. In [9], authors introduced EdgeAP, where a management server orchestrates different applications in WLAN access points that are connected to it. These applications are decided based on the user demand and the users cannot have a fixed set of applications in a given access point. Therefore, the EdgeAP architecture is similar to HEC proposed in [2].

Cloudlet Computing is also a similar concept where the cloud computing capabilities are brought closer to the user [1]. However, similar to the previous approaches, Cloudlet Computing is also managed by service providers. Hence, the exact location of deployment, services that are running in the Cloudlet, and hardware/software specifications are entirely determined by the service providers [16]. Therefore, while Cloudlet Computing can enhance the performance of data processing at the network edge, it does not yield the full autonomy that WEC can provide the users.

In terms of applications of WLAN Edge Computing, authors in [44] highlight that artificial intelligence (AI) in the network edge would play a key role in future IoT and mobile computing landscape. While there are a number of challenges in implementing AI in the edge, there are various techniques such as federate learning, gradient compression, model compression and partitioning that help enable the edge AI implementations. Moreover, providing Industrial IoT as a Service (IIoTaaS) is an upcoming trend in future manufacturing industries. IIoTaaS provides storage and computational resources closer to industrial plants and minimize the latency and risk of data breaches [4].

A common challenge with WLAN-based EC is that the WLANs were not seen as very reliable networks that can provide guaranteed connectivity and latency. Hence, WLAN-based EC was not considered as a viable solution for providing

computational facilities at the network edge. However, the latest versions of the IEEE 802.11 standard series offers compelling solutions to these challenges [13]. The IEEE 802.11ax standard which is better known as Wi-Fi 6 features radical changes such as orthogonal frequency division multiple access (OFDMA) that enables Wi-Fi to provide guaranteed access to all or a subset of its users [22]. Moreover, Wi-Fi 6 comes with increased bandwidth support and it is set to be expanded to 6 GHz range as well (Wi-Fi 6E) [12]. In addition to that Wi-Fi 7 (IEEE 802.11be) is also underway with more features like multi-link operation and time-sensitive networking to further enhance the capacity and reliability of Wi-Fi links [8]. With these enhancements, the most recent and upcoming Wi-Fi standards would provide guaranteed performance in terms of latency and connectivity. The increased bandwidth can be used to easily facilitate the increasing number of IoT devices or any other sensors connected to the network.

Visible light communication, Li-Fi, optical wireless communication are also widely explored for WLAN deployments due to its high capacity and privacy [31–33]. The IEEE 802.11 standards expand into the optical region of the electromagnetic spectrum with the latest IEEE 802.11bb light communication standard [23]. Hence, now it is possible to build visible light communication links that comply with the IEEE 802.11 standards, which encourages a heterogeneous WLAN ecosystem with much higher capacity and link reliability.

Therefore, is it evident that the development of emerging WLANs paves the way to support the WLAN-based EC with their ability to provide guaranteed access [13,14], higher throughput and bandwidth. Therefore, exploring the combination of EC and emerging WLANs is a timely contribution that can improve how the devices that are connected to WLANs can utilize EC facilities.

In that regard, this article presents a novel architecture for deploying WLAN-based EC (WEC) systems in households and industrial settings. The WEC system implementation in both scenarios are shown in Fig. 1, which will be explained in detail in Sect. 2. With this architecture, the following contributions are made in the article.

- We introduce a novel edge computing architecture where the edge computing device is connected to the household/industrial WLAN. The architecture is a flexible and scalable solution as the user can determines the specifications of the edge computer depending their requirements.
- We introduce an app-based ecosystem where the user installs the applications (apps) they require and these apps will execute the functions related to different services.
- We evaluate the performance of the proposed architecture using Wi-Fi 5 and Wi-Fi 6 networks to demonstrate that emerging WLANs such as Wi-Fi 6 networks enables smooth operation of edge computing at the WLAN edge.

The rest of the article is organized as follows. The proposed architecture is presented in Sect. 2. The Sect. 3 introduces key applications for WLAN EC and their requirements. The performance evaluations and analyses are presented in Sect. 4. Finally, Sect. 5 concludes the paper.

2 Proposed Architecture of the WLAN-Based Edge Computing

This section will demonstrate the need for WLAN-based Edge Computing using a simple networking scenario and present the proposed architecture for WLAN-based Edge Computing. Furthermore, the last subsection will go into the details of the construction of the Edge Computer.

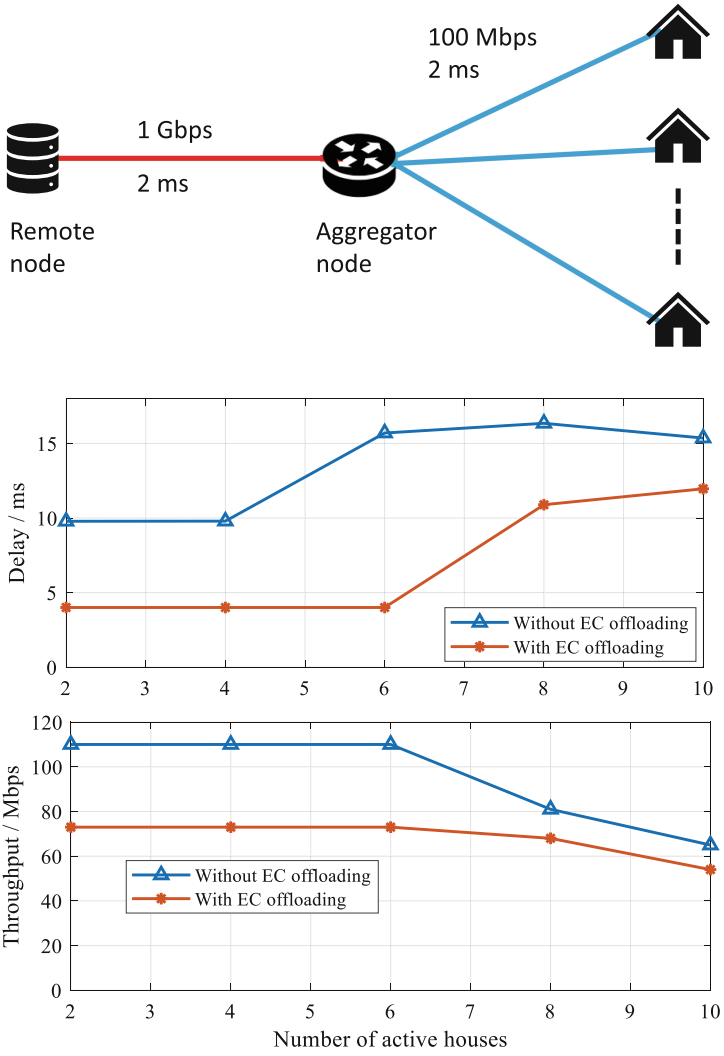


Fig. 2. Impact of offloading traffic to on-premises Edge Computers.

2.1 The Need of Edge Computing in WLAN

Consider the scenario given in Fig. 2 where a network which has 10 user premises with 110 Mbps of peak demand each. All houses are connected to a router with 1 Gbps uplink capacity and the user premises are connected to the router using 100 Mbps links. Two experiments were done with and without WLAN-based EC facilities at user premises. In the first experiment, it is assumed that 100% of the peak demand is send to the remote node. As shown in the delay and throughput plots shown in Fig. 2, the delay and throughput of users suffer when the total network load is close to the peak (1 Gbps). On the other hand, if 30% of the peak demand can be served on-premises, the traffic that needs to be transported in the core network reduces and the results are reflected in Fig. 2. The overall network metrics experience a significant improvement in the latter case.

2.2 WLAN-Based Edge Computing Architecture

The proposed novel WLAN-based Edge Computing architecture is shown in Fig. 1. The proposed architecture can be equally applied to household and industrial scenarios to carry out different tasks. The architecture features a WLAN access point and it is connected to an edge computer. Edge computers can be built using general-purpose computers, single-board computers such as Raspberry Pi, or more advanced miniature computers such as NVIDIA Jetson Nano [5]. The user devices are connected to the access point via emerging WLAN technologies while the edge computer can be connected via WLAN or wired (Ethernet) connections. As shown in the industrial scenario in Fig. 1, there could be multiple access points to serve the coverage area. However, they are connected a single edge computer which is more powerful than a household edge computer.

2.3 Architecture of WLAN-Based Edge Computer

The architecture of the WLAN-based Edge Computer is shown in Fig. 3. It consists of all the components of a general-purpose computer such as hardware, operating system, drivers, libraries and software applications. In addition to that, the WEC is connected a WLAN access point via Ethernet or WLAN.

The hardware components of the WEC are mainly the central processing unit (CPU), graphics processing unit (GPU), memory, storage, network interfaces and other peripherals to communicate with the external sources. Depending on the user requirements certain components might have high-end specifications. For instance, an industrial WEC that controls robots might need high performance CPU and GPU whereas a household WEC might need more storage capacity.

The operating system of the WEC will manage the hardware and software that run on top of it. The libraries installed in the operating system play a key role in facilitating the applications. Depending on the requirements of the applications, users can install libraries such a machine learning and GPU. In Fig. 3, dashed entities represent industrial WEC components. For instance, industrial

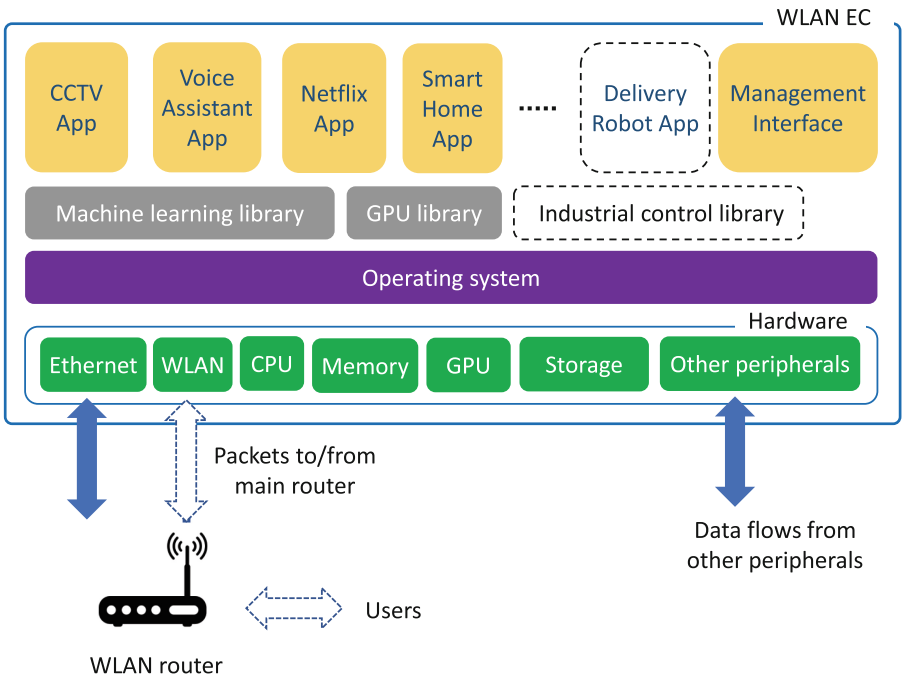


Fig. 3. The proposed architecture of the WLAN edge computer.

robot control would require libraries such as simultaneous localization and mapping (SLAM) [21].

The applications are installed by the user based on their requirements. For instance, if the user needs their Netflix account to cache latest movies in the WEC, they can install the Netflix application and configure it. Similarly, if the user wants to run a certain machine learning surveillance on their CCTV (closed circuit television) feed and send alert, they can install a CCTV application in the WEC and forward the CCTV feed to the WEC. Installation of applications is done by the management interface of the WEC. This management interface can be accessed by a computer by typing in the WEC's IP address. Using this interface, the user can install/uninstall, configure applications and install libraries. The app-based ecosystem is one of the key differentiators of the proposed architecture from Cloudlet Computing where the edge computer is used to execute a certain set of calculations [1]. The proposed architecture uses fully-fledge apps in the edge computer which runs standalone computational tasks. It should be noted that these apps are different from the apps installed in mobile phones or computers. The apps for the WEC are specifically designed for the WECs to carry out a specific set of tasks. In addition to the standard security measures provided by the computing system, WEC apps can also run the machine

learning based anomaly detection algorithms to identify any vulnerabilities and notify the users.

As discussed earlier, the connection between the access point and the WEC can be Ethernet or WLAN. For instance, the resource allocation mechanisms of Wi-Fi 6/6E/7 greatly suits this task as they can provide dedicated network slices for certain connections. The user devices will be connected to the access point and certain groups of users will have dedicated network slices. In addition to the data flows from the main router, the WEC can interface with other peripherals using its own interfaces such as audio ports, Bluetooth interfaces and USB. This way the WEC can provide additional services via its own interfaces.

3 Application Scenarios of WEC

Emerging user applications can be supported through both cloud and edge computing [15]. These computing paradigms offer distinct advantages that cater to various aspects of application requirements. However, bandwidth intensive applications with a ultra-low delay requirement can greatly benefit from edge computing. In this section, a few application scenarios that can benefit from WEC is discussed with their requirements such as data rate, latency, computational power, and memory.

3.1 Real-Time Security Monitoring Applications

The CCTV systems are widely used nowadays for surveillance purposes. However, security officers are often employed to monitor them on a 24×7 basis. With the increasing number of cameras, it is becoming a costly and ineffective endeavor. Hence, there has been attempts to automate this process using machine learning and cloud computing [40]. However, as we discussed earlier, cloud computing poses its own limitations in data rate and latency. Especially, if the CCTV system is large and consists of high-definition cameras. Therefore, deploying a WEC for this task helps reducing the cost and operational latency to a significant degree. In addition, the CCTV footage will stay local preserving the privacy of the individuals. If the WEC is attached to the other systems of the house or industrial building, it is possible to execute security measure without delays.

3.2 Industry 5.0

Industrial control is another area where edge computing is highly applied. With Industry 4.0, most of the factories and other industrial spaces were equipped with smart machines and sensors to efficiently mass produce different products. However, Industry 5.0 envisions to bring mass personalization and human interaction to the factories [25]. Industry 5.0 requires a version of edge computing that can be modified easily. Therefore, WEC suits well for the upcoming Industry 5.0 spaces for a number of reasons [28]. WEC with emerging WLANs can

provide guaranteed access to the machines and other devices unlike older versions of WLANs (Wi-Fi 5 or older). WEC allows the industries to maintain their proprietary data on their own system without compromising the security and privacy. Since the libraries and apps are locally maintained, updates can be easily done providing a scalable and flexible solutions for industry automation.

3.3 Personalized In-Home Care

With the ageing population in many countries, it is vital to provide in-home care for elderly people and disabled individuals. While there are different sensors that capture visual, auditory and medical sensors that can collect data from individuals, transportation of these data and processing them in cloud servers is a challenging task in terms of data volume, latency and privacy. Hence, there have been attempts to address these concerns using MEC and network slicing [29]. With the proposed WEC architecture, it is possible to address these concerns in an effective way due to the low latency and high privacy. Moreover, since the WEC supports installation and configuration of new apps, the users can install and customize apps to monitor individuals with various medical conditions.

3.4 Work-from-Home Caching

With the advent of work-from-home (WFH) culture, it is necessary to access the workplace data in a fast and reliable manner. At the moment, most companies use Google Drive, Dropbox or Outlook OneDrive for workplace file storage and sharing. Only a small fraction of the companies can afford to deploy there own file servers. Most of the general file sharing services either have long delay or requires to store the files locally. Hence, moving between devices, working with tablets is cumbersome. However, the proposed architecture can serve as a local server and an app can manage the copies and update the versions [3]. Furthermore, specific access restrictions and other custom access policies can be easily implemented using the apps.

3.5 Voice/Virtual Assistants

Voice assistants such as Google Home, and Amazon Alexa are quite popular these days and are helpful in number of avenues of our daily lives. At the moment, these devices come as separate hardware with in-built processing and communication capabilities. However, with the proposed WEC, it is possible to build apps that can serve as voice assistants. Authors in [41] proposes to use EC to improve the performance of existing voice assistants by reducing the cloud access delay. If the WEC is equipped with audio input/output, the whole functionality can be achieved in the WEC and can be integrated to household audio systems. Furthermore, it will give the voice assistants seamless access to the physical aspects such as smart home sensors and control devices for monitoring and controlling.

Table 1. Simulation Parameters

Application	Direction	Packet Size	Data rate
Netflix	Downlink	1500 bytes	8 Mbps
2 × WFH cache	Downlink	1200 bytes	2 × 1 Mbps
10 × General Internet	Downlink	1000 bytes	10 × 0.6 Mbps
5 ×CCTV	Uplink	1400 bytes	5 ×1 Mbps
2 × WFH cache	Uplink	1200 bytes	2 × 1 Mbps
Sensor gateway	Uplink	700 bytes	0.2 Mbps
10 × General Internet	Uplink	1000 bytes	10 × 0.6 Mbps

3.6 Internet of Everything (IoE)

With the IoE concept, numerous sensors and other devices are embedding into households and industrial spaces. These sensors and devices acquire sensitive information from the surroundings. With the typical cloud computing systems, this information has to be sent to the cloud which will pose a security and privacy threat [7]. However, most of functionalities in the cloud can be installed in the edge computer and execute at the edge [38]. This will help preserving the privacy and decreasing the latency of the responses. Therefore, IoE devices can be connected to the WEC and the management of the IoE devices can be done through a suitable app installed in the WEC.

3.7 Caching of Content Delivery Services

Video services such as Netflix and YouTube use caching at different levels of the network for seamless delivery. As describe earlier, a personal level content caching and storage can help reduce the congestion in the core/transport networks and scalability of content delivery systems [20,24]. Furthermore, WEC can intelligently coordinate with the core network to do the caching at low congestion periods to ease the burden the core network.

3.8 In-Home AR/VR Engines

AR/VR technologies are penetrating our daily lives in terms of gaming, entertainment and education. The cloud-based operation of AR/VR is almost impossible due to data rates involved and the stringent latency requirements. A number of studies are carried out to evaluate to the feasibility of using MEC for AR/VR rendering [6,30]. Therefore, bringing these functionalities home with WEC would further enhance the performance of AR/VR applications.

4 Performance Evaluation

The viability of the proposed architecture is evaluated by implementing the proposed WEC in the ns-3 simulation environment [19]. The ns-3 simulation

environment supports both legacy Wi-Fi 5 and Wi-Fi 6 networks. Two simulation studies are conducted to observe the viability of WEC in Wi-Fi 5 and Wi-Fi 6 networks. These simulations are carried out with the same network and traffic configurations. The latency of the user applications are observed to evaluate the performance of the proposed WECs.

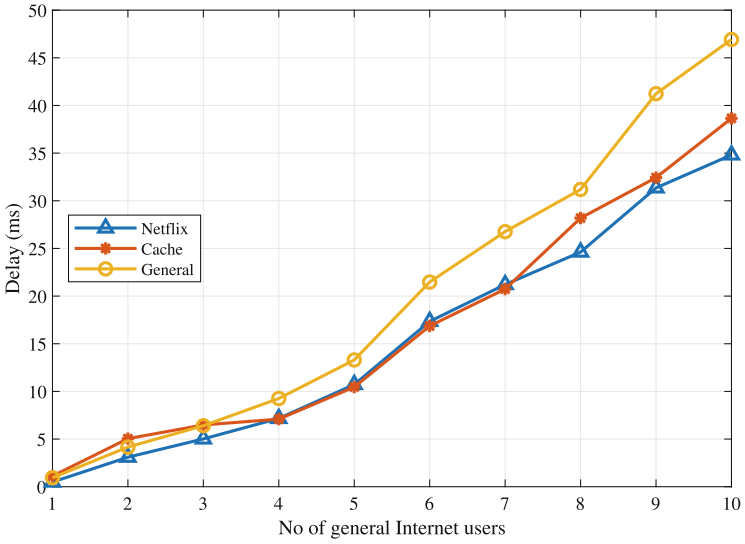


Fig. 4. Latency of downlink user applications using Wi-Fi 5 networks in a WLAN-based EC.

The network considered for the simulations consists of the traffic shown in Table 1. As shown in the table, both uplink and downlink traffic are considered for the simulations. Different types of applications are also simulated. The Netflix, WFH cache, CCTV, and sensor applications implemented where they are using the apps in the WEC, which might use or not use the Internet connection. The general Internet users do not connect to the WEC, however, they use the bandwidth of the Wi-Fi access point to connect to the Internet. The simulation studies increase the load on the Wi-Fi access point by increasing the number of general Internet users from 1 to 10 and observe the performance of all the traffic traversing the WEC and WLAN.

Figures 4 and 5 show the performance of the WEC with Wi-Fi 5 network. As it can be seen from these figures, the latency of the network depends on the load of the network. For the downlink traffic, the latency has increased almost $\times 10$ for all the traffic categories when the general Internet users were increased from 1 to 10. Similarly, uplink also demonstrates a consistent increase in the latency with increasing number of users. Hence, it is difficult to use Wi-Fi 5 networks for WECs as the network performance is highly dependant on the network load. The main reason behind this behaviors is that Wi-Fi 5 networks do not have

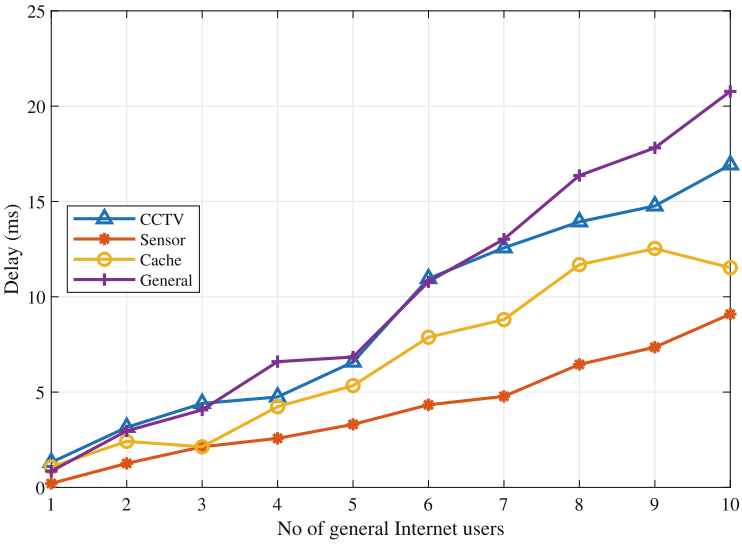


Fig. 5. Latency of uplink user applications using Wi-Fi 5 networks in a WLAN-based EC.

the ability to allocate bandwidth based on the latency requirements. The Wi-Fi 5 networks are based on a contention-based medium access control (MAC) protocol. While it results in low latencies under low load, the latency will quickly rise up with increasing number of users. Therefore, though it is possible to build WECs using Wi-Fi 5 networks, it will not be able to consistently offers the low latency and highly reliable services of a WEC.

On the other hand, the Wi-Fi 6 network can be configured to have a constant latency for required applications. Figures 6 and 7 demonstrate the delay performance of downlink and uplink, respectively. In this configuration, the general purpose traffic is not set to have a constant latency. The rest of the user applications are configured to have a constant latency by allocated bandwidth portions for them. As depicted in Fig. 6, the Netflix and WFH cache traffic stays at the same latency level while the general Internet user latency increased with increasing number of users. In this case, the general Internet user latency can be higher than that of Wi-Fi 5 as the dedicated bandwidth were allocated for Netflix and WFH cache. A similar behavior is observed in the uplink scenario shown in Fig. 7. Only the general Internet users will have an increase in the latency while the rest of the user applications experience a constant delay. Therefore, it is clear that with Wi-Fi 6 networks it is viable to build WECs that can provide guaranteed latency. It is worthwhile noting that some application such as CCTV experience less latency under the Wi-Fi 5 networks due to the contention-based MAC protocol in legacy Wi-Fi networks. However, when it comes to Wi-Fi 6 networks, the scheduled access may result in higher latency, however, it will be consistent with the increasing network load.

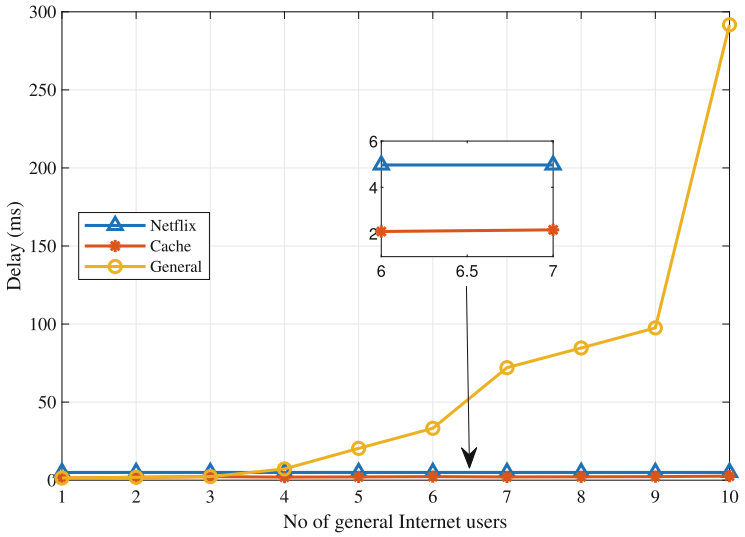


Fig. 6. Latency of downlink user applications using Wi-Fi 6 networks in a WLAN-based EC.

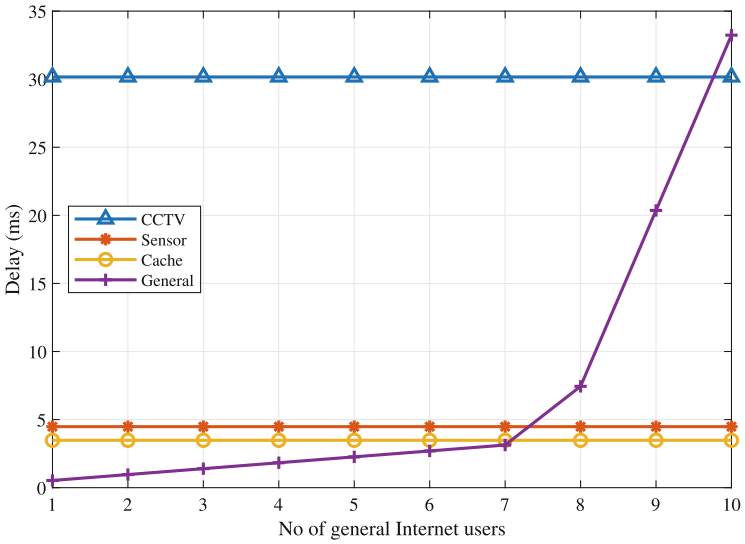


Fig. 7. Latency of uplink user applications using Wi-Fi 6 networks in a WLAN-based EC.

As it can be seen from the above analysis, the Wi-Fi 6 networks appear to be a viable solution for building WECs. The Wi-Fi 5 networks have an unpredictable latency, therefore, are not suitable for WECs as WECs are expected to maintain a constant latency. However, with the improve network resource allocation mechanisms in Wi-Fi 6, it is possible to ensure guaranteed latency and reliability for the user applications. Hence, using Wi-Fi 6, it is possible to build WECs that can bring the advantages offered by MECs into the home environment, bringing the edge computing another step closer to the end users.

5 Conclusion

Bringing Edge Computing facilities to WLANs can provide many benefits for a number of user applications such as IoE, smart home, caching services in terms to latency, privacy and data rate. However, so far, the unreliable WLAN technologies did not provide a proper platform for WLAN-based EC systems. With the emergence of Wi-Fi 6/6E/7, Li-Fi, and optical wireless communications, WLANs are becoming more reliable as they are capable of providing guaranteed latency for user applications. Therefore, it is necessary to evaluate the possible architectures for EC using these emerging WLANs. This article introduces a novel architecture which consists of a WLAN Edge Computing based on Wi-Fi 6. The WEC operates with a collection of apps that the user can install/customize to suit their applications. We analysed the viability of the proposed architecture by using Wi-Fi 5 and Wi-Fi 6 networks. The results confirm the fact though Wi-Fi 5 networks are not reliable enough to build WECs, the Wi-Fi 6 networks appears to be the best platform for WECs at the moment with constant latency and reliability, help bringing the edge computing one more step closer to the users.

References

1. Babar, M., Khan, M.S., Ali, F., Imran, M., Shoaib, M.: Cloudlet computing: recent advances, taxonomy, and challenges. IEEE Access **9**, 29609–29622 (2021)
2. Babou, C.S.M., Fall, D., Kashihara, S., Niang, I., Kadobayashi, Y.: Home edge computing (HEC): design of a new edge computing technology for achieving ultra-low latency. In: Liu, S., Tekinerdogan, B., Aoyama, M., Zhang, L.-J. (eds.) EDGE 2018. LNCS, vol. 10973, pp. 3–17. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-94340-4_1
3. Bilal, K., Shuja, J., Erbad, A., Alasmary, W., Alanazi, E., Alourani, A.: Addressing challenges of distance learning in the pandemic with edge intelligence enabled multicast and caching solution. Sensors **22**(3), 1092 (2022)
4. Borsatti, D., Davoli, G., Cerroni, W., Raffaelli, C.: Enabling industrial IoT as a service with multi-access edge computing. IEEE Commun. Mag. **59**(8), 21–27 (2021)
5. Cass, S.: Nvidia makes it easy to embed AI: the Jetson nano packs a lot of machine-learning power into DIY projects-[hands on]. IEEE Spectr. **57**(7), 14–16 (2020)

6. Chakareski, J., Gupta, S.: Multi-connectivity and edge computing for ultra-low-latency lifelike virtual reality. In: 2020 IEEE International Conference on Multimedia and Expo (ICME), pp. 1–6. IEEE (2020)
7. Chinchawade, A.J., Lamba, O.S.: Security and privacy challenges in internet of everything (IoE) with security requirements. J. Eng. Technol.-Suresh Gyan Vihar Univ. **7**(2) (2021)
8. Deng, C., et al.: IEEE 802.11 be Wi-Fi 7: new challenges and opportunities. IEEE Commun. Surv. Tutor. **22**(4), 2136–2166 (2020)
9. Desiniotis, C., Majors, J., Miremadi, C.: EdgeAP: enabling edge computing on wireless access points (2020)
10. Dias, I., Ruan, L., Ranaweera, C., Wong, E.: From 5G to beyond: passive optical network and multi-access edge computing integration for latency-sensitive applications. Opt. Fiber Technol. **75**, 103191 (2023). https://doi.org/10.1016/j.yofte.2022.103191
11. Ding, J., Nemati, M., Ranaweera, C., Choi, J.: IoT connectivity technologies and applications: a survey. IEEE Access **8**, 67646–67673 (2020). https://doi.org/10.1109/ACCESS.2020.2985932
12. Drahoš, P., Kučera, E., Haffner, O., Pribiš, R., Beňo, L.: Trends in industrial networks including APL, TSN, WiFi-6E and 5G technologies. In: 2022 Cybernetics & Informatics (K&I), pp. 1–7. IEEE (2022)
13. Edirisinghe, S., Galagedarage, O., Dias, I., Ranaweera, C.: Recent development of emerging indoor wireless networks towards 6G. Network **3**(2), 269–297 (2023)
14. Edirisinghe, S., Ranaweera, C., Lim, C., Nirmalathas, A., Wong, E.: Universal optical network architecture for future wireless LANs [invited]. J. Opt. Commun. Netw. **13**(9), D93–D102 (2021). https://doi.org/10.1364/JOCN.426215
15. Fereira, R., Ranaweera, C., Schneider, J.G., Lee, K.: Optimal node selection in communication and computation converged IoT network. In: 2022 IEEE International Conference on Parallel & Distributed Processing with Applications (ISPA), pp. 539–547 (2022). https://doi.org/10.1109/ISPA-BDCloud-SocialCom-SustainCom57177.2022.00075
16. Fernández-Cerero, D., Fernández-Rodríguez, J.Y., Álvarez-García, J.A., Soria-Morillo, L.M., Fernández-Montes, A.: Single-board-computer clusters for cloudlet computing in Internet of Things. Sensors **19**(13), 3026 (2019)
17. Filali, A., Abouaomar, A., Cherkaoui, S., Kobbane, A., Guizani, M.: Multi-access edge computing: a survey. IEEE Access **8**, 197017–197046 (2020)
18. Hasarinda, R., Tharuminda, T., Palitharathna, K., Edirisinghe, S.: Traffic collision avoidance with vehicular edge computing. In: 2023 3rd International Conference on Advanced Research in Computing (ICARC), pp. 316–321. IEEE (2023)
19. Henderson, T.R., Lacage, M., Riley, G.F., Dowell, C., Kopena, J.: Network simulations with the NS-3 simulator. SIGCOMM Demonstration **14**(14), 527 (2008)
20. Jayasundara, C., Zukerman, M., Nirmalathas, T.A., Wong, E., Ranaweera, C.: Improving scalability of VoD systems by optimal exploitation of storage and multicast. IEEE Trans. Circuits Syst. Video Technol. **24**(3), 489–503 (2014). https://doi.org/10.1109/TCSVT.2013.2290397
21. Khairuddin, A.R., Talib, M.S., Haron, H.: Review on simultaneous localization and mapping (SLAM). In: 2015 IEEE International Conference on Control System, Computing and Engineering (ICCSCE), pp. 85–90. IEEE (2015)
22. Khorov, E., Kiryanov, A., Lyakhov, A., Bianchi, G.: A tutorial on IEEE 802.11 ax high efficiency WLANs. IEEE Commun. Surv. Tutor. **21**(1), 197–216 (2018)
23. Khorov, E., Levitsky, I.: Current status and challenges of Li-Fi: IEEE 802.11 bb. IEEE Commun. Standards Mag. **6**(2), 35–41 (2022)

24. Lall, S., Sivakumar, R.: Will they or won't they?: toward effective prediction of watch behavior for time-shifted edge-caching of Netflix series videos. In: 2021 IEEE/ACM Symposium on Edge Computing (SEC), pp. 257–270. IEEE (2021)
25. Leng, J., et al.: Industry 5.0: prospect and retrospect. J. Manuf. Syst. **65**, 279–295 (2022)
26. Li, H., Shou, G., Hu, Y., Guo, Z.: Mobile edge computing: progress and challenges. In: 2016 4th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud), pp. 83–84 (2016). https://doi.org/10.1109/MobileCloud.2016.16
27. Liu, Y., Peng, M., Shou, G., Chen, Y., Chen, S.: Toward edge intelligence: multiaccess edge computing for 5G and internet of things. IEEE Internet Things J. **7**(8), 6722–6747 (2020)
28. Maddikunta, P.K.R., et al.: Industry 5.0: a survey on enabling technologies and potential applications. J. Ind. Inf. Integr. **26**, 100257 (2022)
29. Martiradonna, S., Cisotto, G., Boggia, G., Piro, G., Vangelista, L., Tomasin, S.: Cascaded WLAN-FWA networking and computing architecture for pervasive in-home healthcare. IEEE Wirel. Commun. **28**(3), 92–99 (2021)
30. Mehrabi, A., Siekkinen, M., Kämäräinen, T., yl Jski, A.: Multi-tier CloudVR: leveraging edge computing in remote rendered virtual reality. ACM Trans. Multimed. Comput. Commun. Appl. (TOMM) **17**(2), 1–24 (2021)
31. Nirmalathas, A., et al.: High-speed optical wireless communications for local area networks. In: 2018 23rd Opto-Electronics and Communications Conference (OECC), pp. 1–2. IEEE (2018)
32. Nirmalathas, A., et al.: Indoor optical wireless access networks-recent progress [invited]. J. Opt. Commun. Netw. **13**(2), A178–A186 (2021). https://doi.org/10.1364/JOCN.403485
33. Nirmalathas, T.A., et al.: Gigabit/s optical wireless access and indoor networks. In: 2020 Optical Fiber Communications Conference and Exhibition (OFC), pp. 1–3 (2020)
34. Ranaweera, C., et al.: Design and deployment of optical x-haul for 5G, 6G, and beyond: progress and challenges [invited]. J. Opt. Commun. Netw. **15**(9), D56–D66 (2023). https://doi.org/10.1364/JOCN.492334
35. Ranaweera, C., et al.: Optical x-haul options for 5G fixed wireless access: which one to choose? In: IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 1–2 (2018). https://doi.org/10.1109/INFCOMW.2018.8406906
36. Ranaweera, C., et al.: Rethinking of optical transport network design for 5G/6G mobile communication. IEEE Future Netw. Tech Focus (2021)
37. Rimal, B.P., Maier, M., Satyanarayanan, M.: Experimental testbed for edge computing in fiber-wireless broadband access networks. IEEE Commun. Mag. **56**(8), 160–167 (2018)
38. Sachdev, R.: Towards security and privacy for edge AI in IoT/IoE based digital marketing environments. In: 2020 fifth International Conference on Fog and Mobile Edge Computing (FMEC), pp. 341–346. IEEE (2020)
39. Shahzadi, S., Iqbal, M., Dagiuklas, T., Qayyum, Z.U.: Multi-access edge computing: open issues, challenges and future perspectives. J. Cloud Comput. **6**(1), 1–13 (2017)
40. Shirsat, S., Naik, A., Tamse, D., Yadav, J., Shetgaonkar, P., Aswale, S.: Proposed system for criminal detection and recognition on CCTV data using cloud and machine learning. In: 2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN), pp. 1–6. IEEE (2019)

41. Xu, L., Iyengar, A., Shi, W.: Cha: a caching framework for home-based voice assistant systems. In: 2020 IEEE/ACM Symposium on Edge Computing (SEC), pp. 293–306. IEEE (2020)
42. Yu, Y., et al.: Optimization and deployment of survivable fiber-wireless (FiWi) access networks with integrated small cell and WiFi. In: 2015 IEEE International Conference on Ubiquitous Wireless Broadband (ICUWB), pp. 1–5 (2015). https://doi.org/10.1109/ICUWB.2015.7324491
43. Zhou, F., Hu, R.Q., Li, Z., Wang, Y.: Mobile edge computing in unmanned aerial vehicle networks. IEEE Wirel. Commun. **27**(1), 140–146 (2020)
44. Zhou, Z., Chen, X., Li, E., Zeng, L., Luo, K., Zhang, J.: Edge intelligence: paving the last mile of artificial intelligence with edge computing. Proc. IEEE **107**(8), 1738–1762 (2019)
45. Zhu, R., Liu, L., Song, H., Ma, M.: Multi-access edge computing enabled Internet of Things: advances and novel applications (2020)

Crowdsourcing, Platforms and Localization

A Novel Task Assignment Adjustment Method in Spatial-Temporal Crowdsourcing

Bingyi Sun[1,3], Jiaxu Cui[2,3], Hongtao Bai[1,3], and Yonggang Zhang[2,3](✉)

[1] Public Computer Education and Research Center, Jilin University, Changchun, China
{bysun,baiht}@jlu.edu.cn
[2] College of Computer Science and Technology, Jilin University, Changchun, China
{cjx,zhangyg}@jlu.edu.cn
[3] Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun 130012, People's Republic of China

Abstract. With the rapid development of mobile networks and the ubiquity of mobile devices, spatial-temporal crowdsourcing, which refers to assigning spatial-temporal tasks to moving workers, has drawn increasing attention. Many researchers aim at various task assignment methods in spatial-temporal crowdsourcing. However, unexpected situations reduce the reliability of the original assignment, such as the absence of reserved workers. To solve the problem, we propose a novel task assignment adjustment method in spatial-temporal crowdsourcing. We design a multi-objective optimization algorithm to minimize the adjustment and maximize the total matching degree in the reassignment process. The experimental results on three real data sets show that the proposed method can improve the total matching degree by about 10% while minimizing the adjustment compared with the baselines.

Keywords: Task assignment · Spatial-temporal crowdsourcing · Multi-objective optimization

1 Introduction

With the increasing popularity of intelligent mobile devices and the rapid development of wireless network technology, Spatial-temporal Crowdsourcing (SC), as a new crowdsourcing model, came into being. Different from traditional crowdsourcing, tasks of SC are spatial-temporal dependent, which means requesters submit tasks with specified execution locations and sensing period. As the smart device carriers, workers need physically move to tasks' locations and accomplish them within the task sensing period.

In spatial-temporal crowdsourcing, task assignment is a necessary research issue, which directly relates to the platform revenue and user experience. However, almost all of the solutions to solve the critical issue follow the rule of

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 105–124, 2024.
https://doi.org/10.1007/978-3-031-63992-0_7

immutability [21–23], i.e., after the platform formulates the task allocation scheme, the scheme will not change [11]. Actually, the assignment schemes need to be adjusted dynamically, which is contrary to the rule of immutability. For example, a worker with higher matching degree logs in the task assignment platform at a certain moment, but the platform has recruited enough workers to the spatial-temporal task. According to the rule of immutability, even though the worker has a high matching degree with the spatial-temporal task, he cannot get the opportunity to execute it [19].

Some researchers try to predict the worker future state, and assign the spatial-temporal tasks according to the prediction [4,28,34]. However, they ignore the prediction bias and the workers' willingness. It will lead to the absence of selected workers, causing low assignment quality. Consider the following example: Referring to workers' historical trajectories, the SC platform finds that Bob is often near McDonald's at around 12:30 p.m. on working days, so the platform plans to assign Bob to the task, which is taking photos of the customer flow near the McDonald. But actually, Bob has applied for half a day off because of a bad cold and driven home at 12:00 p.m., so he could not accomplish the task, and the platform should reassign it.

The above example shows that in practical situations, due to the absence of workers, the prediction-based spatial-temporal task assignment needs to be adjusted appropriately to ensure the reliability. Gummidi et al. also pointed that breaking the rule of immutability in spatial-temporal crowdsourcing task assignment would help to reallocate tasks, and it advantages to improve the total matching degree and task acceptance [11]. However, the following challenges still exist during the task assignment adjustment process. First, when should the spatial-temporal crowdsourcing tasks be reassigned? The platform may find workers will not perform the task at different times, so it is difficult to find a fixed time to reallocate the task. If the reassignment occurs after the start time of spatial-temporal task, other platforms may book the potential workers. Secondly, how to reassign spatial-temporal crowdsourcing tasks? The original assignment scheme has existed, an important problem is how to adjust the original assignment scheme properly to ensure the normal execution of all tasks.

To deal with the above challenges, this article proposes a Spatial-temporal Task Reassignment Method (STR) based on prediction, which consists of four modules: monitoring module, judgement module, candidate worker and task selection module, and task reassignment module. For the first challenge, when the number of workers for a spatial-temporal task is less than the lower boundary of task requirement, the monitoring module and the judgement module will work together to trigger a task reassignment process. The candidate worker and task selection module and the task reassignment module are designed for the second challenge. When task reassignment is triggered, the two module search for candidate tasks and workers, and then reassign the spatial-temporal tasks to minimize the adjustment to the original scheme and maximize the total matching degree.

The main contributions are summarized as follows:

– Propose a reassignment method for spatial-temporal tasks based on prediction. This method breaks the rule of immutability in spatial-temporal

task assignment based on prediction to improve the reliability of assignment scheme.
- To maximize the total matching degree and minimize the adjustment of the original assignment scheme, we employ the multi-objective algorithm to reassign the spatial-temporal tasks.
- We evaluate the proposed spatial-temporal task reassignment method on three different real data sets to verify its effectiveness.

The remainder of the article is organized as follows. In Sect. 2, we review related work on task assignment in SC. In Sect. 3, we first give an overview about basic concepts used in the article, and then define the assignment problem. Section 4 introduces the process of STR, and then presents each module in detail. Section 5 verifies the proposed method through experiments. In Sect. 6, we conclude the article and outline the future work.

2 Related Work

According to the availability of task and worker information, spatial-temporal crowdsourcing task assignment can be divided into two categories: online task assignment and offline task assignment [2].

Online task assignment means the spatial-temporal task is pre-submitted to the platform, but the worker enters the platform dynamically. The platform does not know which workers will log in before the task starts, but needs to decide whether to allocate spatial-temporal tasks to the workers immediately according to availability information [3,6,31]. Liu et al. recruited workers to perceive the traffic flow within the task sensing period, and then coarsely selected the most suitable road section for deployment according to the perception data [16]. Cheng et al. considered combining different task assignment platforms to jointly complete multiple tasks online. As each platform focuses on different areas, they complete tasks as much as possible by assisting each other and sharing remuneration [6]. Liu et al. dynamically adjusted assignment schemes in the online spatial-temporal task assignment process by constantly monitoring the information of spatial-temporal tasks and workers obtained currently [13,14].

In offline task assignment, the platform owns worker information before the task starts, and based on the historical records or prediction information to allocate the tasks to workers [1,12,30]. Offline spatial-temporal crowdsourcing task assignment mainly includes three methods: greedy strategy, task assignment based on graph model and task assignment based on machine learning method. Greedy strategy is widely used in offline spatial-temporal task assignment. Zhao et al. defined a task acceptance probability, and then in order to obtain multiple worker groups, they employed greedy strategy to calculate the maximum task acceptance probability. Workers in one group cooperate to complete spatial-temporal tasks. She et al. proposed the Greedy-GEACC algorithm. Workers select the tasks they are interested in, and ensure that the tasks do not conflict with each other [32]. They pointed out that the Greedy-GEACC algorithm has the advantage of strong scalability in large datasets [17]. Wei

et al. proposed a greedy GP-BS method to select the most effective matching pair of sensing region and workers by jointly considering the relationship among perceptual regions and worker mobility [25]. In the task assignment based on graph model, the spatial-temporal task assignment is transformed into minimum cost maximum flow (MCMF) problem. HCDT model was proposed in 2019 [33]. After obtaining the time preference tensor of workers, HCDT model converts the spatial-temporal tasks assignment into MCMF problem to obtain the optimal solution satisfying the execution distance constraints of workers and the time constraints of spatial-temporal tasks [35]. Machine learning is a hot research topic currently. The spatial-temporal task assignment also employs methods based on machine learning. Wang et al. proposed the EMOPSO algorithm, which uses an improved particle swarm optimization algorithm to obtain the assignment scheme to achieve the two objectives of task coverage maximization and platform cost minimization [24]. Zhu et al. proposed DLMV method to collect environmental information at designated points in the city [37]. They predicted vehicle trajectory by deep learning, and then assigned tasks to vehicles according to the predicted vehicle trajectory.

However, all the above research follows the rule of immutability during the task assignment by default, so they cannot deal with the failure of spatial-temporal task execution caused by the absence of workers. Some recent work has pointed the irrationality of the rule of immutability [7,11]. Cheng et al. proposed a series of greedy assignment methods to solve the problem that activities cannot be executed normally caused by the attribute information changes of workers or activities [7]. On the basis of satisfying the constraint of minimizing changes to the original allocation scheme as much as possible, the matching degree of all activities should be maximized.

Our work is different from the existing research in the following four aspects: 1) Compared with the online spatial-temporal task assignment method, the assignment in our work is before the task start time, which avoids potential worker losing due to insufficient workers. 2) Compared with the offline spatial-temporal task assignment method, our work focuses on the adjustment of the existing scheme, so it breaks the rule of immutability that most work follows. 3) Compared with the work of Cheng et al. [7], our reassignment method has two objectives, so it is more practical. 4) In the process of reassignment, we not only consider the effect of deleting edges from the original assignment graph, but also the effect of adding edges to the original assignment graph.

3 Problem Statement

In this section, we introduce a set of necessary preliminary concepts. Table 1 lists the definitions used in the article.

The worker set is W, $W = \{w_1, w_2, ..., w_{|W|}\}$, each worker has attribute set At_w. Specifically, At_w contains tag_w, ps_w, max_w and b_w. tag_w is the tag set preferred by the worker. ps_w shows the sequence of location and time tuples that workers have appeared, $ps_w = \{(\text{location}, \text{time}), ...\}$. max_w is the upper

Table 1. Symbols and descriptions.

Symbol	Description
s	A spatial-temporal task
At_s	Attribute set of task s
l_s	Location of task s
t_s^{st}	Start time of task s
t_s^{et}	End time of task s
tag_s	Tag set of task s
c_s	Cost of performing task s
ub_s	Upper boundary of workers required by s
lb_s	Lower boundary of workers required by s
(lo_s, la_s)	GPS coordinates of s location
S	Spatial-temporal task set
w	A worker
At_w	Attribute set of worker w
ps_w	Sequence of location and time tuples
tag_w	Tag set of task w
b_w	Budget of w
max_w	Upper boundary of tasks planned to complete by w
W	Worker set
γ	Trigger task
W_γ	Available worker set of γ
S_γ^c	Candidate task set of γ
W_γ^c	Candidate worker set of γ
$\mathbf{M}_{ij}$	Matching degree between w_i and s_j
$\mathbf{M}$	Matching degree Matrix
$\mathcal{G}$	Bipartite graph of original assignment scheme
$\mathcal{G}'$	Bipartite graph of reassignment scheme
$\mathcal{W}$	Worker node set in bipartite graph
$\mathcal{S}$	Task node set in bipartite graph
$\mathcal{E}$	Edge set in bipartite graph

boundary of the number of spatial-temporal tasks that the worker plans to complete. b_w is the worker budget. Similarly, the spatial-temporal task set is S. For each spatial-temporal crowdsourcing task s, At_s represents its attribute set, including task execution location (l_s), start time (t_s^{st}), end time (t_s^{et}), tag set (tag_s), upper boundary of workers required by the task (ub_s), lower boundary of workers required by the task (lb_s), and possible cost of completing the task (c_s). Locations are represented by GPS coordinates ((lo_s, la_s)).

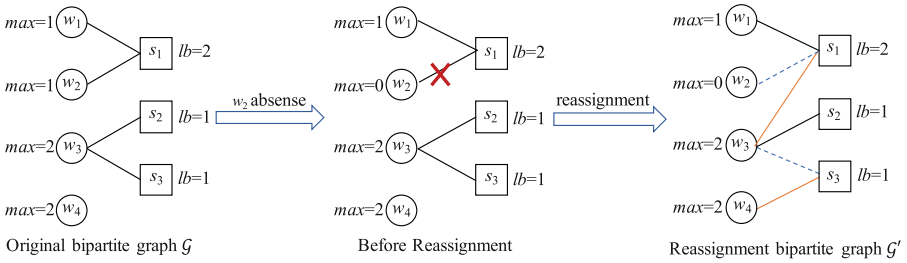


Fig. 1. An example of a bipartite graph of spatial-temporal task reassignment.

By mining and calculating worker trajectory and historical matching information, we predict the matching matrix of worker set and spatial-temporal task set, represented by $\mathbf{M}$, where each element $\mathbf{M}_{ij}$ is the matching degree of worker w_i and spatial-temporal task s_j. $\mathcal{G}$ represents the bipartite graph constituted by the original assignment scheme, and $\mathcal{G}'$ represents the bipartite graph after reassignment, as shown in Fig. 1. Due to the absence of w_2, the lower boundary of workers for s_1 is not met and needs to be reassigned. Therefore, s_1 becomes the trigger spatial-temporal task γ, W_γ is the set of available workers of γ, including all workers except w_2. Comparing with $\mathcal{G}$, $\mathcal{G}'$ deletes edges $e(w_2, s_1)$ and $e(w_3, s_3)$, adds edges $e(w_3, s_1)$ and $e(w_4, s_3)$. We employ $|\mathcal{G} - \mathcal{G}'|$ to represent the number of edges deleted from the original assignment graph, and $|\mathcal{G}' - \mathcal{G}|$ to represent the number of edges added to the original bipartite graph. Therefore, in Fig. 1, $|\mathcal{G} - \mathcal{G}'| = 2$, $|\mathcal{G}' - \mathcal{G}| = 2$.

Problem Formulation: Given the worker set W, spatial-temporal task set S, predicted workers and task matching matrix $\mathbf{M}$, and the current assignment scheme $\mathcal{G}$. When a task becomes a trigger task γ, we select the candidate worker set W_γ^c and candidate spatial-temporal task set S_γ^c from W and S respectively, and then reassign in the solution space composed of γ, W_γ^c and S_γ^c. The objectives of the reassignment scheme $\mathcal{G}'$ are to maximize the matching probability of all tasks and minimize the adjustment of the original allocation scheme under the constraints, as shown in Eq. (1)–Eq. (6):

$$maximize \quad \sum_{s_j \in S_\gamma^c \cup \gamma} \sum_{w_i \in W_\gamma^c} \mathbf{M}_{ij} \cdot \theta_{ij} \tag{1}$$

$$minimize \quad |\mathcal{G} - \mathcal{G}'| \times d + |\mathcal{G}' - \mathcal{G}| \times a \tag{2}$$

$$s.t. \quad \theta_{ij} = \begin{cases} 1, & \text{if} \quad e(w_i, s_j) \in \mathcal{G}'.\mathcal{E}, \\ 0, & \text{otherwise}, \end{cases} \tag{3}$$

$$\forall w_i \in W_\gamma^c, \quad \sum_{j=1}^{|S|} \theta_{ij} \leq max_{w_i}, \tag{4}$$

$$\forall s_j \in S_\gamma^c, \quad lb_{s_j} \leq \sum_{i=1}^{|W|} \theta_{ij} \leq ub_{s_j}. \tag{5}$$

$$\forall w_i \in W_\gamma^c, \quad \sum_{j=1}^{|S|} \theta_{ij} c_{s_j} \leq b_{w_i}, \tag{6}$$

where θ_{ij} indicates whether w_i has been assigned to s_j in $\mathcal{G}'$. $\theta_{ij} = 1$ means s_j will be allocated to w_i and vice versa (Eq. (3)). Adjustments to the original assignment scheme include deleting edges from the original assignment bipartite graph and adding edges to the graph. d in Eq. (2) represents the cost of deleting edges, and a means the cost of adding edges. Therefore, Eq. (2) reflects the objective of reducing the adjustment to the original assignment scheme as much as possible. Equation (4) and Eq. (5) show the limitation of the number of tasks that workers plan to participate in and the worker requirement limitation of spatial-temporal tasks, respectively. Equation (6) shows the budget constraint of workers.

4 Spatial-Temporal Task Reassignment Method

In this section, we first introduce the workflow of the reassignment method for spatial-temporal tasks, as shown in Fig. 2, to answer the two questions of when and how to do the reassignment. Then, we introduce the candidate worker and task selection module and the task reassignment module in detail.

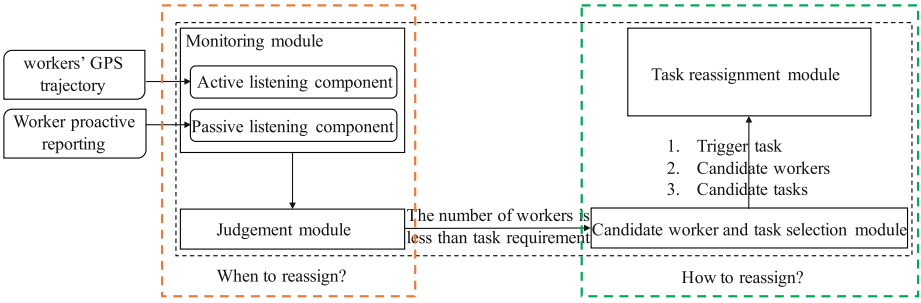


Fig. 2. STR method workflow.

4.1 Workflow of Spatial-Temporal Task Reassignment

As shown in Fig. 2, the platform uses the monitoring module to interact with the selected workers after formulating the original assignment scheme for spatial-temporal tasks. If the trajectories of workers can be monitored, the platform will collect the real-time GPS trajectory through the active listening component in

the monitoring module. When the platform finds that the reserved worker cannot reach the task location within the task sensing period, it transmits the worker ID to the judgement module. If it is difficult for the platform to collect the trajectories of workers, a passive listening component is used to collect workers' report, and then transmit the ID of the reserved worker to the judgement module. We believe the monitoring method is feasible, because if workers agree with the platform to collect trajectories, the platform does not need to disturb workers' normal activities to calculate whether workers can reach the task location within the task sensing period, and if it is difficult for the platform to collect worker trajectories, absent workers can actively report in advance. By interacting with the workers, the monitoring module finds the absent workers as early as possible to get sufficient time for the subsequent reassignment.

After sending the ID of absent workers and the ID of spatial-temporal tasks that may become trigger tasks to the judgement module, the judgement module measures whether the employers of the spatial-temporal task is less than the lower boundary of workers required by it. If it is less than the lower boundary, the spatial-temporal task is regarded as a trigger task. The absent worker IDs are transmitted to the candidate worker and task selection module to touch off the reassignment process. To reserve workers with high matching degree as much as possible and reduce the risk of being recruited by other spatial-temporal crowdsourcing platforms, tasks will be reassigned once the platform deems it as a trigger task.

In order to improve the efficiency of spatial-temporal task reassignment, we design the candidate worker and task selection module to select candidate task set S_γ^c and candidate worker set W_γ^c, and regard these two sets and the trigger task γ as the solution space to reduce the search space of reassignment scheme. After selecting the candidate workers and the candidate spatial-temporal tasks, the task reassignment module starts to work. The objectives of the reassignment scheme $\mathcal{G}'$ is to maximize the total matching degree and minimize the modification of the original allocation scheme under constraints.

4.2 Candidate Worker and Task Selection Module

In the candidate worker and task selection module, we use two methods to find suitable candidate worker set and spatial-temporal task set: matching degree sorting method and selecting similar workers (spatial-temporal task).

Matching Degree Sorting. Since one objective of reassignment is to maximize the total matching degree, it expects to select some candidate workers and candidate spatial-temporal tasks to ensure that the matching degree of candidate workers and γ is higher than with the original assignment task, i.e., candidate task. The matching degree sorting selection process is shown in Algorithm 1.

First, the candidate task set $S_\gamma^{c'}$ and the candidate worker set $W_\gamma^{c'}$ are initialized (line 1). Next, we initialize a list H to store pairs of workers and tasks in line 2. Then, for each spatial-temporal task s_j, check whether the number of reserved

Algorithm 1. Matching degree sorting

Input: **M**: matching degree matrix; S: spatial-temporal task set; γ: trigger task; $\mathcal{G}$: original assignment scheme; β: selecting number.
Output: $S_\gamma^{c'}$: candidate task set of γ; $W_\gamma^{c'}$: candidate worker set of γ.

1: Initialize $S_\gamma^{c'}$ and $W_\gamma^{c'}$: $S_\gamma^{c'} \leftarrow \varnothing$, $W_\gamma^{c'} \leftarrow \varnothing$;
2: Initialize a list H to store pairs of workers and tasks;
3: **for** $\forall s_j \in S$ **do**:
4: **if** $|e(w_*, s_j) \in \mathcal{G}.\mathcal{E}| > lb_{s_j}$ **then**:
5: **for** each w_i assigned to s_j in $\mathcal{G}$ **do**:
6: **if** w_i is not absent **then**:
7: Calculate $\Delta = \mathbf{M}_{i\gamma} - \mathbf{M}_{ij}$;
8: Insert w_i and s_j into H by descending order of Δ;
9: **end if**
10: **end for**
11: **end if**
12: **end for**
13: $S_\gamma^{c'} \leftarrow$ top β tasks in H;
14: $W_\gamma^{c'} \leftarrow$ top β workers in H;
15: **return** $S_\gamma^{c'}$ and $W_\gamma^{c'}$;

workers in the current assignment scheme is greater than the lower boundary of workers required by s_j. If so, it means that s_j can provide workers for the trigger task γ (lines 3 and 4), where w_* represents any worker node connected to s_j in $\mathcal{G}$. If the number of workers reserved for s_j is greater than the lower boundary of workers required by s_j, we calculate the matching degree gain Δ brought by reassigning the workers w_i to the trigger task γ (line 7). Then, insert (w_i, s_j) into the temporary storage list H according to the descending order of the matching degree gain (line 8). Finally, we select the first β spatial-temporal tasks from H as the candidate task set obtained by sorting method (line 13), and the first β workers as the candidate worker set obtained by sorting method (line 14).

Select Similar Candidate Workers and Spatial-Temporal Tasks. We select similar candidate workers from three perspectives: workers with similar spatial-temporal relation to the trigger task, workers with similar attributes to the absent worker, and workers of other tasks that have similar attributes to the trigger task. We will introduce the three approaches in detail. Algorithm 2 shows the process of selecting candidate workers and spatial-temporal tasks by similarity.

A) Workers who are similar to the trigger task. Because the spatial-temporal task has strict restrictions on sensing period and location, only the selected workers arrive at the designated location within the sensing period, task could be completed. Therefore, it is necessary to select workers who has similar spatial-temporal relationship to the trigger task. We mine the historical

Algorithm 2. Similarity ranking

Input: $\mathbf{M}$; W_γ; S; γ; $\mathcal{G}$; α.
Output: $S_\gamma^{c''}$: similar candidate task set; $W_\gamma^{c''}$: similar candidate worker set.

1: Initialize $S_\gamma^{c''}$ and $W_\gamma^{c''}$: $S_\gamma^{c''} \leftarrow \varnothing$, $W_\gamma^{c''} \leftarrow \varnothing$;
2: Initialize three storage list: H_1, H_2, H_3;
3: **for** $\forall w_i \in W_\gamma$ **do**:
4: Calculate the similarity between w_i and γ, and then store it into H_1;
5: Calculate the similarity between w_i and the absent worker, and store it intoH_2;
6: **end for**
7: Rank the workers in H_1 and H_2 in descending order, and select the first α workers from H_1, and then store them into $W_\gamma^{c''}$. Select the first α workers from H_2, who does not in $W_\gamma^{c''}$, and then store them into $W_\gamma^{c''}$;
8: **for** $\forall s_j \in S$ **do**:
9: Calculate the similarity, $sim_{s_j,\gamma}$, between s_j and γ, and store it into H_3;
10: **end for**
11: Sort tasks in H_3 in descending order;
12: **while** $\alpha > 0$ **do**:
13: $s_j = \mathrm{Pop}(H_3)$;
14: $W_\gamma^{c''} \leftarrow \{w_i | w_i \in \{e(w_*, s_j) \in \mathcal{G}.\mathcal{E}\} \quad \text{and} \quad w_i \notin W_\gamma^{c''}\}$;
15: $\alpha = \alpha - |\{w_i | w_i \in \{e(w_*, s_j) \in \mathcal{G}.\mathcal{E}\} \quad \text{and} \quad w_i \notin W_\gamma^{c''}\}|$;
16: **end while**
17: **for** $w_i \in W_\gamma^{c''}$ **do**:
18: Initialize storage list H;
19: **for** each s_j reserve w_i in $\mathcal{G}$ **do**:
20: Calculate $\Delta = \mathbf{M_{i\gamma}} - \mathbf{M_{ij}}$, and insert w_i and s_j into H by descending order of Δ;
21: **end for**
22: $S_\gamma^{c''} \leftarrow S_\gamma^{c''} \cup Pop(H)$;
23: **end for**
24: **return** $S_\gamma^{c''}$ and $W_\gamma^{c''}$;

trajectories and calculate the frequency of workers passing through the specified location during the trigger task sensing period. Then, select the first α available workers with high frequency to join the candidate worker set $W_\gamma^{c''}$. The process is shown in Lines 4 and 7 of Algorithm 2.

B) Workers with similar attributes to the absent workers. Workers' historical trajectories, preferences and execution records play a decisive role in predicting whether they can complete the spatial-temporal task. If the workers could not reach the specified location within the sensing period, they would not be able to complete the spatial-temporal task. Similarly, if the workers are not interested in the task, they would not execute it. Therefore, selecting workers with similar attributes to the absent workers can improve the suitability of candidate workers in three aspects: time, space and preference, and this method effectively reduces the search space. Specifically, we get the spatial-temporal matrix $\mathbf{LT}_{w_{i'}}$ of all absent workers of the triggering task γ

and the spatial-temporal matrix $\mathbf{LT}_{w_i}$ of available worker w_i by prediction [20,26], and then design Eq. (7) to calculate the spatial-temporal similarity between the absent workers and an available worker w_i, which is shown by $sim^{lt}_{w_i}$:

$$sim^{lt}_{w_i} = \left\| (\sum_{w_{i'} \in W_{ab}} \mathbf{LT}_{w_{i'}}) - \mathbf{LT}_{w_i} \right\|_2, \tag{7}$$

The W_{ab} is the absent worker set of the trigger task γ. Then, we calculate the similarity between the tag set of absent workers and w_i by set similarity, as shown in Eq. (8):

$$sim^{tag}_{w_i} = \frac{(\bigcup_{w_{i'} \in W_{ab}} tag_{w_{i'}}) \cap tag_{w_i}}{(\bigcup_{w_{i'} \in W_{ab}} tag_{w_{i'}}) \cup tag_{w_i}}, \tag{8}$$

Therefore, the similarity between the absent workers and w_i is:

$$sim_{w_i} = sim^{lt}_{w_i} + sim^{tag}_{w_i}, \tag{9}$$

After obtaining the similarity of available workers and the absent workers, we select the first α workers with the high similarity and add them into the similar candidate worker set $W^{c''}_{\gamma}$. As shown in lines 5 and 7 of Algorithm 2.

C) Workers of other tasks that have similar attributes to trigger tasks. If the attributes of a task s_j are similar to the trigger task γ, the workers performing s_j may have a high matching degree with γ. In order to find other similar tasks with γ, we measure the attribute similarity from four dimensions: location, time, tag and other scalar attributes. Intuitively, if the distance between s_j and γ is closer, the location similarity between them is higher. Therefore, we design Eq. (10) as the attenuation function to represent the similarity, and it will decrease as distance increases.

$$sim^{dist}_{s_j,\gamma} = e^{-g \cdot dist(s_j,\gamma)}, \tag{10}$$

where $dist(s_j, \gamma)$ is the distance function used to calculate the distance between γ and s_j, and g is a hyperparameter. $sim^{dist}_{s_j,\gamma}$ belongs to the interval of $(0, 1]$. For the time attribute of two spatial-temporal tasks, the more overlapping sensing period, the more similar they will be. Therefore, we employ the set similarity method to calculate the time similarity between γ and s_j.

$$sim^{time}_{s_j,\gamma} = \frac{\{t^{st}_{s_j}, t^{st}_{s_j} + 1, \cdots, t^{et}_{s_j}\} \cap \{t^{st}_{\gamma}, \cdots, t^{et}_{\gamma}\}}{\{t^{st}_{s_j}, t^{st}_{s_j} + 1, \cdots, t^{et}_{s_j}\} \cup \{t^{st}_{\gamma}, \cdots, t^{et}_{\gamma}\}}, \tag{11}$$

If the two tasks have more identical tags, they will be more similar in tags. The calculation formula of tag similarity is as follows:

$$sim^{tag}_{s_j,\gamma} = \frac{tag_{s_j} \cap tag_{\gamma}}{tag_{s_j} \cup tag_{\gamma}}, \tag{12}$$

Since other attributes of spatial-temporal tasks are scalars, we use a tuple to contain them. We calculate the similarity between attribute tuples of different tasks by L2 normal form.

$$sim_{s_j,\gamma}^{other} = \left\| (ub_{s_j}, lb_{s_j}, c_{s_j}) - (ub_\gamma, lb_\gamma, c_\gamma) \right\|_2, \tag{13}$$

Therefore, the similarity between the attributes of γ and s_j can be obtained.

$$sim_{s_j,\gamma} = sim_{s_j,\gamma}^{dist} + sim_{s_j,\gamma}^{time} + sim_{s_j,\gamma}^{tag} + sim_{s_j,\gamma}^{other}. \tag{14}$$

We sort the spatial-temporal tasks according to attribute similarity in descending order, and the first α workers who are not in the candidate worker set are selected and added into the similar candidate worker set $W_\gamma^{c''}$. As shown in lines 8 to 16 of Algorithm 2.

The same as matching degree sorting, to get similar task candidate set, we check the original assignment scheme of each candidate worker,i.e., w_i. If w_i performing γ brings a greater gain in matching degree than performing the originally assigned task s_j, we add s_j into the temporary storage list H. The spatial-temporal tasks in H are sorted according to the gain of matching degree. The s_j that brings the maximum matching degree gain is taken from H and added into the candidate spatial-temporal task set $S_\gamma^{c''}$. As shown in lines 17 through 23 of Algorithm 2.

Candidate Workers and Candidate Spatial-Temporal Tasks. The final set of candidate workers is $W_\gamma^c = W_\gamma^{c'} \cup W_\gamma^{c''}$. Similarly, we obtain the final set of candidate spatial-temporal tasks by $S_\gamma^c = S_\gamma^{c'} \cup S_\gamma^{c''}$.

4.3 Task Reassignment Module

As the problem definition in Eq. (1)–Eq. (6), there are two objectives during reassignment, which are maximizing the matching degree of all tasks and minimizing the changes to the original assignment scheme, so we seek multi-objective optimization methods to achieve the reassignment. In addition, since the search space of the reassignment has been greatly reduced by the candidate worker and task selection module, only including γ, the candidate worker set W_γ^c, and the spatial-temporal task set S_γ^c. In order to obtain the global optimal assignment scheme as far as possible, we explore several swarm intelligence algorithms which can solve the multi-objective problem in the task reassignment module.

We try to integrated many multi-objective swarm intelligence algorithms, such as awGA [10], NSGA2 [9], MOEAD [29], NSGA3 [8] and RVEA [5], and decide to employ NSGA3 as the default multi-objective reassignment method through experiments. Each individual on the Pareto front obtained by the NSGA3 is an optimal reassignment scheme, and we could select according to the actual demand.

5 Performance Evaluation

This section will answer the following questions through experiments: (1) Comparing the multi-objective swarm intelligence algorithms integrated into the task reassignment module, which algorithm is more suitable? (2) Comparing with the baselines, how much doe effectiveness of single trigger task reassignment improve by STR method? How much effectiveness does the two selecting candidate workers and tasks methods contribute, respectively? (3) How much influence of the adding edges cost and the deleting edges cost on the original bipartite graph during reassignment? (4) Compared with the baselines, how much does effectiveness of dynamic multi-task reassignment improve by using STR method?

5.1 Dataset and Experimental Settings

We conduct extensive experiments on three real-world datasets, Geolife [36], F-NYC [27] and Meetup-HT [15]. Geolife is a trajectory intensive dataset. F-NYC is a check-in dataset, which includes sparse check-in records. Meetup-HT is an event-based social network dataset, which includes historical participation records. We follow the preprocessing method in [20,26] to get datasets and matching degree matrix obtained through prediction. The information of the three datasets is shown in Table 2, where $|\boldsymbol{R}|$ is the record number of all workers, and $|\boldsymbol{L}|$ is the number of locations in the dataset.

Table 2. Basic statistics of datasets.

Dataset	$\mid W \mid$	$\mid S \mid$	$\mid tag \mid$	$\mid R \mid$	$\mid L \mid$
Geolife	85	150	68	11091	114
F-NYC	272	600	6445	12	33150
Meetup-HT	644	1421	2981	9687	–

We choose the total matching degree (ToMD) [7,18] and adjustment cost (Cost) as the metrics of reassignment methods. ToMD shows the reliability of the reassignment scheme compared with the original scheme. According to the second objective function, the reassignment scheme should minimize adjustments to the original assignment strategy as much as possible, so we set Cost to monitor the adjustment. These two metrics provide a clear description about reassignment schemes on different objectives.

We integrate Geatpy[1], which is written by Python, into the spatial-temporal task reassignment module to achieve the multi-objective swarm intelligence algorithms, such as awGA, NSGA2, MOEAD, NSGA3 and RVEA. ξ-In [7], which is the most advanced spatial-temporal crowdsourcing task reassignment method, is adopted as a baseline. In order to reflect the necessity of the sorting method

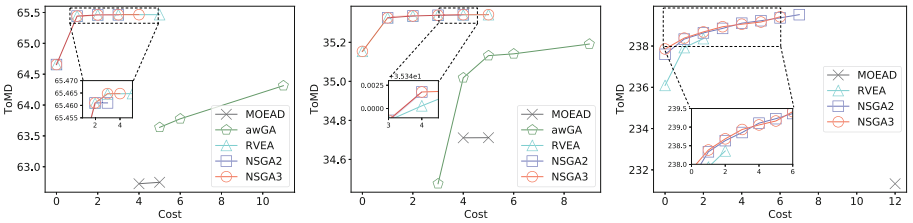
[1] http://geatpy.com/.

and the similarity method during candidate worker and spatial-temporal task selection process, we design two additional comparison methods: STR-Sort and STR-Sim, which only use sorting method or similarity method to select candidate workers and tasks, respectively. In addition, in order to reflect the difference between the original allocation scheme and the reassignment scheme facing absent workers, we use Non-reass to represent the total matching degree after worker absence but without reassignment. As the ξ-In algorithm only considers the effect of removing edges from the original allocation scheme and deems it as a constraint, to compare with ξ-In, the default value of removing edges cost (d) is 1, and adding edges cost (a) is 0.

All experiments are performed on the same hardware environment, equipped with Intel (R) Xeon (R) CPU E5-2680 v4@2.40GHz and 128 GB memory.

5.2 Comparison of Multi-objective Reassignment Algorithms

In order to answer the first question, we compare the swarm intelligence algorithms mentioned in Sect. 4.3 in the spatial-temporal crowdsourcing task reassignment. Figure 3 shows the Pareto fronts of different multi-objective algorithms.



(a) Pareto frontiers of different multi-objective optimization algorithms on Geolife

(b) Pareto frontiers of different multi-objective optimization algorithms on F-NYC

(c) Pareto frontiers of different multi-objective optimization algorithms on Meetup-HT

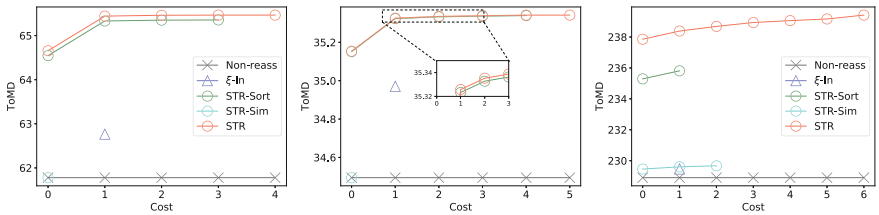
Fig. 3. Comparison of different multi-objective algorithms in the task reassignment module.

In Fig. 3, both NSGA2 and NSGA3 algorithms present ascendant Pareto fronts in the three datasets. We find that NSGA3 algorithm is slightly better than NSGA2 in the local magnification, because NSGA3 preserves more population diversity in the process of selecting the next generation. Therefore, we select NSGA3 as the default reassignment algorithm in the spatial-temporal task reassignment module. The Pareto front of RVEA algorithm is slightly lower than NSGA2 and NSGA3, which may be because RVEA algorithm is suitable for high-dimensional space composed of more objective functions. Compared with the above three algorithms, awGA and MOEAD algorithms do not find the solution when the Cost is 0 or 1, and in the case of overspending, the total matching degree is lower than NSGA3, NSGA2 and RVEA algorithms. In addition, MOEAD algorithm fails to find a feasible allocation scheme on the

Meetup-HT dataset. Therefore, in the task reassignment module, the algorithm that uses non-dominated sorting to solve the multi-objective problem is more suitable than the algorithm that transforms the multiple objectives into a single objective.

5.3 Comparison of Single Trigger Task Reassignment

To answer the second question, we randomly select reserved workers as absent workers. When a spatial-temporal task becomes a trigger task, the platform will reassign it. We compare the STR method with all of the baselines, and show the results in Fig. 4.



(a) The impact of different reassignment methods on the two evaluation metrics on Geolife

(b) The impact of different reassignment methods on the two evaluation metrics on F-NYC

(c) The impact of different reassignment methods on the two evaluation metrics on Meetup-HT

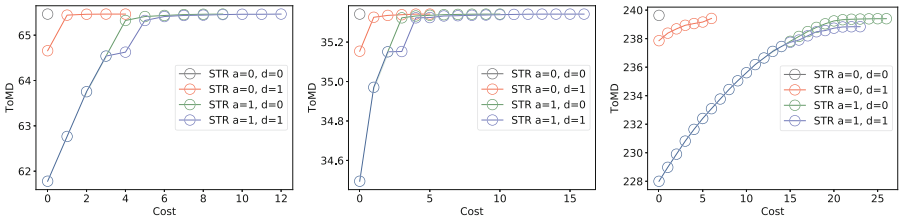
Fig. 4. Comparison of single trigger task reassignment by different methods.

The ToMD of Non-reass method is the lowest among all the methods on the three datasets, because the trigger task γ cannot be completed due to the absence of workers. According to the original assignment scheme, we calculate the total matching degree by spatial-temporal tasks except γ. Since the ξ-In reassignment method has one objective, we obtain only one reallocation scheme, as shown in Fig. 4, where is the blue triangle. Comparing with Non-reass method, the ToMD improves dramatically, but ξ-In deletes edges on the original allocation scheme, so the Cost is not 0 on all three datasets, and its ToMD is lower than STR method. Then discuss STR-Sort and STR-Sim methods. Comparing with STR-Sim, both of the STR-Sort and STR bring higher ToMD under the same Cost, but the Pareto front of STR-Sort is lightly lower than STR on the three data sets. Therefore, we consider that STR-Sort contributes more to STR, and STR-Sim could provides supplement to STR-Sort on selecting candidate workers and spatial-temporal task.

5.4 The Cost of Different Operational on the Reassignment Scheme

To answer the third question, we set four cost coefficient for evaluating the second objective, which are $a = 0, d = 0$; $a = 0, d = 1$; $a = 1, d = 0$ and $a = 1, d = 1$. $a = 0, d = 0$ means that we do not consider the cost of adjustment to the original scheme, which indicates the reassignment process does not consider the

second objective. $a = 0, d = 1$ only focuses on the cost of deleting edges from the original assignment bipartite graph, which is consistent with the ξ-In method. $a = 1, d = 0$ indicates only concern about the cost of adding edges to the original assignment bipartite graph. $a = 1, d = 1$ means that the cost includes adding and deleting edges on the original assignment graph.



(a) Impact of the cost of different operations on Geolife

(b) Impact of the cost of different operations on F-NYC

(c) Impact of the cost of different operations on Meetup-HT

Fig. 5. The influence of the cost of different operations during task reassignment.

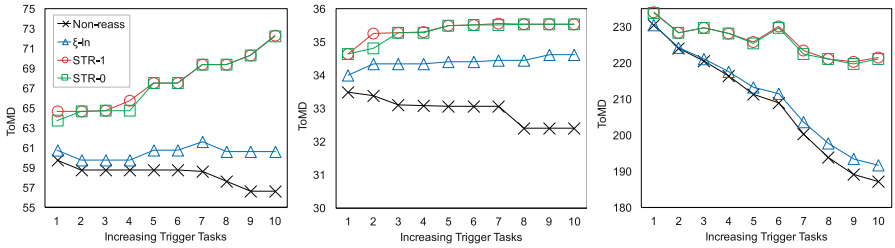
As shown in Fig. 5, $a = 0, d = 0$ is equivalent to only considering the first objective during the reassignment process, which does not concern the cost from changing assignment scheme, so it represents the highest ToMD. When set the cost coefficient according to the other three cases, the total matching degree of the STR method improves gradually as the total Cost increases. Among them, the Pareto front obtained by setting $a = 0, d = 1$ is more advanced than that obtained by setting $a = 1, d = 0$. It proves the effectiveness of the original assignment scheme, i.e., the platform assign spatial-temporal tasks to the workers with high matching degree. Therefore, the STR method is more willing to assign trigger task to the workers with residual execution capacity rather than snatch workers from other tasks during reassignment. When the cost coefficient is $a = 1, d = 1$, since both the deletion and addition of edges to the original scheme cause an increasing Cost, the Pareto front obtained by STR method with this setting is the lowest among the four coefficient settings.

5.5 Comparison of Dynamic Multi-task Reassignment

Next, we explore the case of reassigning multiple trigger tasks in succession. Due to space limitations, we show when the Cost is 0 and 1, the curve of the total matching degree changing with the increasing number of trigger tasks, and define them as STR-0 and STR-1, respectively. The results are represented in Fig. 6.

The Non-reass curve shows if the platform does not reassign the trigger task, the total matching degree will gradually decrease, so it is necessary to reassign trigger tasks. Comparing the ToMD curve of ξ-In and Non-reass, we find that the reassignment method is effective as the number of trigger tasks increases. Then, in Fig. 6, both of the total matching degree curves of STR-0 and STR-1 are higher

than that of ξ-In method on the three datasets, which shows the effectiveness of our STR. In addition, we find that the curve of STR-1 is slightly higher than STR-0. It proves swap scheme improves the total matching degree during the reassignment process, where the swap scheme means removing a certain worker from one spatial-temporal task, and assigning this worker to the trigger task.



(a) Reassignment of continuous multi trigger tasks on Geolife

(b) Reassignment of continuous multi trigger tasks on F-NYC

(c) Reassignment of continuous multi trigger tasks on Meetup-HT

Fig. 6. Comparison of Dynamic Multi-task reassignment by different methods.

6 Conclusion

In this article, to adjust task allocation scheme dynamically, we proposed a task reassignment method based on prediction in spatial-temporal crowdsourcing, which names STR. In STR, monitoring module and judgement module were used to determine whether trigger a task reassignment process. Once a task become a trigger task, we need select candidate workers and tasks by sorting and similarity methods to reduce the search space. Then, we formulated the reassignment as a multi-objective optimization problem, and employed swarm intelligence algorithms to solve it. Sufficient experiments built on three real datasets, and the results showed that comparing with the baselines, the STR method improves the total matching degree by about 10% while minimizing the adjustment on the original assignment scheme.

Although the proposed reassignment method can improve the total matching degree by about 10%, several further studies to make it more powerful are worth mentioning. How to balance the decision time and assignment quality will be considered in future work. Swarm intelligence algorithm has global optimal guarantee, but it often has a slow rate of convergence when dealing with a large number of participants and tasks. On the contrary, greedy strategies have advantages of short execution time when dealing with large-scale tasks, but can only obtain local optimal solutions. Therefore, how to balance assignment quality and decision time is worth exploring in the future.

Acknowledgements. This research was supported by the National Natural Science Foundation of China (62206105), the Natural Science Foundation of Jilin Province (20210101172JC, 20210101181JC), and the Fundamental Research Funds for the Central Universities, JLU.

References

1. Bei, X., Zhang, S.: Algorithms for trip-vehicle assignment in ride-sharing. In: Proceedings of the 32nd AAAI Conference on Artificial Intelligence, pp. 3–9 (2018)
2. Chen, Y., Lv, P., Guo, D., Zhou, T.: A survey on task and participant matching in mobile crowd sensing. J. Comput. Sci. Technol. **33**(4), 768–791 (2018)
3. Chen, Z., Cheng, P., Zeng, Y., Chen, L.: Minimizing maximum delay of task assignment in spatial crowdsourcing. In: Proceedings of the 35th IEEE International Conference on Data Engineering, pp. 1454–1465 (2019)
4. Cheng, P., Lian, X., Chen, L., Shahabi, C.: Prediction-based task assignment in spatial crowdsourcing. In: Proceedings of the 33rd IEEE International Conference on Data Engineering, pp. 997–1008 (2017)
5. Cheng, R., Jin, Y., Olhofer, M., et al.: A reference vector guided evolutionary algorithm for many-objective optimization. IEEE Trans. Evol. Comput. **20**(5), 773–791 (2016)
6. Cheng, Y., Li, B., Zhou, X., Yuan, Y., Wang, G.: Real-time cross online matching in spatial crowdsourcing. In: Proceedings of the 36th IEEE International Conference on Data Engineering, pp. 1–12 (2020)
7. Cheng, Y., Yuan, Y., Chen, L., Giraud-Carrier, C.: Event-participant and incremental planning over event-based social networks. IEEE/ACM Trans. Knowl. Data Eng. **33**(2), 474–488 (2021)
8. Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, Part I: solving problems with box constraints. IEEE Trans. Evol. Comput. **18**(4), 577–601 (2014)
9. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
10. Gen, M., Cheng, R.: Genetic Algorithms and Engineering Optimization. Wiley, New York (2000)
11. Gummidi, S.R.B., Xie, X., Pedersen, T.B.: A survey of spatial crowdsourcing. ACM Trans. Database Syst. **44**(2) (2019)
12. Liu, C.H., Piao, C., Ma, X., Yuan, Y.: Modeling citywide crowd flows using attentive convolutional LSTM. In: Proceedings of the 37th IEEE International Conference on Data Engineering, pp. 217–228 (2021)
13. Liu, W., Yang, Y., Wang, E., Wang, H.: Dynamic online user recruitment with (non-) submodular utility in mobile crowdsensing. IEEE/ACM Trans. Netw. **29**(5), 2156–2169 (2021)
14. Liu, W., Yang, Y., Wang, E., Wu, J.: Dynamic user recruitment with truthful pricing for mobile crowdsensing. In: 39th IEEE IEEE Conference on Computer Communications, pp. 1113–1122 (2020)
15. Liu, X., He, Q., Tian, Y., Lee, W., McPherson, J., Han, J.: Event-based social networks: linking the online and offline social worlds. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1032–1040 (2012)
16. Liu, Z., Chen, L., Tong, Y.: Realtime traffic speed estimation with sparse crowdsourced data. In: Proceedings of the 34th IEEE International Conference on Data Engineering, pp. 329–340 (2018)
17. She, J., Tong, T., Chen, L.: Conflict-aware event-participant arrangement. In: Proceedings of the 31st IEEE International Conference on Data Engineering, pp. 735–746 (2015)

18. She, J., Tong, Y., Chen, L.: Utility-aware social event-participant planning. In: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, pp. 1629–1643 (2015)
19. Song, T., Tong, Y., Wang, L., She, J.: Trichromatic online matching in real-time spatial crowdsourcing. In: Proceedings of the 33rd IEEE International Conference on Data Engineering, pp. 1009–1020 (2017)
20. Sun, B., Wei, X., Cui, J., et al.: Social activity matching with graph neural network in event-based social networks. Int. J. Mach. Learn. Cybern. **14**(6), 1989–2005 (2023)
21. Tong, Y., She, J., Ding, B., Wang, L., Chen, L.: Online mobile micro-task allocation in spatial crowdsourcing. In: Proceedings of the 32nd IEEE International Conference on Data Engineering, pp. 49–60 (2016)
22. Tong, Y., Wang, L., Zhou, Z., Ding, B.: Flexible online task assignment in real-time spatial data. Proc. VLDB Endow. **10**(11), 1334–1345 (2017)
23. Tong, Y., Zeng, Y., Ding, B., Wang, L., Chen, L.: Two-sided online micro-task assignment in spatial crowdsourcing. IEEE Trans. Knowl. Data Eng. **5**(2), 2295–2309 (2021)
24. Wang, L., Yu, Z., Han, Q., Guo, B.: Multi-objective optimization based allocation of heterogeneous spatial crowdsourcing tasks. IEEE Trans. Mob. Comput. **17**(7), 1637–1650 (2018)
25. Wei, X., Li, Z., Liu, Y., Gao, S.: SDLSC-TA: subarea division learning based task allocation in sparse mobile crowdsensing. IEEE Trans. Emerg. Top. Comput. **9**(3), 1344–1358 (2021)
26. Wei, X., Sun, B., Cui, J., et al.: Location-and-preference joint prediction for task assignment in spatial crowdsourcing. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. **42**(3), 928–941 (2023)
27. Yang, D., Zhang, D., Zheng, V.W., Yu, Z.: Modeling user activity preference by leveraging user spatial temporal characteristics in LBSNs. IEEE Trans. Syst. Man Cybern. Syst. **45**(1), 129–142 (2015)
28. Yang, Y., Liu, W., Wang, E., Wu, J.: A prediction-based user selection framework for heterogeneous mobile crowdsensing. IEEE Trans. Mob. Comput. **18**(11), 2460–2473 (2019)
29. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**(6), 712–731 (2007)
30. Zhang, X., Yang, Z., Liu, Y., Tang, S.: On reliable task assignment for spatial crowdsourcing. IEEE Trans. Emerg. Top. Comput. **7**(1), 174–186 (2019)
31. Zhao, B., Xu, P., Shi, Y., Tong, Y.: Preference-aware task assignment in on-demand taxi dispatching: an online stable matching approach. In: Proceedings of the 33rd AAAI Conference on Artificial Intelligence, pp. 2245–2252 (2019)
32. Zhao, Y., Guo, J., Chen, X., Hao, J.: Coalition-based task assignment in spatial crowdsourcing. In: Proceedings of the 37th IEEE International Conference on Data Engineering, pp. 241–252 (2021)
33. Zhao, Y., Xia, J., Liu, G., Su, H.: Preference-aware task assignment in spatial crowdsourcing. In: Proceedings of the 33rd AAAI Conference on Artificial Intelligence, pp. 2629–2636 (2019)
34. Zhao, Y., Zheng, K., Cui, Y., Su, H.: Predictive task assignment in spatial crowdsourcing: a data-driven approach. In: 36th International Conference on Data Engineering, pp. 13–24 (2020)
35. Zhao, Y., Zheng, K., Yin, H., Liu, G.: Preference-aware task assignment in spatial crowdsourcing: from individuals to groups. IEEE Trans. Knowl. Data Eng. (2020)

36. Zheng, Y., Zhang, L., Xie, X., Ma, W.Y.: Mining interesting locations and travel sequences from GPS trajectories. In: Proceedings of the 18th International Conference on World Wide Web, pp. 79–800 (2009)
37. Zhu, X., Luo, Y., Liu, A., Tang, W.: A deep learning-based mobile crowdsensing scheme by predicting vehicle mobility. IEEE Trans. Intell. Transp. Syst. **22**(7), 4648–4659 (2021)

PPS: A Publish-Process-Subscribe Middleware for Predictive Supply Chains

Amir Jabbari[1(✉)], Gowri Ramachandran[1], Sidra Malik[2], and Raja Jurdak[1]

[1] School of Computer Science, Queensland University of Technology, Brisbane, Australia
e.amirjabbari@gmail.com, {g.ramachandran,r.jurdak}@qut.edu.au
[2] Data61, CSIRO, Eveleigh, Australia
sidra.malik@data61.csiro.au

Abstract. IoT deployments collect sensor data in supply chains, industrial monitoring, and smart environments. Raw sensor data is sent to remote dashboards or the cloud. Raw data have limited insights compared to processed data, which enables valuable insights through predictive analytics and rule-based alerts. Existing data-sharing middleware, like publish-subscribe frameworks, lacks on-the-fly data processing. This forces users to handle raw data. Built-in data processing in middleware reduces user process management overhead and provides near real-time insights. We propose a novel Publish-Process-Subscribe (PPS) middleware that provides on-the-fly data processing abilities while preserving the loose coupling benefits of the publish-subscribe paradigm. Our middleware simplifies the process management efforts for the end users by allowing them to register processes and map them to real-time data streams. Besides, PPS dynamically allocates computing resources based on the data rate and perceived processing demands. As a result, the end-users receive processed data in near real-time without manually setting up processing nodes. Experimental results show that our PPS offers significant benefits at the cost of minimal performance overhead compared to the traditional publish-subscribe middleware. We conclude that the significant reductions in replicated computations among subscribers outweigh the overheads of our framework, rendering it an attractive option for predictive supply chains.

Keywords: IoT Applications · Broker · Publish-Subscribe · Supply Chain Monitoring · Predictive Insights

1 Introduction

Current supply chains operate globally, including numerous interconnected organisations, extensive distribution, and complex transactions. There is an increasing need to improve efficiency by adopting modern technologies and logistics due to concerns about supply-demand balancing, traceable oversight,

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 125–144, 2024.
https://doi.org/10.1007/978-3-031-63992-0_8

just-in-time processes and adaptive supply chains. Emerging supply chains use sensor-based logistics in an industry-wide trend of transforming non-digital supply chains into digital ones with near real-time information covering asset conditions and locations. Sensor-based technology enables supply chains to link physical assets with their virtual counterparts, enabling automatic traceability. This traceability system uses a temporal data model to track the raw materials and their relevant events through the supply chain.

Supply chains are transitioning to proactive predictions from traditional reactive methods. Therefore, supply chains must support prediction to ensure they receive on-the-fly insights and make proactive decisions. Reactive models, which respond to events after they occur, may need to be revised in such dynamic environments. Proactive predictions, on the other hand, allow supply chain managers to anticipate future events and take preventive actions to optimise their operations [22]. Therefore, there is a need for immediate processing of the shared raw temporal data to extract predictive and analytic insights. Internet of Things (IoT) devices publish this temporal data as messages through data sharing middlewares following either a publish-subscribe or request-reply messaging model [19,23,25]. In the supply chain context where many IoT devices publish many messages simultaneously, the Publish-Subscribe messaging model is widely utilised and well established due to their efficiency in low communication overheads and support of loosely coupled many-to-many messaging [25,30]. In this model, message publishers publish messages under topics, and subscribers must subscribe to topics to receive updates. Unlike Publish-Subscribe, the Request-Reply messaging model is more suitable for resource-constrained systems as there must be a request per message to get replies [23]. However, no matter the size of the system, none of these messaging models offers built-in support for processing published messages. Moreover, existing cloud or edge-based data sharing applications [6,15,18] that are developed on top of the mentioned messaging models offer data processing as an external feature. To gain insights, the user must manually activate data processing features in a reactive manner. Therefore, existing applications also lack automated processes that help predictive supply chains to gain insights proactively.

Shared data within the Publish-Process-Subscribe (PPS) middleware must be processed to extract insights in a near-real-time fashion [7]. Insights are key components for predictive supply chains, and a series of automated actions are required to gain insights from published messages and gathered data. In a supply chain, many IoT-based sensors produce data simultaneously, and existing technologies can generate analysis on each of them separately by adopting edge computing [20]. However, the shared data of multiple sensors carry insights when combined and processed for the fusion of the computations. These computations must be executed immediately to gain near-real-time insights. Therefore, there is a need for computational infrastructure that supports a vast number of processes. However, the lack of data processing within the data-sharing middleware makes the message subscribers hire or invest in computing units that can execute complex algorithms on shared data [9]. Due to the high expenses of building com-

putational infrastructure and the complexity of computations required to gain predictive and analytic insights, most supply chain nodes tend to offload these computations to other computing nodes, such as third-party cloud computing platforms. However, offloading computations to another computation provider increases inefficiencies, operational costs, and delays. Also, different subscribers must replicate the execution of the same substantial computations on the same shared data to extract insights individually.

To address the above challenges, we propose PPS, a data sharing and processing middleware that provides automated processing features within the messaging model and delivers insights instead of raw data. We design a data-sharing and processing middleware for publish-subscribe systems by executing computations on data to generate insights to allow the data consumers (or subscribers) to receive processed insights in near real-time from the data broker. Our middleware automates data processing by dynamically scheduling processing tasks on computing units based on the perceived load. The contributions of this paper are:

- We design a built-in data processing feature within the publish-subscribe data-sharing middleware where published raw data is processed on the fly at the PPSManager (so-called broker in traditional publish-subscribe architecture).
- We develop a load-balancing algorithm to dynamically allocate and deallocate computing nodes to prevent bottlenecks at PPSManager.
- We evaluate PPS using a proof-of-concept through a simulated supply chain use case. Our evaluation results indicate that PPS significantly reduces the total number of computations needed to gain insights while incurring minor increases in broker's CPU and Memory usage.

The rest of the paper is organised as follows: Sect. 2 introduces the existing messaging middleware and discusses related works. Section 3 presents the detailed architecture definition for our proposed Publish-Process-Subscribe (PPS) framework, and Sect. 4 presents the system description. Section 5 presents the system's safety and liveness analysis, and Sect. 6 discusses the proposed PPS implementation and its evaluations. The last section concludes the paper with future work.

2 State of the Art and Related Work

There are a variety of platforms and studies conveying data processing opportunities in the IoT application context. Many rely on third-party cloud providers like Amazon [1] to ease the development of IoT applications and set up a serverless data-sharing environment. Using serverless computing, application developers delegate computational maintenance and concerns to cloud vendors [4]. However, data processing features in these serverless platforms are offered as extensions that must be activated and managed by clients after the shared data is received. Figure 1 shows different attitudes towards the raw data in data sharing middlewares. Assuming publish-subscribe as the default messaging model

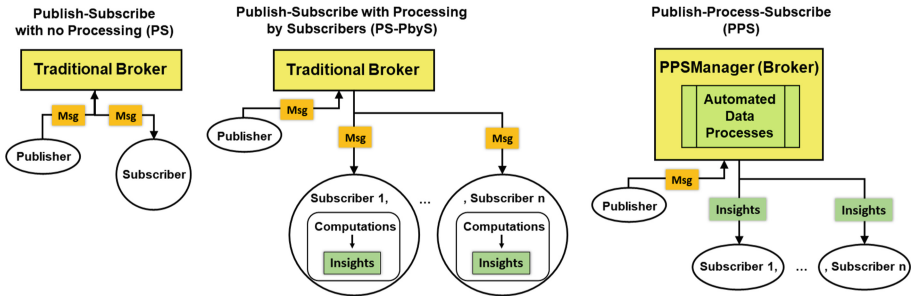


Fig. 1. Evaluation candidates: Traditional Publish-Subscribe (left), Publish-subscribe with Processing at Subscribers (centre), and PPS (right).

in this figure, all the existing studies and platforms forward data processing to the subscribers' side in a Publish-Subscribe with Processing by Subscribers (PS-PbyS) fashion. In serverless platforms, vendors act as subscribers. After receiving and storing, and collecting the shared data, vendors' computing units perform pre-defined processes on data based on the pre-defined settings [18]. However, in this reactive manner of processing shared data, the delay costs of gaining insights cause inefficiencies for predictive purposes. Furthermore, supply chain stakeholders, namely publishers and subscribers, must pay for these computational services.

Produced and shared data of IoT-based devices are valuable datasets for many applications. Machine learning models are trained and offered as paid services using these datasets. These models are used in applications like Digital Twin, where IoT and Artificial Intelligence are combined to interpret and visualise data [16]. Using Digital Twin, manufactured products or the whole IoT system's physical aspects are connected to the virtual prototypes to assess, validate and verify the system operation [24]. Raw datasets can be traded with computational infrastructure once the IoT data marketplace feeds the Digital Twin's input. Typical IoT data marketplaces' target is only to connect data buyers to sellers, and they lack data processing support [12,14,26,28]. However, the IoT data marketplace has the potential to offer data buyers like application developers the opportunity of exchanging computational infrastructure for collecting data with data sellers like supply chain stakeholders. In [20], the authors propose machine learning and edge computing support in the IoT data marketplace. Therefore, these computations can be executed at the edge or cloud and provide analytical and predictive insights based on the trained machine learning models on the gathered data in near-real-time. However, existing data marketplaces also lack an entity that facilitates and schedules these computations before data reaches the computing units. Therefore, they have the same architecture as the PS-PbyS approach in Fig. 1, where the shared data is delivered to the subscribers first, and computations are executed after.

Processing of shared data is one of the significant concerns in the decentralised applications of the Internet of Things (IoT) that require messaging protocols. The publish-subscribe messaging paradigm is frequently used in modern IoT and business deployments because of its resource efficiency and support for loosely-coupled messaging. In this paradigm, subscribers register their interest in an event, or a pattern of occurrences, in systems based on the publish-subscribe interaction and are then asynchronously alerted of events created by publishers. Many variations of the publish-subscribe paradigm have been presented, each tailored to a unique application or network architecture [8]. The common denominator underlying these versions is the complete decoupling of the communication entities in time, space, and synchronisation.

Publish-subscribe capabilities can adopt either the network and application layer or the content-based protocols [15]. In the first category, network layer protocols rely on the broker as an entity that runs on a server, like MQTT (Message Queuing Telemetry Transport) [25] and AMQP (Advanced Message Queuing Protocol) [11]. The broker receives the messages and routes them to the subscriber. On the other hand, content-based protocols are designed to run without the central broker. Therefore, they move overhead from the broker to the network, slowing down the network traffic when the throughput grows. Even though the content-based publish-subscribe models perform well under hardware limitations, they still need to sacrifice parameters like delivery nodes to decrease workload traffic and their performance compared to broker-based systems [15]. For instance, in [6], authors have proposed architecture in Edge-to-Cloud Continuum for adaptive data-driven routing on top of the CUPUS [3], their former content-based publish-subscribe implementation. However, to increase the performance metrics like less delay in data delivery, they limit the number of subscribers and prioritise them as top-K based on the producers' requirements. This reduced latency results from reduced delivery nodes and communication traffic and does not guarantee message delivery to all subscribers. Broker-based network layers rely on a central entity called the broker that can be integrated with cloud-based systems, like Kafka [17,29]. In Kafka's architecture, messages are divided by topics and must be published in different disc partitions for each topic. Each publisher chooses the topic and the partitions to publish to, then writes the messages in a round-robin method to the partitions. Subscribers should read messages from the partitions, and the number of subscribers to a topic cannot be more than the number of partitions. Otherwise, they might be unable to read the messages and become inactive [15]. Also, each read generates an offset commit request, which adds to the broker's workload. Workload increase in Kafka leads the system to ignore real-time data sharing and limit the subscriber's message consumption [21]. Kafka, on its own, lacks adequate support for stateful data processing. Therefore, external frameworks like Flink [5] must handle processes. However, setting up Flink data stream processing as a Kafka consumer object does not imply automated processing as there is a need for controlling and triggering the executions by a developer since Flink is lazy loaded [27]. Therefore, there is no guarantee that all subscribers in Kafka can benefit from the raw

shared data simultaneously and receive insights through automated processes in Flink. Furthermore, Flink's resource management is demanding, especially with varying data arrival rates. Therefore, dynamic resource allocation support is still challenging in systems that use this framework for stream processing [31]. On the other hand, Flink's development and operational complexity require a professional back-end team in supply chains to manage and control the system, which might be against the incentives of supply chains' stakeholders that lack a suitable computational infrastructure.

As shown in Fig. 1, existing data sharing middlewares and IoT-based applications lack a processing stage that automates data processing, task scheduling and computing node allocations. Also, the edge or cloud-based platforms lack delivering predictive/analytic insights from the shared data to subscribers, and they are the subscribers who reactively execute computations. These computations' operational costs are handed to each subscriber individually, and providing the processing requirements like resource allocations, and bottleneck prevention is all on the client's end. We propose PPS middleware, where data processing features are automated within the data sharing middleware, and insights are delivered to subscribers instead of raw data. The broker of this architecture, PPSManager, dynamically allocates and deallocates resources and maintains load balancing on the server's side. In the next section, we provide a detailed discussion regarding the architecture of PPS.

3 Architecture Definitions and Assumptions

In this section, the architecture of our proposed PPS middleware is discussed and definitions and assumptions are provided to describe the system.

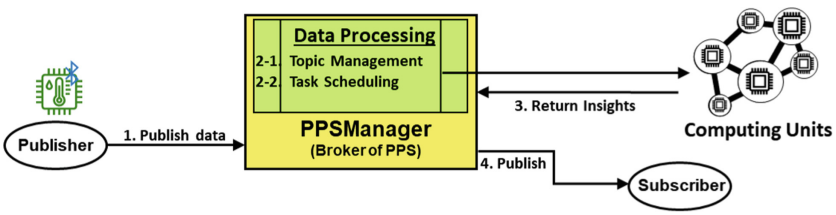


Fig. 2. Overview of the framework

3.1 Overview of Architecture

Figure 2 provides an overview of the steps to share and process published data within our proposed PPS architecture. Unlike the traditional broker-based publish-subscribe frameworks like Kafka [21], messages in our proposed architecture are published to a processing manager called PPSManager instead of

the broker. Besides satisfying all the services of a traditional broker, this processing manager facilitates the data processing requirements before sharing the outcome of processing (insights) with the subscribers. The key components of our proposed PPS data sharing and processing middleware indicated in Fig. 2 are defined below, followed by their interactions descriptions:

Publishers: are the IoT-based sensors, organisations and anyone who provides updates and information published as messages with a standard structure under specific topics. Publishers must register with the system, and this registration is also shared as a message within PPS middleware. All messages have topics that are available to other stakeholders within the application ecosystem, and interested subscribers can subscribe to them.
Topic: is descriptive metadata in a string format that can come in multiple levels. For example, topic "TTISensor/001" means the data is related to a temperature sensor with the sensor ID "001".
Subscribers: are nodes that subscribe to topics and receive all messages published under those topics. Similar to publishers, subscribers must also register with the system, and this registration is shared as a message within PPS middleware.
PPSManager: This trusted entity works as a broker in PPS architecture, with additional data processing responsibilities to manage topics. In a supply chain, predicting the expiry date of a product based on its current state would be an example of data processing. In this case, whenever a supply chain entity such as a warehouse or carrier publishes the data, the PPSManager receives the data and executes the"predictExpiryDate()" process and relays the outcome to the relevant subscribers. Using topics, PPSManager identifies, schedules, and allocates tasks on a random computing unit for further computations and insights extraction.
Computing units: are the participants registered to provide computational services to the framework. These computers register to the network. PPSManager maintains and manages the registered computers for further task allocations.
Insights: are predictive and analytic insights gained from executed computations on the shared raw data.

3.2 Assumptions

There are also key assumptions in the proposed PPS middleware architecture that are described below: **Assumption 1:** A correct PPSManager (broker) is a trusted entity that is always online. **Assumption 2:** A correct publisher always publishes the messages following the topic and the protocol structure recommended by the PPS. **Assumption 3:** PPS manages a sufficient number of computing units to meet the processing demands of publishers and subscribers. For simplicity, we assume the computing units belong to PPSManager. **Assumption 4:** Correct publishers trust the PPSManager to process the raw data and share the insights with the subscribers. **Assumption 5:** The computational tasks do not include malicious code, including malware. **Assumption 6:** The

computational tasks are terminated within a bounded delay. **Assumption 7:** All the communication between PPS, publishers, subscribers, and computational units are secured. All messages published by the correct publishers are delivered exactly the same as the messages received by the PPSManager. **Assumption 8:** If a subscriber receives a processed message, all the other interested subscribers must receive the same processed message within a bounded delay. **Assumption 9:** The order of the messages is preserved. Therefore, if message m is received before message m' by a correct subscriber, all subscribers will receive the same messages in the same order. **Assumption 10:** PPS do not charge the publishers and subscribers for the computation. We will investigate a pricing model for PPS in our future work. **Assumption 11:** We assume the maximum computation time for each computation. This can be calculated by running the computations repeatedly for n number of times at the PPSManager.

3.3 System Model

We consider a system with a broker $\mathcal{B}$ and a set of publishers $\mathcal{P}$ and subscribers $\mathcal{S}$. Each subscriber $s_m \epsilon \mathcal{S}$ needs certain processed data d_x for its application. To get d_x, the subscriber, s_m, must receive raw data, r_x from broker $\mathcal{B}$ and execute a computation c_x to receive the processed data d_x. Existing publish-susbcribe systems only allow the subscriber to receive r_x without any processing support for generating d_x on the fly. Allowing the broker to process r_x using the computation c_x, the subscriber can directly receive the processed data d_x without any processing locally. Traditional publish-subscribe systems allow subscribers to procure computation resources $\mathcal{R}$ to run computations. Assuming the subscriber is interested in receiving several processed data d_1, d_2, ..., d_x, ... d_n, it has to have sufficient computing resources to process all the data. Offloading the processing to the broker would reduce the compute management process for the subscriber while still providing access to the processed data. To that end, our work extends $\mathcal{B}$ with support for computations through a set of managed computing resources, $\mathcal{R}$.

In traditional publish-subscribe, each subscriber has to run the processing on shared r_x locally to extract d_x. This study calls this reactive manner of processing raw data PS-PbyS as Publish-Subscribe, with Processing by Subscribers. If the system has m subscribers, then the total number of computations needed to convert r_x to d_x for all the subscribers is m*c_x. Our proposed publish-process-subscribe processes c_x once per data item and directly shares d_x without needing local computations at the subscriber's end. As the complexity of c_x increases, our approach significantly saves the resource consumption of subscribers. Figure 3 compares our proposed PPS system and PS-PbyS approach regarding the number of computations executed to get processed data d_x as insights. As the message arrival rate increases, the total number of computations to obtain d_x differ significantly.

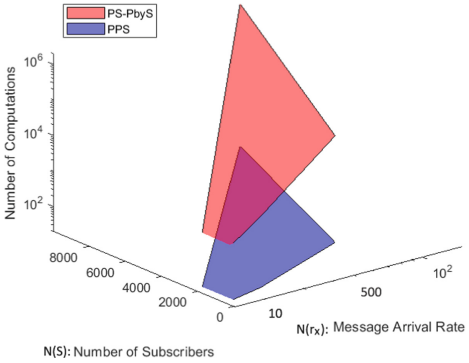


Fig. 3. Number of Computations Comparison in PPS and PS-PbyS

3.4 Existing and Newly Introduced Control Packets

Control packets are the exchanged packets between the server and clients. In this section, we introduce new control packets for the proposed PPS architecture to augment existing MQTT control packets [8].

Table 1. Existing Control Packets (✓) and Proposed Control Packets (⊕)

Control Packet	Description	Status
CONNECT	Connection request	✓
PUBLISH	Publish message	✓
SUBSCRIBE	Subscribe request	✓
UNSUBSCRIBE	Unsubscribe request	✓
ADVERTISE	Advertise topics	⊕
DISCOVER	Discover topics	⊕
QUERY	Query registered computing units	⊕

Using these control packets, publishers can ADVERTISE topics they use to publish messages, sell the data published under those topics in the data marketplace and help subscribers to DISCOVER these topics. Besides subscribers, data buyers can also DISCOVER topics and connect to sellers through PPSManager. QUERY control packets help the PPSManager to query a list of registered computing units for task scheduling and allocations.

Based on the mentioned assumptions and introduced control packets, the system description is provided below.

4 Architecture of PPS Middleware

Publishers and subscribers in this architecture must connect to the PPSManager the same as they connected to the broker in MQTT [25]. They use the same APIs to Publish and Subscribe to different messages, and Subscribers can Discover topics to investigate their interest in subscribing to new shared data. Publishers can Advertise topics of their publicly published messages, and buyers in the Data Marketplace can aim to trade their computational infrastructure with data [20]. As mentioned in assumption 8, computing units belong to the PPSManager and are registered separately to the networks based on their availability. There is a query of the registered computing units at PPSManager with connection details like their IP addresses. PPSManager has a wildcard subscription to all topics. Therefore, it receives all messages, investigates published messages by their topics, and allocates computation-related messages to these computing units. These messages include temporal data that is needed for insights extraction. PPSManager investigates these topics and allocates them as a task with an updated topic to the computing units. Computing units are picked randomly by PPSManager, and the required functions and algorithms are predefined on the computing units. Based on the topic of the allocated tasks, they automatically execute related functions and algorithms. The insights are published under the Advertised topics to subscribers, who can subscribe to the topics by discovering them using the proposed control packets in Table 1. Computations are bound to a deadline, and computing units must execute these computations before the deadline [13].

Messages in our proposed PPS are published under topics, and each topic can trigger specific functions to be executed on the information. For instance, in a supply chain, a distribution truck unloads products at different places. Multiple IoT sensors in the truck publish temporal messages regarding the container's temperature, current location and speed. Message publishers in this example are the Time and Temperature (TTI), Relative Humidity (RH), and Global Positioning systems (GPS) sensors. TTI sensors publish temperature-related messages and trigger functions required for calculating the remaining shelf life of a perishable product and spoilage rate prediction algorithms based on the shared temperature values. Latitude and Longitude details of the truck's current location published by Global Positioning systems (GPS) sensors trigger functions related to calculating the travelled distance and the algorithms for finding the optimal and nearest destination to unload the products. Executing these computations requires a computational infrastructure capable of running complex computations.

Shared data published by each publisher provides information regarding that sensor only. For instance, temperature sensors include temperature-related values only, and if computations need to extract insights from these values, they must combine them with other published messages. Therefore, if computations are executed on a single publisher's data, they provide limited analysis of that sensor. However, to extract insights, there is a need to execute computations on the fusion of data from different publishers. Therefore, the processing must

happen within the middleware, and shared data must be combined to result in insights. As existing data-sharing middlewares lack a processing layer within their messaging models, subscribers must collect and process shared data individually. Executing computations after raw shared data are received by subscribers results in insight in a reactive manner. In the mentioned existing data sharing middlewares that lack built-in data processing support, subscribers must collect, analyse and process data separately. Therefore, considering n subscribers subscribing to the same topic in a publish-subscribe middleware, subscribers execute n same processes on the same shared message separately. On the other hand, if shared data like the temporal published messages by sensors are not processed instantly, the computation results are useless for predictive supply chains. Our Publish-Process-Subscribe (PPS) data sharing and processing middleware addresses this issue with a built-in processing layer where shared data are processed automatically within the messaging middleware. PPS trades off insights and reduced computation replication with a bounded delay for predictive supply chains.

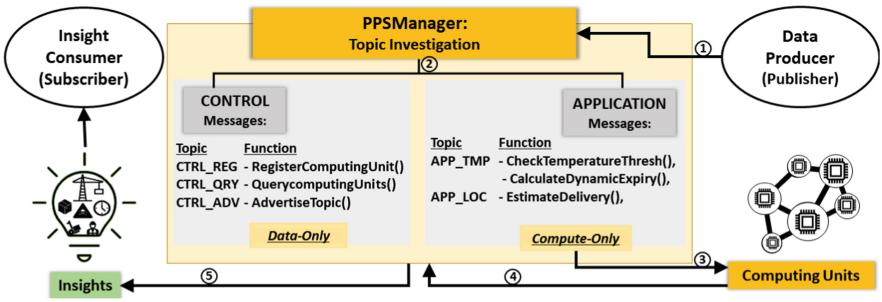


Fig. 4. Message Topic Investigation

4.1 Topic Management

PPSManager in PPS middleware architecture provides data processing facilities. Therefore, PPSManager must investigate topics to distinguish messages that carry temporal data as data extraction materials from messages that include general information like a new node registration in the system. As indicated in Fig. 4, we categorise messages into two separate categories, the Control Messages and the Application Messages.

Control Messages are Data-only messages representing statistical information published to the network to make an announcement. These messages are published to the PPSManager for certain actions like registering a computing node or a new subscriber. There is no need to execute complex computations to extract insights from these messages, so they will be forwarded directly to their topic subscriber after running the required actions.

On the other hand, *Application Messages* are Compute-only messages, indicating that these messages include vital information that must be executed to gain insights. After investigating the topic of each message, PPSManager treats these messages as tasks and allocates them to random computing units for insights extraction. Each topic triggers specific functions and algorithms for execution by computing units. For instance, as shown in Fig. 4, messages published under APP_TMP show PPSManager that they must be treated as tasks, and the required functions to be executed on them are to check the temperature threshold and calculate the dynamic expiry date. Each Data-Only message includes the payload section, following the JSON message format and JSON payload structure. Payloads of each message contain the required values for computations, and tasks are the required computations that must be executed on these values.

4.2 Task Scheduling and Load Balancing

Each message's topics help PPSManager determine whether this message needs computations or must be forwarded to its topic subscribers directly. Messages that include temporal data in their payload are allocated as tasks. These tasks are scheduled instantly as jobs that need to be processed, and the priority for tasks to be allocated is first in, first served. In our proposed task scheduling algorithm for PPS, every time a new message arrives, PPSManager queries registered computing units to list available computing units and allocates tasks to them in a round-robin fashion. Task scheduling is done in the message arrival order, and allocations on the computing units start with the first available computing unit. Equation 1 determines the required number of computing units (N(C.U.)) for executing tasks. In this Equation, C_{Dem} and C_{Cap} are Computation Demand and Computation Capacity. Computation Demand(C_{Dem}) is calculated with R, the message arrival rate indicating the number of messages received per Second, multiplied by the number of functions each topic needs to compute, N_f. Computation Capacity C_{Cap} is also the number of computations that the system can handle per second.

$$N(C.U.) = C_{Dem}/C_{Cap} \tag{1}$$

$$C_{Dem} = R * N_f \tag{2}$$

In [13], the authors implemented an algorithm for task scheduling where computing units are scheduled based on the deadline. The initial allocations must satisfy the required number of computing units, and once the deadline is at risk, a new computing unit is injected to assist with the computations. Likewise, in our PPS middleware, as allocations are random and instant, the first random set of computing units that can execute all functions of computations within the bounded time is chosen for the job. In this best-fit-first method, resources must execute computations in a time-bounded manner. The maximum execution time

in demand capacity derived from the Eq. 2, ensures we avoid under-provisioning the initial number of resources. We use ET_{max} as a restriction to guarantee that the service rate is faster than the data arrival rate.

Algorithm 1: Task Scheduling and Load Balancing.

Input: **D** ← Deadline,
M ← A set of Registered Computing Units,
T_{int} ← Sensors' Delay Interval per Message,
Result: Allocate Computing Units

```
calcluate R: Rate of Message Arrival using Sensors' Delay Interval T_int;
calcluate Total Computing Units Needed using the Equation 1 ;
for <each i ∈ Tasks> do
    for <each j ∈ Total Computing Units Needed> do
        Allocate M[j] to execute Task[i];
        for Duration of T_int Publish Message Interval do
            Idle M[j];
        end for
        if No result for Task[i] in D then
            Allocate Task[i] to M[j+1];
            Terminate M[j];
        end if
    end for
end for
return the Set of Allocated Computing Units
```

5 Safety and Liveness Analysis

While the liveness property implicitly ensures that the system will advance and that "something positive will always happen", the safety property establishes that "something wrong will never happen" [2]. In this part, we demonstrate our proposed PPS's liveliness and safety qualities.

5.1 Safety Properties

Publisher's safety: A correct PPSManager can guarantee the published messages are delivered to topic subscribers. **Messages' safety:** All correct subscribers receive the same messages, including the same information and computational results. Also, all the correct subscribers receive the messages in the same order it has been delivered to all other subscribers. **PPSManager's safety:** All correct computing units are online. PPSManager can Query the registered computing units and allocate tasks on them, which must be executed within a bounded delay. If a task is not completed by a deadline, that task is allocated to another computing unit immediately.

There are always enough computing units available for executing computations. The minimum number of available computing units ($minN_{CU}$) for task allocations, resulting from a QUERY control packet by the PPSManager must match this Eq. 3:

$$minN_{CU} = n(T_{adv} \cup T_{discov}) * C_{Dem} \tag{3}$$

Where T_{adv} is the set of Advertised topics by publishers and T_{discov} is the set of Discovered topics by subscribers. Computation Demand (C_{Dem}) also results from Eq. 2.

5.2 Liveness Properties

Messages' liveness: Messages published correctly by the correct publishers will be delivered to the message's topic subscribers within a bounded delay. **Computing units' liveness:** Correct computing units are always online, and in each Query, there are at least an equal number of computing units with the number of topics. **Computations Liveness:** Computations are bounded by a deadline, and if computing units fail to execute computations or go offline, computations are allocated to another computing unit.

6 Implementation and Evaluation

This section discusses the implementation and evaluation of the proposed Publish-Process-Subscribe (PPS) middleware.

6.1 Experimental Setup

We implemented the PPS framework on a local testbed using Mosquitto, an open-source message broker that implements the MQTT protocol Broker. MQTT is one of the widely used publish-subscribe communication protocols in IoT applications. The broker-specific functionalities are implemented on top of MQTT, called PPS-Managers in our middleware. To obtain a real-world experiment, we used Teensy 2.0 development boards, DHT11 sensors to produce temporal temperature and humidity data, and HC-05 Bluetooth transceivers to communicate with a Raspberry Pi to publish data. An Amazon EC2 instance was also set up to work as the PPSManager, and the computations were allocated on an Apple M2 chip with an 8-core CPU and 10-core GPU with 100GB/s memory bandwidth for both PPS and PSPbyS scenario. Application-specific software were implemented in Arduino, Python and NodeJS.

6.2 Use Case Application of PPS Data Sharing and Processing Middleware

To justify the usability of our proposed PPS platform, we rely on a real-life need for data processing within a data-sharing middleware where the remaining life of perishable products in a supply chain distribution must be calculated. As mentioned in Sect. 2, [10] proposed a Decision Support System (DSS) to efficiently manage and prevent perishable loss by incorporating established functions for food quality assessment in dynamic settings. They used this equation:

$$KQ = \frac{Q_0 - Q_L}{k}$$

to calculate the remaining life of a perishable, where the Q_0 presents the current quality of the product and Q_L is the quality limit. K is the spoilage rate which is calculated as:

$$k = k_{ref} * exp^{\frac{EA}{R} * \frac{1}{T_{ref}} - \frac{1}{T}}$$

in which T is the given temperature and k_{ref} indicates the spoilage rate at a reference temperature T_{ref}. EA and R are also the activation energy and gas constant [10].

In our scenario, a truck distributes perishable products to different stores. There are sensors in the truck's container that publish temporal temperature data and GPS sensors to publish the truck's current location. Published temperature values are used in the mentioned equations to calculate the remaining life of perishables. Location parameters, namely latitude and longitude values, are also published to indicate the updated location of the distribution truck. Messages, including temperature and location data, are published as Application messages and trigger temperature and location-related functions. For instance, the temperature threshold is assessed (by CheckTemperatureThresh() function), and the dynamic expiry date (by CalculateDynamicExpiry() function) is calculated automatically using the payload values and mentioned equations. However, if the shared data gets processed within the proposed PPS data-sharing and processing middleware, it can benefit a proactive supply chain with near-real-time insights. Otherwise, in a reactive method like the MQTT, where computations are pushed at the subscribers' end, computational results only help supply chain stakeholders perform reactive actions.

6.3 Performance Evaluation

In this section, experimental results and evaluations are discussed. We conducted experiments to evaluate our PPS middleware performance and to compare it with the existing PS-PbyS approach. To compare and contrast PPS and PS-PbyS approaches in a similar environment, we evaluated an ideal setup for the PS-PbyS approach where the subscriber has a powerful computational infrastructure and executes computation immediately after messages are delivered.

In these evaluations, based on the perishable products use case Sect. 6.2, DHT11 sensors are used to generate temporal temperature and humidity values, and using a Raspberry Pi as a publisher, messages with a payload including these values are published. These messages are published under a specific "Temperature" topic, and this topic triggers related functions to execute. Temperature and humidity-related functions include the calculation of perishables' dynamic expiry and the remaining life, as well as the Moving Average function to predict the next expected temperature and the remaining life based on the previously published messages. The minimum execution time of the mentioned functions is 251 Milliseconds after more than 10000 runs, and the number of messages sent for each message arrival rate has been 5000 messages sent to the broker (PPSManager in PPS) per each arrival rate run.

Figures 5 and 6 indicate the end-to-end delay of extracting insights by executing computations in both PS-PbyS approaches and our proposed PPS middleware. Due to the high latency of the messages being executed in the PS-PbyS approach and to better illustrate the distinctions, we plot the end-to-end delay in the logarithmic scale.

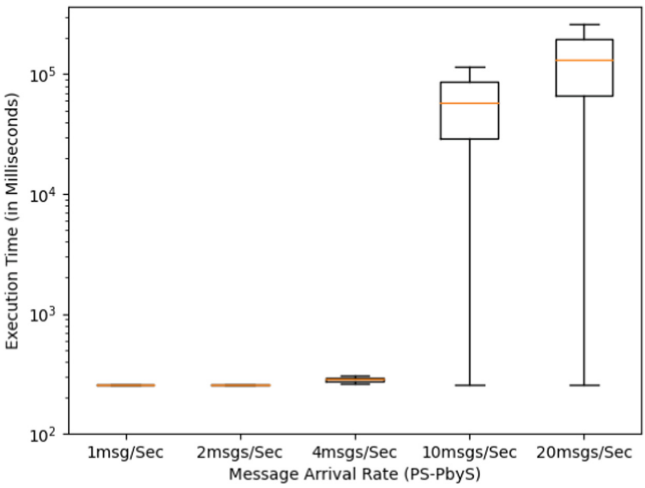


Fig. 5. The End-to-End Delay in PS-PbyS

While the message arrival rate is low, the end-to-end delay in PS-PbyS fashion and the PPS has the same trend with the minimum execution time fluctuations in milliseconds, within the one message per Second to the four messages per Second arrival rate. However, when the number of published messages increases, there is a significant change in the total execution time for 10 and 20 messages published per Second as the service rate falls behind the message arrival rate. This drastic change in execution time occurs when the load increases and more messages arrive at the subscribers' end. This enormous rise in delay for 10 and 20 messages published per Second leads to significant changes in the graph where the lower message arrival rates (1, 2 and 4 msgs/Sec in Fig. 5) seem flattened. In this experiment, one publisher produces the raw data and the subscriber in the PS-PbyS approach receives and processes the raw data. This subscriber then executes the same computations and functions that are evaluated in PPS to extract the required insights. However, as the PS-PbyS approach lacks load balancing, an inevitable bottleneck occurs as the message arrival rate increases.

In PPS, this bottleneck is avoided by obtaining the resource allocations in Algorithm 1, where the PPSManager first subtracts the required number of computing units using Eq. 3 from the list of registered computing units by the QUERY control packet in Table 1. As mentioned earlier in Sect. 4.2, this resource allocation follows a round-robin fashion and allocates tasks efficiently to the

minimum number of resources available. PPSManager uses this algorithm to choose a sufficient number of resources from the list of registered computing units, and after processing the topics Sect. 4.1, it allocates each message to an available computing unit and activates them to execute the computations. The number of activated computing units for task allocations is mentioned in Table 2. This Table 2 shows the results of dynamic and efficient resource allocations as the message arrival rate fluctuates.

Table 2. Computing Unit (CU) Distribution for Various Message Arrival Rate in PPS

Arrival Rate	Number of Activated Computing Units	Activated Computing Units
1msg/s	1	[CU1]
2msgs/s	1	[CU1]
4msgs/s	3	[CU1, CU2, CU3]
10msgs/s	5	[CU1, CU2, CU3, CU4, CU5]
20msgs/s	10	[CU1, CU2, ..., CU10]

Figure 6 shows the logarithmic scale of end-to-end delays in PPS. As the PPS handles the different message arrival rates by allocating the needed number of resources when a message is published and an event occurs, the end-to-end delay remains stable thanks to the PPSManager's load balancing. The number of activated computing units for each message arrival rate is also mentioned in Table 2, and the number of activated computing units to execute the tasks is the minimum and efficient number of computing units vital to prevent system bottlenecks. This minimum number of required computing units is derived from the earlier discussed in Eq. 1.

There are also some inevitable rises in performance while using PPSManager instead of a plain broker. These changes are investigated in Table 3, and as the results indicate, CPU and Memory usage increase once we add data processing features on top of the PPSManager. However, the network's bandwidth decreases significantly on the subscribers' end as the computations are executed before delivering insights. Therefore, by handling processes automatically, PPS middleware decreases throughput from the subscribers' end.

Table 3. Broker's Performance in PS-PbyS and PPS for 20msgs/s.

Data Processing Model	Average CPU Usage	Average Memory Usage	Broker's Network Bandwidth	Subs' Network Bandwidth
PS-PbyS	5.2%	0.1%	922.35 kbit/s	996.21 kbit/s
PPS	48.9%	1.5%	952.45 kbit/s	289.93 kbit/s

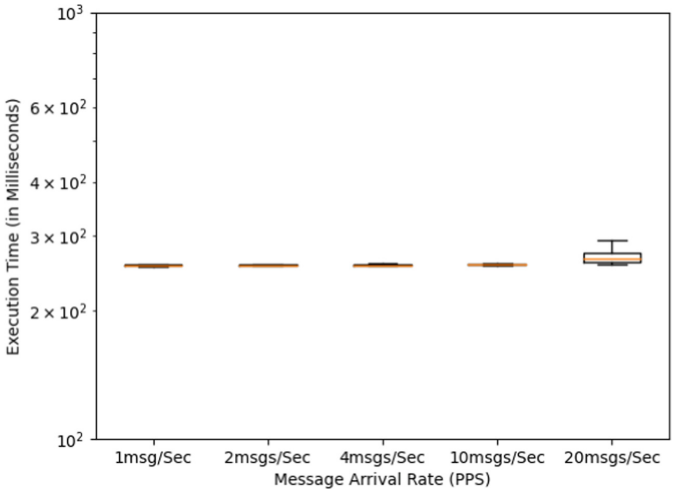


Fig. 6. The End-to-End Delay in PPS

7 Conclusion

We proposed a Publish-Process-Subscribe framework architecture, in which the processes are automated, and predictive and analytic computations are executed rigorously to provide insights in favour of modern predictive supply chains. An interesting direction for future work is to decentralise the system and to guarantee the trustworthiness of processes and computations that are executed in our processing layer. Another open requirement is privacy-preserving techniques in which we guarantee stakeholders that the data are only shared with whom it is meant to be and the computations and computational results do not risk privacy concerns of the data or data producer.

Acknowledgement. This work was funded and supported by iMOVE Australia Cooperative Research Centre (CRC).

References

1. Amazon Web Services. https://aws.amazon.com/
2. Alpern, B., Schneider, F.B.: Recognizing safety and liveness. Distrib. Comput. **2**(3), 117–126 (1987)
3. Antonić, A., Marjanović, M., Pripužić, K., Žarko, I.P.: A mobile crowd sensing ecosystem enabled by CUPUS: cloud-based publish/subscribe middleware for the Internet of Things. Futur. Gener. Comput. Syst. **56**, 607–622 (2016)
4. Baldini, I., et al.: Serverless computing: current trends and open problems. In: Chaudhary, S., Somani, G., Buyya, R. (eds.) Research Advances in Cloud Computing, pp. 1–20. Springer, Singapore (2017). https://doi.org/10.1007/978-981-10-5026-8_1

5. Carbone, P., Katsifodimos, A., Ewen, S., Markl, V., Haridi, S., Tzoumas, K.: Apache flink: stream and batch processing in a single engine. Bull. Tech. Committee Data Eng. **38**(4) (2015)
6. Čilić, I., Žarko, I.P.: Adaptive data-driven routing for edge-to-cloud continuum: a content-based publish/subscribe approach. In: González-Vidal, A., Mohamed Abdelgawad, A., Sabir, E., Ziegler, S., Ladid, L. (eds.) GIoTS 2022. LNCS, vol. 13533, pp. 29–42. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-20936-9_3
7. Cui, L., Gao, M., Dai, J., Mou, J.: Improving supply chain collaboration through operational excellence approaches: an IoT perspective. Ind. Manag. Data Syst. **122**(3), 565–591 (2022)
8. Eugster, P.T., Felber, P.A., Guerraoui, R., Kermarrec, A.M.: The many faces of publish/subscribe. ACM Comput. Surv. (CSUR) **35**(2), 114–131 (2003)
9. Fahmideh, M., Zowghi, D.: An exploration of IoT platform development. Inf. Syst. **87**, 101409 (2020)
10. Fikar, C.: A decision support system to investigate food losses in e-grocery deliveries. Comput. Ind. Eng. **117**, 282–290 (2018)
11. Garcia, C.A., Naranjo, J.E., Garcia, M.V.: Analysis of AMQP for industrial Internet of Things based on low-cost automation. In: Iano, Y., Arthur, R., Saotome, O., Kemper, G., Padilha França, R. (eds.) BTSym 2019. SIST, vol. 201, pp. 235–244. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-57548-9_22
12. IOTA-Foundation: IOTA Data Marketplace. https://data.iota.org/#/demo/#list
13. Jabbari, A., Masoumiyan, F., Hu, S., Tang, M., Tian, Y.C.: A cost-efficient resource provisioning and scheduling approach for deadline-sensitive mapreduce computations in cloud environment. In: 2021 IEEE 14th International Conference on Cloud Computing (CLOUD), pp. 600–608. IEEE (2021)
14. Krishnamachari, B., Power, J., Kim, S.H., Shahabi, C.: I3: An IoT marketplace for smart communities. In: Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services, pp. 498–499 (2018)
15. Lazidis, A., Tsakos, K., Petrakis, E.G.: Publish-subscribe approaches for the IoT and the cloud: functional and performance evaluation of open-source systems. Internet Things **19**, 100538 (2022)
16. Minerva, R., Lee, G.M., Crespi, N.: Digital twin in the IoT context: a survey on technical features, scenarios, and architectural models. Proc. IEEE **108**(10), 1785–1824 (2020)
17. Narkhede, N., Shapira, G., Palino, T.: Kafka: The Definitive Guide: Real-Time Data and Stream Processing at Scale. O'Reilly Media, Inc. (2017)
18. Nasirifard, P., Jacobsen, H.A.: A serverless publish/subscribe system. arXiv preprint arXiv:2210.07897 (2022)
19. Nelson, B.J.: Remote Procedure Call. Carnegie Mellon University (1981)
20. Sajan, K.K., Ramachandran, G.S., Krishnamachari, B.: Enhancing support for machine learning and edge computing on an IoT data marketplace. In: Proceedings of the First International Workshop on Challenges in Artificial Intelligence and Machine Learning for Internet of Things, pp. 19–24 (2019)
21. Scott, D., Gamov, V., Klein, D.: Kafka in Action. Simon and Schuster (2022)
22. Seyedan, M., Mafakheri, F.: Predictive big data analytics for supply chain demand forecasting: methods, applications, and research opportunities. J. Big Data **7**(1), 1–22 (2020)
23. Shelby, Z., Hartke, K., Bormann, C.: The constrained application protocol (CoAP) (2014)

24. Sleuters, J., Li, Y., Verriet, J., Velikova, M., Doornbos, R.: A digital twin method for automated behavior analysis of large-scale distributed IoT systems. In: 2019 14th Annual Conference System of Systems Engineering (SoSE), pp. 7–12. IEEE (2019)
25. OASIS Standard: MQTT version 3.1.1 (2014).http://docsoasis-open.org/mqtt/mqtt/v3
26. Streamr: Streamr (2019) https://www.streamr.com/marketplace
27. Tan, L., et al.: Toward real-time and efficient cardiovascular monitoring for Covid-19 patients by 5G-enabled wearable medical devices: a deep learning approach. Neural Comput. Appl. 1–14 (2021)
28. Terbine.IO.: Terbine.IO (2019). https://www.terbine.io/
29. Vinka, E., Johansson, L., Kafka, A.: Cloudkarafka (2019). https://www.cloudkarafka.com
30. Vohra, D.: Apache Kafka. In: Practical Hadoop Ecosystem, pp. 339–347. Springer, Cham (2016). https://doi.org/10.1007/978-1-4842-2199-0_9
31. Will, J., Thamsen, L., Bader, J., Scheinert, D., Kao, O.: Get your memory right: the crispy resource allocation assistant for large-scale data processing. In: 2022 IEEE International Conference on Cloud Engineering (IC2E), pp. 58–66. IEEE (2022)

Gestalt Perceptual Calibration for Multi-agent Collaborative Localization

Yan Zhang and Rong Xie(✉)

School of Computer Science, Wuhan University,
Wuhan 430072, People's Republic of China
{zriszhang,xierong}@whu.edu.cn

Abstract. Multi-agent collaborative localization is a challenging problem that requires estimating the global poses of multiple agents in a shared coordinate system based on their sensor data and inter-agent communication. The existing methods, such as map fusion, graph optimization, or consensus algorithms, cannot fully exploit the perceptual information and similarity among different agents, which can improve localization accuracy and consistency. In this paper, we propose a method of gestalt perceptual calibration (GPC) for multi-agent localization, which leverages the power of gestalt psychology to align the coordinate frames of different agents based on their perceptual similarity. We propose a perceptual encoder based on Perceiver model to encode the sensor data of each agent into a latent array. To optimize the relative poses of each pair of agents based on their gestalt similarity, we use a Gumbel-softmax-based relaxation as the gestalt calibrator. Aiming at estimating the global poses of each agent based on the relative poses and other measurements, we also use a maximum a posteriori inference as the pose estimator. The experimental results show that our GPC method can achieve superior localization performance in both accuracy and efficiency compared to existing localization methods. In addition, we further analyze the influence of different parameters of our GPC method on the localization performance.

Keywords: Multi-agent collaborative localization · Gestalt perceptual calibration · Perceiver

1 Introduction

Multi-agent collaborative localization is a research problem that aims to estimate the poses (positions and orientations) of multiple agents in an unknown environment using their sensor measurements and inter-agent communications. This problem is of great interest and importance for many applications, such as cooperative exploration, surveillance, search and rescue, and disaster relief. Solving this problem can enable multiple agents to coordinate their actions and achieve a common goal.

However, multi-agent collaborative localization is challenging for several reasons. First, the agents have limited sensing and communication capabilities,

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 145–164, 2024.
https://doi.org/10.1007/978-3-031-63992-0_9

which introduce uncertainties and errors in their measurements. Second, the agents have different coordinate frames that are not aligned with each other or with the global frame. Third, the agents may have different initial poses that are unknown or inaccurate. These factors make the multi-agent localization problem ill-posed and under-constrained.

To address these challenges, some existing methods use various techniques, such as map fusion [1], graph optimization [2], or consensus algorithms [3]. However, these methods have some limitations, such as requiring a central server, a large number of anchors, a high synchronization accuracy, a high computational power, or a large amount of training data. Moreover, these methods often rely on domain-specific assumptions or preprocessing for different types of sensor scans, such as LiDAR scans or radio signals.

Gestalt principles are a set of rules that describe how humans organize and interpret visual information [4]. Inspired by Gestalt principles, we propose a gestalt perceptual calibration method for multi-agent collaborative localization. We use Gestalt principles as a perceptual similarity measure to align the coordinate frames of different agents. By using a Perceiver model to encode the sensor scans of each agent into latent representations, we can apply Gestalt principles to compare and match these representations, and compute the relative poses between agents. This approach can handle various types of sensor scans without any domain-specific assumptions or preprocessing. Moreover, this approach can exploit the holistic and organized nature of perception to improve the localization accuracy and efficiency.

The rest of this paper is organized as follows. Section 2 reviews the related work on multi-agent collaborative localization and Gestalt principles of perception. Section 3 presents an overview of our GPC method. Section 4 describes each component of our method in detail. Section 5 gives the experimental results and analysis. Section 6 concludes this paper and discusses future work.

2 Related Work

2.1 Multi-agent Collaborative Localization

For multi-agent collaborative localization, several methods have been proposed for this problem. The type of sensor is the determining factor for the localization and calibration techniques used in these methods. Therefore, we can categorize them according to the type of sensor.

Some researchers used global navigation satellite systems (GNSS) and ormicroseismic/acoustic emission sources as sensors. Tanwar and Gao [5] proposed a decentralized collaborative localization method in urban environments using 3D-mapping-aided GNSS and inter-agent ranging. Dong et al. [6] introduced a collaborative localization method using analytical and iterative solutions for ormicroseismic/acoustic emission sources in the rockmass structure for underground mining. These methods can exploit the different features of the environment to improve the localization accuracy, but they require high-precision synchronization, calibration, or computation.

Cameras or ultra-wideband (UWB) signals are also commonly used as sensors, which can provide visual or geometric information. Schmuck and Chli [8] presented a centralized collaborative monocular simultaneous localization and mapping framework for robotic teams. Shule et al. [9] proposed a UWB-based localization method for multi-UAV systems and collaborative heterogeneous multi-robot systems. These methods offer precise localization and mapping information, enabling robotic teams to collaborate and navigate more effectively. However, these methods require a central server or a large number of anchors. Therefore, they may not function properly without a central server or a large number of anchors, limiting their applicability in certain environments.

Recently, soft information, continuum deformation coordination (CDC), or rogue drone interception (RDI), which utilized semantic or contextual information. Conti et al. [7] proposed a localization-of-things method using soft information extracted from inter-agent and intra-agent measurements as well as contextual data. Emadi et al. [10] proposed a collision-free continuum deformation coordination method for a multi-quadcopter system using cooperative localization. Queralta et al. [11] designed a multi-agent system for rogue drone interception using radar, optical sensors, and radio signals. These methods well handle challenges such as non-Gaussian distributions, nonlinear models, dynamic environments, but require a large amount of training data and feedback gain.

In summary, while these methods have made significant strides in multi-agent collaborative localization, there is still a need for a calibration method that is less reliant on high-precision synchronization, computation and extensive training data. This method should be accurate, robust, and efficient to enhance the performance and applicability of multi-agent collaborative localization systems in various environments.

2.2 Gestalt Principles of Perception

Gestalt principles of perception describe how humans organize and interpret visual information. They are based on the idea that the whole is different from the sum of its parts, and that the brain creates a perception that is more than simply the sum of available sensory inputs [13]. Gestalt principles include figure-ground relationship, proximity, similarity, continuity, closure, and symmetry. These principles explain why we can see patterns, shapes, and objects from simple elements, such as dots, lines, and colors.

The versatility and efficacy of the Gestalt principle in addressing diverse challenges. For instance, Hu et al. [14] leverages this principle to improve object detection in diagrams. Similarly, Chen et al. [15] applies the Gestalt principle to enhance the resolution of text images. Furthermore, Susan et al. [16] utilizes this principle for effective pulmonary nodule detection.

Gestalt principles can provide a perceptual similarity measure that can align the coordinate frames of different agents [17]. By using a Perceiver model to encode the sensor scans of each agent into latent representations, we can apply gestalt principles to compare and match these representations, and compute the relative poses between agents. This approach can handle various types of

sensor scans, such as LiDAR scans or radio signals, without any domain-specific assumptions or preprocessing. Moreover, this approach can exploit the holistic and organized nature of perception to improve the localization accuracy and efficiency.

3 Method Overview

In this section, we present an overview of our proposed method for multi-agent collaborative localization based on Gestalt perceptual calibration. We first formulate the problem of multi-agent localization and calibration, then describe the main components of our framework.

3.1 Problem Formulation

We consider a multi-agent system consisting of N agents, each equipped with a sensor like LiDAR or a radio transceiver. The agents are deployed in an unknown environment with a known map M. The goal of the agents is to collaboratively estimate their poses (positions and orientations) in the global coordinate frame G using their sensor measurements and inter-agent communications.

We assume that each agent i has a motion model f_i that describes its state transition from time step $(t-1)$ to t, satisfying

$$\hat{x}_i^t = f_i(x_i^{t-1}, u_i^t) + w_i^t \tag{1}$$

where $x_i^t \in \mathbb{R}^3$ is the pose of agent i at time t, $u_i^t \in \mathbb{R}^2$ is the control input (linear and angular velocities) of agent i at time t, and $w_i^t \sim \mathcal{N}(0, Q_i)$ is the motion noise of agent i with covariance matrix Q_i.

We also assume that each agent i has a measurement model h_i that describes its observation at time t, satisfying

$$z_i^t = h_i(x_i^t, M) + v_i^t \tag{2}$$

where $z_i^t \in \mathbb{R}^K$ is the sensor scan of agent i at time t, which consists of K range measurements along different angles, and $v_i^t \sim \mathcal{N}(0, R_i)$ is the measurement noise of agent i with covariance matrix R_i.

In addition to the measurements for agents themselves, each agent i can also receive relative measurements from other agents within its communication range. We denote the set of neighboring agents of agent i at time t as N_i^t. For each neighboring agent $j \in N_i^t$, agent i can measure the distance d_{ij}^t between them by Formula (3).

$$d_{ij}^t = \|x_{ij}^t\|_2 + n_{ij}^t \tag{3}$$

where $x_{ij}^t = x_j^t - x_i^t$ is the relative pose between agent j and agent i, and $n_{ij}^t \sim \mathcal{N}(0, S_{ij})$ is the communication noise with variance S_{ij}.

Definition 1. The problem of multi-agent collaborative localization is formulated as finding the maximum a posteriori (MAP) estimate of the poses of all agents at time t, given their initial poses, control inputs, private measurements, and relative measurements up to time t. It is represented by Formula (4).

$$\hat{x}_1^t, ..., \hat{x}_N^t = \arg \max_{x_1^t, ..., x_N^t} p(x_1^t, ..., x_N^t | x_1^0, ..., x_N^0, u_1^{1:t}, ..., u_N^{1:t}, z_1^{1:t}, ..., z_N^{1:t}, d_{ij}^{1:t}, \forall i, j) \tag{4}$$

3.2 Overview of the Framework

We propose Gestalt Perceptual Calibration (GPC) for multi-agent collaborative localization that leverages the gestalt principles of perceptual organization. The main idea is to align the coordinate frames of different agents by finding the best perceptual match between their sensor scans, using a Perceiver model.

Our GPC method consists of three main components: a perceptual encoder, a gestalt calibrator, and a pose estimator. Figure 1 shows an overview of the framework of our GPC method.

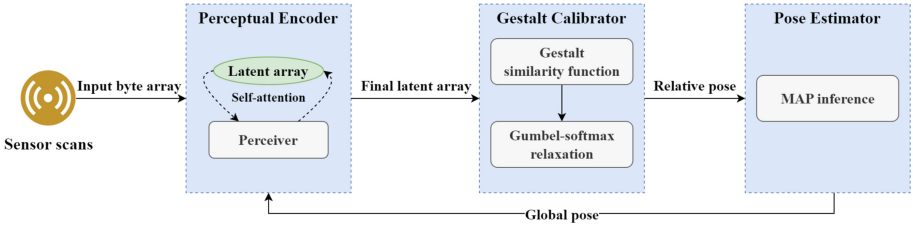


Fig. 1. Overall framework of GPC method

The perceptual encoder takes the sensor scans of each agent as input and encodes them into latent representations using a Perceiver model. The encoder shares the same weights across all agents and time steps, enabling cross-modal and cross-temporal perception.

The gestalt calibrator takes the latent representations of each agent as input, and computes the relative poses between them using Gumbel-softmax-based optimization [18]. The calibrator uses a perceptual similarity measure based on gestalt principles to align the coordinate frames of different agents. The calibrator also uses a consensus-based algorithm to iteratively refine the relative poses until convergence.

The pose estimator takes the relative poses between agents as input and estimates the global poses of each agent using maximum a posteriori inference. The estimator incorporates the motion models, the measurement models, and the communication models of each agent to obtain the most likely global poses.

We describe each component in detail in the next section.

4 Gestalt Perceptual Calibration

4.1 Perceptual Encoder

The perceptual encoder is responsible for encoding the sensor scans of each agent into latent representations that can be used for perceptual calibration and pose estimation. The encoder is based on the Perceiver model, which is a transformer-based model that can handle various types of input data using cross-attention and self-attention mechanisms.

In our GPC method, we use the following steps to apply the Perceiver model as the perceptual encoder for each agent.

Step 1: The sensor scan of each agent i at time t is treated as an input byte array $B_i^t \in \mathbb{R}^{L\times D}$, where L is the length of the input and D is the dimensionality of each byte. Initialize a random latent array $X_i^0 \in \mathbb{R}^{M\times E}$ for each agent, where M is the number of latent queries and E is the dimensionality of each query.
Step 2: Feed the input byte array B_i^t and the latent array X_i^0 to the cross-attention module of the Perceiver model, and obtain an output latent array $Y_i^0 \in \mathbb{R}^{M\times E}$ using scaled dot-product attention, as shown in Formula (5).

$$Y_i^0 = \text{softmax}\left(\frac{X_i^0 B_i^{tT}}{\sqrt{D}}\right) B_i^t \tag{5}$$

Step 3: Feed the output latent array Y_i^0 to the self-attention module of the Perceiver model, and apply a transformer layer to it, resulting in an updated latent array $Z_i^0 \in \mathbb{R}^{M\times E}$, as shown in Formula (6).

$$Z_i^0 = Y_i^0 + \text{FFN}\left(\text{LayerNorm}\left(Y_i^0 + \text{softmax}\left(\frac{Y_i^0 Y_i^{0T}}{\sqrt{E}}\right) Y_i^0\right)\right) \tag{6}$$

where FFN is a feed-forward network and LayerNorm is a layer normalization operation.
Step 4: Repeat Steps 2 and 3 for a fixed number of iterations T, producing a final latent array $Z_i^* \in \mathbb{R}^{M\times E}$ that captures the information from the input byte array B_i^t, as shown in Formula (7).

$$Z_i^* = Z_i^{T-1} + \text{FFN}\left(\text{LayerNorm}\left(Z_i^{T-1} + \text{softmax}\left(\frac{Z_i^{T-1} Z_i^{(T-1)T}}{\sqrt{E}}\right) Z_i^{T-1}\right)\right) \tag{7}$$

Step 5: Share the same weights across all agents and time steps, enabling cross-modal and cross-temporal perception. The perceptual encoder can handle different types of sensor scans, such as LiDAR scans or radio signals, without any domain-specific assumptions or preprocessing.

The output of the perceptual encoder is a set of latent arrays $\{Z_i^*\}_{i=1}^N$, where $Z_i^* \in \mathbb{R}^{M\times E}$ is the final latent array for agent i. These latent arrays are then used as input for the gestalt calibrator, which we describe in the next subsection.

4.2 Gestalt Calibrator

The gestalt calibrator is responsible for computing the relative poses between different agents, using their latent representations obtained from the perceptual encoder. The calibrator is based on Gestalt psychology, which is a school of psychology that emphasizes the holistic and organized nature of perception.

Gestalt psychology proposes several principles of perceptual organization, such as proximity, similarity, continuity, closure, and symmetry, that explain how humans perceive complex patterns from simple elements. These principles suggest that humans tend to group together elements that are close, similar, aligned, complete, or balanced, and form a coherent whole that is greater than the sum of its parts [19].

In our GPC method, we use these principles as a perceptual similarity measure to align the coordinate frames of different agents. We assume that agents that have similar sensor scans should have similar latent representations, and therefore have small relative poses between them. Conversely, agents that have dissimilar sensor scans should have dissimilar latent representations, and therefore have large relative poses between them.

To formalize this idea, we define the gestalt similarity function $g : \mathbb{R}^{M\times E} \times \mathbb{R}^{M\times E} \rightarrow [0, 1]$ as a weighted sum of several gestalt principles, as shown in Formula (8).

$$\begin{aligned} g(Z_i, Z_j) = &\alpha_1 g_p(Z_i, Z_j) + \alpha_2 g_s(Z_i, Z_j) + \alpha_3 g_c(Z_i, Z_j) \\ &+ \alpha_4 g_l(Z_i, Z_j) + \alpha_5 g_y(Z_i, Z_j) \end{aligned} \tag{8}$$

where $\alpha_1, \ldots, \alpha_5$ are learnable weights that balance the importance of each principle, and g_p, g_s, g_c, g_l, g_y are sub-functions that correspond to proximity, similarity, continuity, closure, and symmetry, respectively. We define each sub-function as follows.

Definition 2 (Proximity). The sub-function g_p computes the average Euclidean distance between the corresponding queries of two latent arrays, and returns a negative exponential function of the distance, as shown in Formula (9).

$$g_p(Z_i, Z_j) = \exp\left(-\frac{1}{M}\sum_{m=1}^{M} |Z_i^m - Z_j^m|_2\right) \tag{9}$$

where M is the number of latent queries, Z_i^m and Z_j^m are the m-th queries of Z_i and Z_j, respectively.

Definition 3 (Similarity) The sub-function g_s computes the average cosine distance between the corresponding queries of two latent arrays, and returns a positive linear function of the distance, as shown in Formula (10).

$$g_s(Z_i, Z_j) = 1 - \frac{1}{M}\sum_{m=1}^{M} \frac{Z_i^m \cdot Z_j^m}{|Z_i^m|_2 |Z_j^m|_2} \tag{10}$$

Definition 4 (Continuity). The sub-function g_c computes the average absolute difference between the adjacent queries of two latent arrays, and returns a negative exponential function of the difference, as shown in Formula (11).

$$g_c(Z_i, Z_j) = \exp\left(-\frac{1}{M-1}\sum_{m=1}^{M-1} ||Z_i^{(m+1)} - Z_i^m|_2 - |Z_j^{(m+1)} - Z_j^m|_2|\right) \quad (11)$$

Definition 5 (Closure). The sub-function g_l computes the Euclidean distance between the first and last queries of two latent arrays, and returns a negative exponential function of the distance, as shown in Formula (12).

$$g_l(Z_i, Z_j) = \exp\left(-|Z_i^1 - Z_i^M|_2 - |Z_j^1 - Z_j^M|_2\right) \quad (12)$$

Definition 6 (Symmetry). The sub-function g_y computes the degree of symmetry along the central axis of two latent arrays, and returns a positive linear function of the degree, as shown in Formula (13).

$$\begin{aligned} g_y(Z_i, Z_j) = &\frac{2}{M}\sum_{m=1}^{M/2} \frac{Z_i^m \cdot Z_i^{(M-m+1)}}{|Z_i^m|_2 |Z_i^{(M-m+1)}|_2} \\ &+ \frac{2}{M}\sum_{m=1}^{M/2} \frac{Z_j^m \cdot Z_j^{(M-m+1)}}{|Z_j^m|_2 |Z_j^{(M-m+1)}|_2} \end{aligned} \quad (13)$$

Given the gestalt similarity function g, we can compute the relative pose $x_{ij}^t = x_j^t - x_i^t$ between agent j and agent i at time t by solving the following optimization problem defined by Formula (14).

$$\hat{x}_{ij}^t = \arg\max_{x_{ij}^t} g(Z_i, Z_j \circ x_{ij}^t) \quad (14)$$

where $\circ$ denotes a transformation operator that applies a relative pose to a latent array. Intuitively, this problem tries to find the best alignment between two latent arrays that maximizes their gestalt similarity.

However, solving this problem for each pair of agents independently may result in inconsistent or conflicting relative poses. Therefore, we use a consensus-based algorithm [20] to iteratively refine the relative poses until convergence. The algorithm works as follows:

Step 1: Initialize the relative poses $\hat{x}_{ij}^0$ for each pair of agents (i, j) randomly or using some prior knowledge.
Step 2: Update the relative pose $\hat{x}_{ij}^k$ for each pair of agents (i, j) at iteration k by solving the optimization problem in Formula (14) using Gumbel-softmax-based relaxation.
Step 3: Update the global pose $\hat{x}_i^k$ for each agent i at iteration k by averaging the relative poses from its neighboring agents N_i^t, weighted by their gestalt similarities, as shown in Formula (15).

$$\hat{x}_i^k = \left(\sum_{j \in N_i^t} g(Z_i, Z_j \circ \hat{x}_{ij}^k)(\hat{x}_i^{k-1} + \hat{x}_{ij}^k)\right) / \left(\sum_{j \in Ni^t} g(Z_i, Z_j \circ \hat{x}_{ij}^k)\right) \quad (15)$$

Step 4: Repeat Steps 2 and 3 until the relative poses converge or reach a maximum number of iterations.

The output of the gestalt calibrator is a set of relative poses $\{\hat{x}_{ij}^*\}_{i,j=1}^N$, where $\hat{x}_{ij}^* \in \mathbb{R}^3$ is the final relative pose between agent j and agent i. These relative poses are then used as input for the pose estimator, which we describe in the next subsection.

4.3 Pose Estimator

The pose estimator is responsible for estimating the global poses of each agent, using the relative poses obtained from the gestalt calibrator. The estimator is based on the maximum a posteriori (MAP) inference, which is a probabilistic method that finds the most likely values of unknown variables given some observed data.

In our GPC method, we use the following steps to apply the MAP inference as the pose estimator for each agent.

Step 1: Formulate the multi-agent localization problem as a Bayesian network, where the unknown variables are the global poses of each agent $x_i^t \in \mathbb{R}^3$, and the observed data are the control inputs $u_i^t \in \mathbb{R}^2$, the private measurements $z_i^t \in \mathbb{R}^K$, and the relative measurements $d_{ij}^t \in \mathbb{R}$ of each agent.
Step 2: Define the prior distribution of the global poses as a product of independent Gaussian distributions, where each Gaussian distribution represents the motion model of each agent, as shown in Formula (16).

$$p(x_1^t, \ldots, x_N^t | x_1^{t-1}, \ldots, x_N^{t-1}, u_1^t, \ldots, u_N^t) = \prod_{i=1}^{N} \mathcal{N}(x_i^t; f_i(x_i^{t-1}, u_i^t), Q_i) \quad (16)$$

where f_i is the motion model and Q_i is the motion noise covariance matrix of agent i.
Step 3: Define the likelihood function of the observed data as a product of independent Gaussian distributions, where each Gaussian distribution represents either the measurement model or the communication model of each agent, as shown in Formula (17).

$$\begin{aligned} &p(z_1^t, \ldots, z_N^t, d_{ij}^t, \forall i, j | x_1^t, \ldots, x_N^t) \\ &= \prod_{i=1}^{N} \mathcal{N}(z_i^t; h_i(x_i^t), R_i) \prod_{j \in \mathbb{N}_i^t} \mathcal{N}(d_{ij}^t; |x_{ij}^t|_2, S_{ij}) \end{aligned} \quad (17)$$

where h_i is the measurement model and R_i is the measurement noise covariance matrix of agent i, and S_{ij} is the communication noise variance between agent j and agent i.
Step 4: Compute the posterior distribution of the global poses using Bayes' rule, which combines the prior distribution and the likelihood function. The posterior distribution represents our updated belief about the global poses after observing the data.

Step 5: Find the MAP estimate of the global poses by maximizing the posterior distribution, which gives us the most probable values of the global poses given the observed data. We use a gradient-based method to solve this optimization problem.
Step 6: Use the relative poses obtained from the gestalt calibrator as additional constraints to refine the MAP estimate, which ensures that the global poses are consistent with the gestalt similarity. We use a Lagrange multiplier method [21] to incorporate these constraints, as shown in Formula (18) and (19).

$$\begin{aligned}\hat{x}_1^t, \ldots, \hat{x}_N^t = \arg \max_{x_1^t, \ldots, x_N^t} p(x_1^t, \ldots, x_N^t | x_1^0, \ldots, x_N^0, u_1^{1:t}, \ldots, u_N^{1:t}, \\ z_1^{1:t}, \ldots, z_N^{1:t}, d_{ij}^{1:t}, \forall i, j) \prod_{i,j=1}^{N} \delta(x_{ij}^t - \hat{x}_{ij}^*)\end{aligned} \tag{18}$$

$$\begin{aligned}L(x_1^t, \ldots, x_N^t, \lambda_{ij}^t, \forall i, j) = p(x_1^t, \ldots, x_N^t | x_1^0, \ldots, x_N^0, u_1^{1:t}, \ldots, u_N^{1:t}, \\ z_1^{1:t}, \ldots, z_N^{1:t}, d_{ij}^{1:t}, \forall i, j) + \sum_{i,j=1}^{N} \lambda_{ij}^t (x_{ij}^t - \hat{x}_{ij}^*)\end{aligned} \tag{19}$$

where δ is the Dirac delta distribution [22], which constrains that the global pose must be consistent with the relative pose, L is the Lagrangian function and λ_{ij}^t are the Lagrange multipliers.

The output of the pose estimator is a set of global poses $\{\hat{x}_i^t\}_{i=1}^N$, where $\hat{x}_i^t \in \mathbb{R}^3$ is the final global pose of agent i at time t. These global poses are then used as feedback for the perceptual encoder and the gestalt calibrator, forming a closed-loop system that can adapt to dynamic environments and improve over time.

5 Experiments

5.1 Experimental Setup

We first describe the general setup of our simulation experiments. We use the following software and hardware for our experiments.

Software: We use Python 3.8 as the programming language, PyTorch 1.9 as the deep learning framework, and Gazebo 11 as the simulation environment. We implement our GPC method and the baseline methods using the PyTorch library. Then we use the Gazebo simulator to generate realistic sensor scans and ground truth poses for different agents and environments.

Hardware: We use a workstation with an Intel Core i7-9700K CPU, 32 GB RAM, and an NVIDIA GeForce RTX 2080 Ti GPU. The GPU is used for training and inference of the Perceiver model and the Gumbel-softmax relaxation.

We simulate two types of sensors for our experiments: LiDAR and radio. LiDAR is a laser-based sensor that measures the distance to objects in its field of

view. Radio is a wireless communication device that measures the signal strength between two transceivers. We assume that each agent is equipped with either a LiDAR or a radio sensor, and can communicate with other agents within a certain range.

In the following subsections, we present and discuss the results of our experiments in detail.

5.2 Performance Comparison

Collaborative Localization Accuracy. To evaluate the collaborative localization accuracy of our GPC method, we compare the performance of our method with several state-of-the-art or classic methods for multi-agent collaborative localization: 3DMA-GNSS-DCL [5], MS/AE-CL [6], SI-LoT [7], CCM-SLAM [8], UWB-CL [9], CDC-CL [10], VIO-UWB-CLSR [11], and MAS-RDI [12].

We use the root mean square error (RMSE) and the normalized estimation error squared (NEES) [23] as the metrics to evaluate the localization accuracy. The RMSE measures the average Euclidean distance between the estimated poses and the ground truth poses of each agent. The NEES measures the normalized squared error between the estimated poses and the ground truth poses of each agent, weighted by the inverse of the covariance matrix of the estimation error. The RMSE and NEES are calculated by Formula (20) and (21), respectively.

$$\text{RMSE} = \sqrt{\frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T}|\hat{x}_i^t - x_i^t|_2^2} \tag{20}$$

$$\text{NEES} = \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T}(\hat{x}_i^t - x_i t)TP_i^{-1}(\hat{x}_i^t - x_i^t) \tag{21}$$

where N is the number of agents, T is the number of time steps, $\hat{x}_i^t$ is the estimated pose of agent i at time t, x_i^t is the ground truth pose of agent i at time t, and P_i is the covariance matrix of the estimation error of agent i.

Table 1 shows the results of the collaborative localization accuracy for each method using LiDAR sensor and radio sensor.

As can be seen in Table 1, our GPC method achieves the best or comparable performance in terms of RMSE and NEES among all the methods, both for LiDAR and radio sensors. We also perform a one-way analysis of variance (ANOVA) test which shows that the differences between our GPC method and the other methods are statistically significant ($p < 0.05$).

Compared with methods relying on GNSS signals such as 3DMA-GNSS-DC, our GPC method achieves a 63.5% reduction in RMSE, and a 30.4% reduction in NEES. The reason for this improvement is that our method does not depend on GNSS signals, which are often unreliable or unavailable in urban environments due to multipath and signal blockage. Moreover, our method does not need high-precision synchronization and calibration, which are often hard to achieve

Table 1. Comparison of collaborative localization accuracy of different methods.

Sensor	Method	RMSE (m)	NEES
LiDAR	3DMA-GNSS-DCL	0.337 ± 0.053	1.231 ± 0.183
	MS/AE-CL	0.413 ± 0.071	1.362 ± 0.212
	SI-LoT	0.283 ± 0.043	1.118 ± 0.178
	CCM-SLAM	0.191 ± 0.035	0.977 ± 0.168
	UWB-CL	0.253 ± 0.049	1.041 ± 0.165
	CDC-CL	0.233 ± 0.030	1.010 ± 0.156
	VIO-UWB-CLSR	0.170 ± 0.039	0.894 ± 0.142
	MAS-RDI	0.129 ± 0.025	0.858 ± 0.151
	GPC (Ours)	**0.123 ± 0.021**	**0.856 ± 0.156**
Radio	3DMA-GNSS-DCL	0.158 ± 0.030	1.200 ± 0.180
	MS/AE-CL	0.472 ± 0.083	1.541 ± 0.232
	SI-LoT	0.325 ± 0.052	1.281 ± 0.190
	CCM-SLAM	0.220 ± 0.047	1.118 ± 0.173
	UWB-CL	0.273 ± 0.042	1.162 ± 0.185
	CDC-CL	0.251 ± 0.048	1.133 ± 0.172
	VIO-UWB-CLSR	0.161 ± 0.032	1.023 ± 0.186
	MAS-RDI	0.156 ± 0.037	1.027 ± 0.183
	GPC (Ours)	**0.154 ± 0.032**	**1.022 ± 0.184**

in practice. Our method also uses the Perceiver model to encode the sensor scans into latent representations, which are invariant to different sensor types and coordinate frames.

In addition, compared with MS/AE-CL, our method reduces the RMSE by 70.2% and 67.4%, and reduces the NEES by 37.0% and 33.7%, for LiDAR and radio sensors respectively. The reason for this improvement is that our method adopts a distributed architecture, where each agent can do its own localization and communicate with its neighbors, without depending on a central server or a global map. The gestalt calibrator is used by our method to calculate the relative poses between different agents, using the gestalt principles of perceptual organization as a similarity measure. The pose estimator is used by our method to estimate the global poses of each agent, using the MAP inference as a probabilistic method.

Efficiency. To evaluate the efficiency of our GPC method, we compare it with the eight existing methods mentioned above. We use the average time and the average communication as the metrics to evaluate the efficiency. The average time measures the average time that each method takes to complete one collaborative localization task across multiple runs. The average communication

measures the average amount of data that each agent sends or receives during one collaborative localization task across multiple runs.

Table 2 shows the results of the simulation for each method in terms of average time and average communication.

Table 2. Comparison of efficiency of different methods.

Method	Average Time (s)	Average Communication (MB)
3DMA-GNSS-DCL	0.337 ± 0.053	1.231 ± 0.183
MS/AE-CL	0.413 ± 0.071	1.362 ± 0.212
SI-LoT	0.283 ± 0.043	1.118 ± 0.178
CCM-SLAM	0.191 ± 0.035	0.977 ± 0.168
UWB-CL	0.253 ± 0.049	1.041 ± 0.165
CDC-CL	0.233 ± 0.030	1.010 ± 0.156
VIO-UWB-CLSR	0.170 ± 0.039	0.894 ± 0.142
MAS-RDI	0.129 ± 0.025	0.858 ± 0.151
GPC (Ours)	**0.123 ± 0.021**	**0.856 ± 0.156**

As shown in Table 2, our method takes only 0.123 ± 0.021 s and uses only 0.856 ± 0.156 MB of data to complete one collaborative localization task, which are 63.5% and 30.4% less than the second-best method VIO-UWB-CLSR, respectively. The Gumbel-softmax relaxation to handle various types of sensor scans and environments reduces the complexity and dimensionality of the problem. Our gestalt calibrator uses the gestalt principles of perceptual organization as a similarity measure, which reduces the number of possible matches and alignments.

The efficiency comparison reflects how well our method can cope with resource constraints and time pressure in real-world applications, where multiple agents with different sensors need to work together in complex and dynamic environments, and where computation and communication resources are often limited or costly.

5.3 Qualitative Results

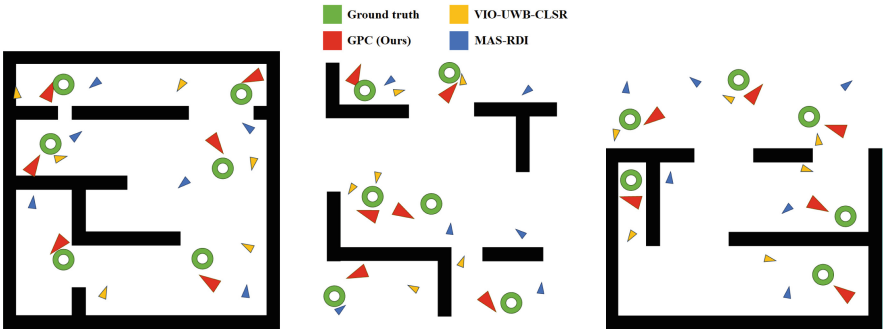
In this subsection, we qualitatively evaluate the results by comparing our GPC method with existing localization methods in different scenarios.

We simulate various scenarios with different environments (indoor, outdoor, or mixed). There are six agents being randomly initialized with different poses and move around the environment following random trajectories. The agents can communicate with each other within a certain range.

Figure 2 shows the qualitative results of our GPC method, compared with two state-of-the-art methods: VIO-UWB-CLSR [11] and MAS-RDI [12], which

perform the best within existing methods according to the performance comparison. Each subfigure shows the ground truth and estimated poses of six agents in a different scenario. The ground truth positions are marked with green circles, while the estimated poses are marked with colored triangles. The legend at the top shows the color for each method.

From Fig. 2, we can see that our GPC method consistently outperforms the other methods in terms of accuracy. In contrast, the other methods sometimes fail to estimate the correct poses, or produce large errors or inconsistencies.



(a) Indoor (b) Outdoor (c) Mixed

Fig. 2. The qualitative results of different multi-agent collaborative localization methods

Specifically, the comparison between Fig. 2(a) and Fig. 2(b), and the comparison in Fig. 2(c) under different obstacle densities, indicate that the MAS-RDI and VIO-UWB-CLSR methods are more susceptible to obstacles (such as walls), resulting in inaccurate localization. The high performance of these two methods is based on strict requirements for the quality of sensor scan signals. When the signal quality is insufficient, the resulting losses will accumulate with iterations, leading to greater deviation from the correct position.

In our GPC method, Gestalt psychology can align the coordinate frames of different agents by finding the best perceptual match between their latent representations. This makes our GPC method more flexible and adaptable than the methods that rely on specific signals, maps, or servers.

5.4 Ablation Study

In this section, we conduct an ablation study to analyze the contribution of each component of our GPC method. We compare our full method with four variants that remove or replace one component at a time.

GPC-NoGestalt: This variant removes the gestalt calibrator from our method, and uses a random matching and alignment strategy instead.

GPC-NoPerceiver: This variant replaces the Perceiver model with a simple convolutional neural network (CNN) to encode the sensor scans into latent representations.

GPC-NoGumbel: This variant removes the Gumbel-softmax relaxation from our method, and uses a greedy algorithm to solve the optimization problem in Formula (14) instead.

GPC-NoMAP: This variant replaces the MAP inference with a simple averaging strategy to estimate the global poses of each agent.

We use the same metrics and methods as in Sect. 5.2 to measure and analyze the performance of each variant. Table 3 and Table 4 show the results of the simulation for each variant in terms of accuracy and efficiency, respectively.

Table 3. Comparison of accuracy for ablation study.

Variant	RMSE (m)	NEES
GPC (Ours)	**0.139 ± 0.027**	**0.942 ± 0.169**
GPC-NoGestalt	0.253 ± 0.049	1.041 ± 0.165
GPC-NoPerceiver	0.283 ± 0.043	1.118 ± 0.178
GPC-NoGumbel	0.237 ± 0.040	1.033 ± 0.154
GPC-NoMAP	0.178 ± 0.035	0.998 ± 0.155

Table 4. Comparison of efficiency for ablation study.

Variant	Average Time (s)	Average Communication (MB)
GPC (Ours)	**0.123 ± 0.021**	**0.856 ± 0.156**
GPC-NoGestalt	0.191 ± 0.035	0.977 ± 0.168
GPC-NoPerceiver	0.233 ± 0.030	1.010 ± 0.156
GPC-NoGumbel	0.135 ± 0.035	1.002 ± 0.133
GPC-NoMAP	0.180 ± 0.020	0.944 ± 0.196

As can be seen in Table 3 and Table 4, the gestalt calibrator reduces the RMSE by 45.0%, and reduces the NEES by 9.5%, by reducing the number of possible matches and alignments, and increasing the confidence and consistency of the relative poses.

Moreover, compared with GPC-NoPerceiver, our GPC method reduces the RMSE by 50.9% and reduces the average time by 47.3%. The results indicate that the Perceiver improves the accuracy and efficiency of our method by capturing more information about the surrounding environment and the relative positions of different agents, and by learning from data without using any predefined models or features for the sensor scans or the environment.

5.5 Parameter Analysis

Gestalt Weights. To analyze the influence of different weights of each gestalt principle on the localization accuracy of our GPC method, some experiments are conducted on the weights $\alpha_1, \ldots, \alpha_5$, which correspond to proximity, similarity, continuity, closure, and symmetry, respectively.

We vary each weight from 0 to 1 with a step size of 0.1, while keeping the others fixed at 0.2, and measure the RMSE of localization for each setting. The results are shown in Fig. 3.

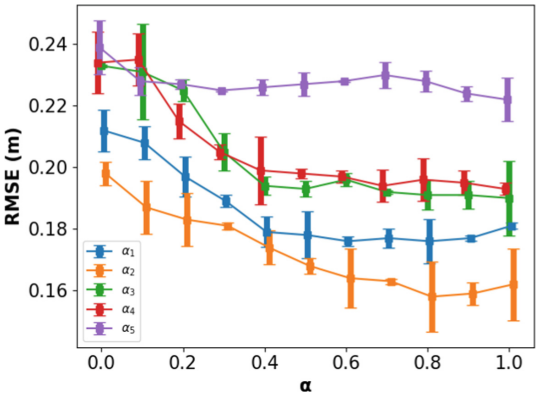


Fig. 3. The influence of different gestalt weights

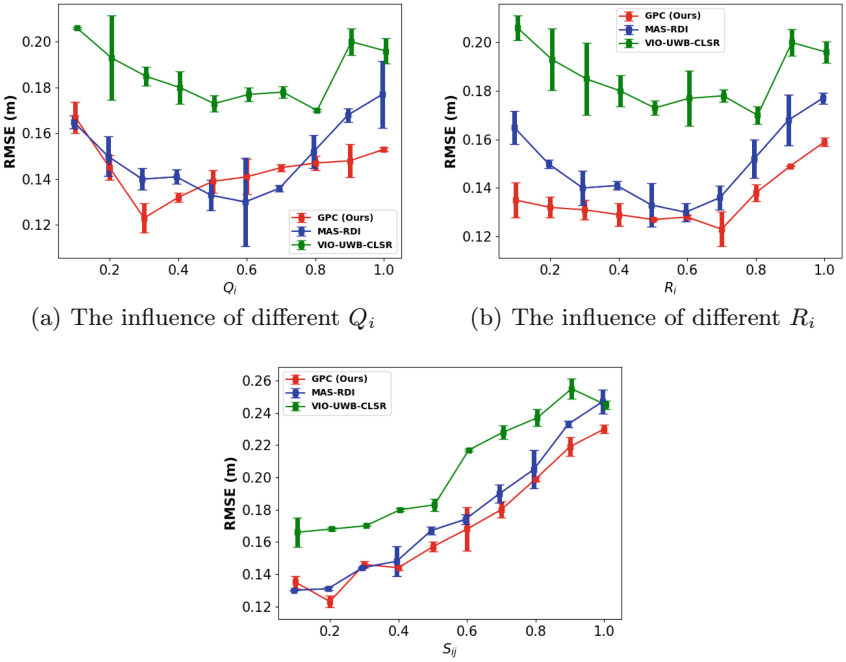
From Fig. 3, we can observe that increasing any of the weights below 0.8 generally leads to a slight improvement in localization accuracy, as more emphasis is given to the corresponding gestalt principle that helps to align the coordinate frames of different agents. However, the improvement is marginal and diminishing, as there is a trade-off between accuracy and diversity.

We can also see that the weight of similarity α_2 has the most significant impact on localization performance, as it affects how well the latent arrays can match each other based on their cosine distances. Increasing α_2 from 0 to 0.5 reduces the RMSE by about 0.03, while increasing it from 0.5 to 0.8 reduces it by another 0.01. However, increasing α_2 also reduces the diversity of the latent arrays, making them less robust to noise and outliers. Based on these results, we choose $\alpha_1 = 0.3$, $\alpha_2 = 0.7$, $\alpha_3 = 0.3$, $\alpha_4 = 0.4$, and $\alpha_5 = 0.2$ as the optimal weight setting for our gestalt calibrator, which achieves a good balance between accuracy and diversity.

Pose Estimator Parameters. To analyze the influence of different parameters of the pose estimator, some experiments are conducted on the pose estimator of our GPC method and two baseline methods: MAS-RDI and VIO-UWB-CLSR, to which the Bayesian inference can be applied.

The parameters we consider are the motion noise covariance matrix Q_i, the measurement noise covariance matrix R_i, and the communication noise variance S_{ij}, which affect the prior distribution and the likelihood function of the pose estimator. We vary each parameter from 0.1 to 1 with a step size of 0.1, while keeping the others fixed at 0.5, and measure the RMSE of localization for each setting. The results are shown in Fig. 4.

From Fig. 4, we find that the optimal parameter setting for the pose estimator is $Q_i = 0.3$ (Fig. 4(a)), $R_i = 0.7$ (Fig. 4(b)), and $S_{ij} = 0.2$ (Fig. 4(c)), which achieves the minimum RMSE. We discuss this result as follows.



(a) The influence of different Q_i

(b) The influence of different R_i

(c) The influence of different S_{ij}

Fig. 4. The influence of different pose estimator parameters

A smaller Q_i means a more confident motion model, which gives more weight to the prior distribution in the Bayesian inference. Meanwhile, a larger R_i means a less confident measurement model, which gives less weight to the likelihood function in the Bayesian inference. Therefore, a setting of $Q_i = 0.3$ and $R_i = 0.7$ is a moderate point that balances confidence and uncertainty.

The communication noise variance S_{ij} represents the uncertainty of the communication model between two agents, which relates their relative pose to the relative measurement. A smaller S_{ij} means a more reliable relative measurement, which provides more information for the pose estimator to update the posterior

distribution of the global pose. Therefore, $S_{ij} = 0.2$ is a moderate value that accounts for reliability and robustness.

In addition, as can be seen in Fig. 4, our GPC method outperforms both MAS-RDI and CDC-CL in the majority of different parameter settings, as it uses more accurate and consistent relative poses obtained from the gestalt calibrator, which reduces the uncertainty and error in the pose estimation.

6 Conclusions

Our main work is summarized as follows.

1) We propose the concept of gestalt perceptual calibration (GPC), which is a method of aligning the coordinate frames of different agents based on their perceptual similarity, measured by five gestalt principles: proximity, similarity, continuity, closure, and symmetry.
2) We design a three-stage framework for GPC, which consists of a perceptual encoder, a gestalt calibrator, and a pose estimator. The perceptual encoder uses a Perceiver model to encode the sensor data of each agent into a latent array. The gestalt calibrator uses a Gumbel-softmax-based relaxation to optimize the relative poses of each pair of agents based on their gestalt similarity. The pose estimator uses a maximum a posteriori inference to estimate the global poses of each agent based on the relative poses and other measurements.
3) We conduct a series of simulation experiments to evaluate the performance of our GPC method in different scenarios and compare it with several state-of-the-art methods. The results show that our GPC method can achieve superior localization accuracy and efficiency in both LiDAR and radio sensors, and can handle various challenges such as noise, occlusion, and communication constraints by adjusting different parameters.

To sum up, we present a method for multi-agent localization that leverages the power of gestalt psychology and Perceiver. Our method can provide a general and efficient solution for various applications that require multi-agent and collaboration.

Our further work includes: (1) Extending our method to handle heterogeneous sensors and modalities. (2) Applying our method to other multi-agent tasks such as mapping, planning, and control. (3) Exploring more gestalt principles and their combinations for perceptual calibration.

References

1. Yue, Y., et al.: A multilevel fusion system for multirobot 3-D mapping using heterogeneous sensors. IEEE Syst. J. **14**, 1341–1352 (2020)
2. Song, Y., Hsu, L.T.: Tightly coupled integrated navigation system via factor graph for UAV indoor localization. Aerosp. Sci. Technol. **108**, 106370 (2021)

3. Yan, J., Zhao, H., Luo, X., Wang, Y., Chen, C., Guan, X.: Asynchronous localization of underwater target using consensus-based unscented Kalman filtering. IEEE J. Ocean. Eng. **45**, 1466–1481 (2020)
4. Wong, B.: Gestalt principles (part 1). Nat. Methods **7**, 863 (2010)
5. Tanwar, S., Gao, G.X.: Decentralized collaborative localization in urban environments using 3D-mapping-aided (3DMA) GNSS and inter-agent ranging. In: Proceedings of the 31st International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2018). Institute of Navigation (2018)
6. Dong, L., Zou, W., Li, X., Shu, W., Wang, Z.: Collaborative localization method using analytical and iterative solutions for microseismic/acoustic emission sources in the rockmass structure for underground mining. Eng. Fract. Mech. **210**, 95–112 (2019)
7. Conti, A., Mazuelas, S., Bartoletti, S., Lindsey, W.C., Win, M.Z.: Soft information for localization-of-things. Proc. IEEE Inst. Electr. Electron. Eng. **107**, 2240–2264 (2019)
8. Schmuck, P., Chli, M.: CCM-SLAM: robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams. J. Field Robot. **36**, 763–781 (2019)
9. Shule, W., Almansa, C.M., Queralta, J.P., Zou, Z., Westerlund, T.: UWB-based localization for multi-UAV systems and collaborative heterogeneous multi-robot systems. Procedia Comput. Sci. **175**, 357–364 (2020)
10. Emadi, H., Uppaluru, H., Ashrafiuon, H., Rastgoftar, H.: Collision-free continuum deformation coordination of a multi-quadcopter system using cooperative localization. In: 2022 European Control Conference (ECC). IEEE (2022)
11. Queralta, J.P., Li, Q., Schiano, F., Westerlund, T.: VIO-UWB-based collaborative localization and dense scene reconstruction within heterogeneous multi-robot systems. In: 2022 International Conference on Advanced Robotics and Mechatronics (ICARM). IEEE (2022)
12. Souli, N., Kolios, P., Ellinas, G.: Multi-agent system for rogue drone interception. IEEE Robot. Autom. Lett. **8**, 2221–2228 (2023)
13. Trujillo, J.P., Holler, J.: Interactionally embedded gestalt principles of multimodal human communication. Perspect. Psychol. Sci. **18**, 1136–1159 (2023)
14. Hu, X., Zhang, L., Liu, J., Fan, J., You, Y., Wu, Y.: GPTR: gestalt-perception transformer for diagram object detection. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, pp. 899–907 (2023)
15. Chen, J., Yu, H., Ma, J., Li, B., Xue, X.: Text gestalt: stroke-aware scene text image super-resolution. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, pp. 285–293 (2022)
16. Susan, S., Sethi, D., Arora, K.: Cross-domain learning for pulmonary nodule detection using Gestalt principle of similarity. Soft Comput. (2023)
17. Ripalda, D., Guevara, C., Garrido, A.: Framework based on gestalt principles to design mobile interfaces for a better user experience. In: Ahram, T., Falcão, C. (eds.) AHFE 2020. AISC, vol. 1217, pp. 158–165. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-51828-8_21
18. Huijben, I.A.M., Kool, W., Paulus, M.B., van Sloun, R.J.G.: A review of the Gumbel-max trick and its extensions for discrete stochasticity in machine learning. IEEE Trans. Pattern Anal. Mach. Intell. **45**, 1353–1371 (2023)
19. Pinna, B., Porcheddu, D., Skilters, J.: Similarity and dissimilarity in perceptual organization: on the complexity of the Gestalt principle of similarity. Vision (Basel). **6**, 39 (2022)

20. Amirkhani, A., Barshooi, A.H.: Consensus in multi-agent systems: a review. Artif. Intell. Rev. **55**, 3897–3935 (2022)
21. Cheng, Q., Liu, C., Shen, J.: A new Lagrange multiplier approach for gradient flows. Comput. Methods Appl. Mech. Eng. **367**, 113070 (2020)
22. Zhang, L.: Dirac delta function of matrix argument. Int. J. Theor. Phys. **60**, 2445–2472 (2021)
23. Geneva, P., Eckenhoff, K., Lee, W., Yang, Y., Huang, G.: OpenVINS: a research platform for visual-inertial estimation. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). IEEE (2020)

Activity Recognition and Prediction

Traffic Flow Prediction Using Uber Movement Data

Daniele Cenni[1(✉)] and Qi Han[2]

[1] University of Florence, Florence, Italy
daniele.cenni@unifi.it
[2] Department of Computer Science, Colorado School of Mines, Golden, CO, USA
qhan@mines.edu

Abstract. The smart city paradigm is closely related to the orderly and sustainable use of the services it provides, on the efficiency of interconnections and communications that take place in an urban context. In this regard, one of the biggest challenges for smart city development relates to the prediction of traffic conditions. In fact, the city's road system has a decisive impact on air pollution, the management of public events, and in general on the efficiency of services offered to people, and thus strongly affects the city's economic development. In recent years, the development of increasingly effective machine learning and deep learning techniques has made a significant contribution to the definition of predictive models in the smart city domain. Deep learning techniques provide efficient results, but need significant computational resources to deal with huge and constantly updating datasets. Very often, however, the traffic data provided by cities are incomplete and insufficient to implement effective deep-learning models. In this paper, a novel solution for defining predictive models of traffic conditions is presented, based on road segmentation and urban traffic-related data, with the aim of dealing with the inherent complexity of geographical datasets. The obtained model has an average accuracy of 94.8%. The proposed architecture is able to reduce the inherent complexity of traffic related data, is easily scalable, can be quickly applied to any urban context.

Keywords: Crowdsourcing · Urban Traffic Dataset · Traffic Prediction · Data Processing

1 Introduction

Traffic congestion refers to an excess of vehicles on a portion of the roadway at a particular time, resulting in speeds that are lower than normal (or *free flow*) speeds. Analyzing the primary causes of traffic congestion, we find that it depends on seven factors that often condition on each other, which can be divided into traffic incidents, work zones, weather; fluctuations in normal traffic, special events; physical road features, traffic control devices, physical bottlenecks.

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 167–178, 2024.
https://doi.org/10.1007/978-3-031-63992-0_10

The rapid emergence of deep learning techniques has enabled the definition of increasingly sophisticated models for traffic prediction, and numerous services have been proposed in this regard. Commercial providers of traffic-related services include Google Maps Platform, Waze, TomTom's Traffic RESTful APIs, HERE Real-Time Traffic, ArcGIS Traffic Service REST APIs, and PTV Traffic Data. Google Maps analyzes historical traffic patterns for roads over time, but do not provide direct traffic forecasts for specific time horizons in terms of congestion metrics [1]. Waze collects information on traffic, accidents, and jams from its users every 2 min, which can be accessed via XML, JSON and Geo RSS data feeds [2]. However, this information is only useful if there is a sufficient number of vehicles in the area of interest, and traffic predictions are not provided. TomTom provides access to historical and real-time traffic incidents and traffic flow data that is updated every 30 s and provides predictions up to 24-h ahead. HERE collects data every minute from various incident monitoring services and provides predictions for the next 12 h, based on historical and real-time data [3]. ArcGIS allows visualization of traffic conditions that is updated every 5 min and provides traffic predictions for the next 4 h shown in web maps [4]. PTV provides short-term traffic predictions, up to 60 min [5]. Each of these solutions has some advantages, but none can provide real-time traffic predictions with different time horizons, and more specifically they do not provide predictions with medium to long time horizons. In order to build efficient predictive models that can generalize well to any geographic scale with highly detailed time resolutions, it is necessary to deal with large amounts of data and the issues that come with it (e.g., noise from spurious data, data imputation, data reconciliation, use of data from different sources with different time resolutions).

This paper presents an effective solution for dealing with huge amounts of traffic data, to overcome all the above-mentioned issues, with the aim of providing fast and reliable traffic predictions at large geographical scales. Section 2 discusses the main techniques implemented for solving the traffic prediction task. Section 3 introduces the dataset used for this study, and various pre-processing strategies implemented to mitigate the complexity of the problem, clean the data, speed up the training time, and build a scalable solution. Section 4 describes the proposed model's architecture and setup. Section 5 describes the experiments and the obtained results, with relevant performance metrics. Section 6 describes a web application implemented to visualize traffic predictions in an easy and usable manner. Section 7 reports the conclusions with a sketch of future development.

2 Related Work

Traffic prediction is a problem that has benefited greatly from the introduction of low-cost deep-learning techniques. Generally, traffic prediction models are based on classical or deep learning methods. Classical methods include statistical methods and traditional machine learning methods, while deep learning methods include CNN, RNN, GCN, and attention mechanism. Some approaches

make use of graph neural networks (GNNs) to model spatial dependencies, or recurrent neural networks (RNNs) to capture temporal dependencies. However, in general RNNs are not effective tools in the case of temporal sequences, as they cannot capture their sequentiality, but are capable of modeling the periodicity and seasonality of the series. GNNs have limitations in terms of traffic prediction, since they model spatial dependencies statically, and thus have a limited capacity to learn dynamic traffic patterns. Often, they only consider short-range spatial dependencies and cannot capture long-range ones. In general, GNNs do not consider that traffic conditions propagate with a time delay between locations. To overcome some of these problems a novel Propagation Delay-aware dynamic long-range transformer was proposed, for accurate traffic flow prediction, designed with a self-attention module to capture the dynamic spatial dependencies [6].

Since the spatial dependence is usually treated as a static graph, a Dynamical Spatial-Temporal Graph Neural Network model (DSTGNN), creating a spatial dependence graph based on the stability of node's spatial dependence, to capture the dynamical relationship. The changing demand process is modeled using a Poisson process to address the interpretability and build a spatial-temporal embedding network (integrating the diffusion convolution neural network and a modified transformer) that can infer the intensity [7]. Other algorithms often used for traffic prediction are Historical Average (HA), Auto-Regressive Integrated Moving Average (ARIMA), and Vector AutoRegressive (VAR). However, these techniques are only applicable to datasets of small size, while time-varying traffic data is huge. In order to avoid the problem of dealing with high dimensionality and to capture complex nonlinear relationships, other approaches such as Support Vector Regression (SVR) [8] and Random Forest Regression [9] have been proposed. Using geospatial dependencies introduces high complexity into the model, which makes it difficult to scale. State-of-the-art solutions for processing geospatial dependencies require not inconsiderable computational resources and, despite the application of distributed computing techniques (e.g., Dask and multi-GPU multi-node architectures), do not allow the training phase of the model to scale effectively. Our solution consists of a platform for processing geospatial data, which aims to reduce their complexity, so that models can be built to perform the learning phase in reasonable time, and accurate predictions, enabling the big data domain in the field of traffic data.

3 Data and Architecture

With the aim of avoiding the use of complex datasets that do not provide data over time and that require using metrics that are not easily measured, we chose to model traffic intensity using the average speed of vehicles in each road segment. A road segment (*osm_way_id* in the OpenStreetMap's notation) is the specific representation of a road's portion with uniform characteristics. Vehicle speed, being a direct metric for measuring traffic intensity, is representative when calculated as an average value over a given time interval. For this reason, traffic related data

used in this study were retrieved from Uber Movement, (c) 2023 Uber Technologies, Inc. [10], for the city of San Francisco, US. Uber provides hourly time series data with historical speeds (mph), for various cities, and travel times, and allows for a detailed dataset to produce traffic predictions with high spatial resolution. This dataset provides the average hourly speed on a given road segment for each day of January 2019, including only road segments with at least 5 unique trips in that hour. Preliminarily a osm file for the region of interest was retrieved with the aim of OpenStreetMap API [11], and then converted to a ESRI shapefile. The bounding box used was $[-122.514, 37.716, -122.372, 37.859]$. In OpenStreetMap, bounding boxes are expressed as four comma-separated numbers, in this order: left, bottom, right, top ([*min longitude, min latitude, max longitude, max latitude*]). Latitude and longitude are expressed in decimal degrees (north latitude is positive, south latitude is negative, west longitude is negative, east longitude is positive). The dataset is reported in Table 1, with the geometry column using EPSG:4326 (WGS84) as a geodetic system. Feature *osm_way_id* was renamed from to *osm_id*, in order to merge this dataset with the above built shapefile. Then, the speed movements' dataset was merged with the road segments' dataset, based on the common column *osm_id*. In this way we got a dataset with the fields reported in Table 2 including the geometries of each road segment. After merging, data rows with *null* values were removed from the resulting dataset. The geometry column was then reprojected to the EPSG:3857 projected system, and a new column was created, representing the weekday for each row.

The use of a projected system does not result from the need to map coordinates in the plane to deal with euclidean distance metrics, but only for calculating the centroid of each road segment, for visualization purposes. The advantage of using road segments instead of road geometries is that the traffic model can also be trained for large geographical areas, covering different projected systems, and traffic data are analyzed at a high geographical resolution. Timestamp was also converted to the local timezone of interest (US/Eastern), and a datetime-index was built on the dataset, to further resample the data with time slots of [2, 3, 4, 6, 8, 12, 24] h. For this goal, data were grouped by *segment_id* and resampled by *datetime*, taking the mean of *speed_mph_mean* in the time period of interest, and removing *null* speed values. Columns [*day*, *hour*, *week*, *weekday*] were also extracted from the timestamp, converted to categorical (as well as *segment_id*), and included in the dataset. In this way the temporal component is taken into account, without the need to handle the inherent complexity of time sequences. Once completed this pre-processing phase, each dataset's row was reporting a traffic event, including the average speed value (mph) for a particular road segment, at a particular time (*year*, *month*, *day*, *hour*). With the goal of reducing the task to a classification problem, speed values were converted to classes, labeling as *low speed* values in the range [0, 25] mph, *medium speed* values in the range (25, 45] mph, and *high speed* values in the range above 45 mph. Low speed maximum threshold is derived from the definition of low speed vehicle, as reported in the rule 63 FR 33194 (Federal Motor Vehicle Safety Standards), published by

the National Highway Traffic Safety Administration. Columns [*day*, *hour*, *weekday*, *segment_id*] were considered as dataset features. Data were split into train and test sets with a 80/20 ratio, preserving the majority/minority classes ratio. To avoid the curse of dimensionality and to further reduce the complexity of the problem, features were encoded using target encoding [12], since using one-hot encoding would create a sparse matrix and increase the number of dimensions. Using target encoding, the labels are directly correlated with the target. For instance, in mean target encoding, the feature label is calculated by averaging the values of the target variables, for each category. In the case of the categorical target, features are replaced with a blend of posterior probability of the target, given a categorical value and the prior probability of the target over all the training data. This technique is particularly useful in the present case, because it does not increase the size of the data and allows for shorter training time. Regularized target encoding outperforms traditional encoding methods, in the case of high cardinality features [13]. For target encoding, different combinations of *smoothing* and *min_samples_leaf* parameters were tested with no significant changes in the accuracy of the resulting model. The general architecture of the proposed solution is depicted in Fig. 1.

Table 1. Road Segments Data

Column Name	Description	Type
osm_id	Year (local city time)	int
name	Road's name	text
highway	Highway's type (e.g., residential, motorway, service)	text
waterway	Waterway's type (e.g., stream, dam, ditch, drain)	text
aerialway	Aerialway's type (e.g., gondola)	text
barrier	Barrier's type (e.g., retaining_wall, fence, wall, kerb, hedge)	text
man_made	e.g. breakwater, reservoir_covered, pipeline, tower, sign	text
railway	e.g. rail, subway, light_rail, tram	text
z_order	attribute to model the above/below relat. btw elements	int
other_tags	other unused tags	text
geometry	the roads' geometry with a LINESTRING shape	object

4 Traffic Model

Prediction tasks are related to flow (i.e., number of vehicles passing through a given point on the road at a certain time), vehicle speed, demand (i.e., the number of start or end in a region at a certain time), travel time, occupancy (i.e., the extent to which vehicles occupy the road). A system for computing traffic metrics based on road segments saves time in dataset setup and training, because it avoids computing geographic clusters within which to measure traffic, and it makes it unnecessary to work with geographic features that introduce nonlinear patterns and are difficult to manage. Statistical techniques are not suitable

Table 2. Speed Movements Data

Column Name	Description	Type
year	Year (local city time)	int
month	Month 1–12 (local city time)	int
day	Day 1–31 (local city time)	int
hour	Hour 0–23 (local city time)	int
utc_timestamp	Unix time for start of hour (UTC)	int
segment_id	Replaced by OSM value "osm_way_id." Movement ID that maps to a specific road segment	text
start_junction_id	Replaced by OSM value "osm_start_node_id." Movement ID that maps to start intersection of traversal	text
end_junction_id	Replaced by OSM value "osm_end_node_id." Movement ID that maps to end intersection of traversal	text
osm_way_id	Corresponding OpenStreetMap Way ID for this segment. Note that one OpenStreetMap Way may contain multiple Movement segments	bigint
osm_start_node_id	Corresponding OpenStreetMap Node ID for this junction	bigint
osm_end_node_id	Corresponding OpenStreetMap Node ID for this junction	bigint
speed_mph_mean	Average speed of Uber vehicles on this road segment in mph	float
speed_mph_stdev	Standard deviation of speeds on this road segment in mph	float

for very large datasets, while deep learning models are often used to deal with big data. Though, in many cases you have to deal with some cities providing large amounts of data (i.e., for many years with a daily time resolution), and others providing few data with a different time range. Deep learning models are computationally expensive, often require the use of GPU clusters, and have high memory requirements that increase rapidly with dataset size, which could make the problem unmanageable. The development of scalable solutions capable of handling large amounts of data (i.e., implementing efficient traffic prediction for large geographical areas and time intervals), and able to update quickly, requires reducing the dimensional complexity of the dataset and using an approach that eliminates the criticality of the various techniques reported. For these reasons, in order to build a scalable solution we chose to implement both a Random Forest model and a gradient boosting model, using the XGBoost library [14], with a *multi:softprob* objective function (to get class probabilities as model's predictions). Model's outputs are probabilities for each traffic congestion related class (i.e., *low*, *medium* and *high speed*), which are suitable to show the prediction's confidence with a color gradient scale on the map. Tree-based models are particularly well suited to be used in this context, since they outperform deep learning approaches on tabular data [15]. The model works as a probabilistic classifier that generalizes the notion of classifiers. Thus, for a given $x \epsilon X$, it assigns probabilities to all $y \epsilon Y$, and probabilities sum to one. Standard classification can be obtained using the optimal decision rule $\hat{y} = \arg\max_y Pr(y = Y|X)$, where X

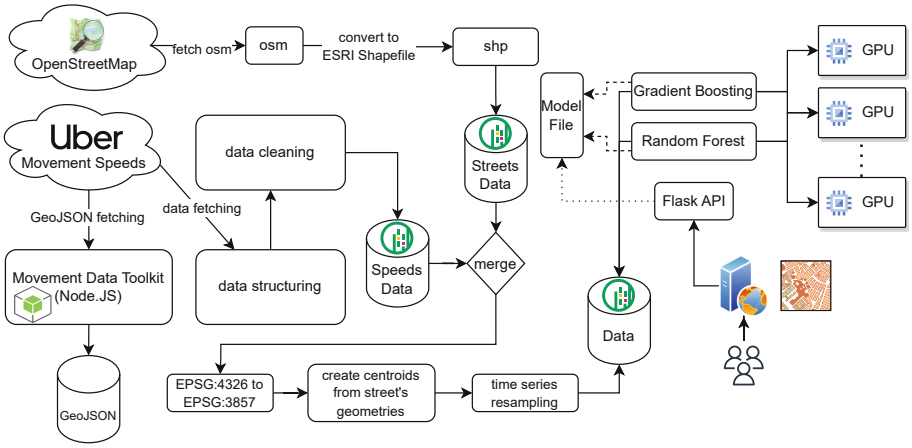


Fig. 1. General Architecture

and Y are respectively the features' set and the model's output. Model's training took place on a Nvidia Titan XP GPU, featuring 3840 CUDA Cores, 12 GB GDDR5X. XGBoost model has the advantage, over a Random Forest model, of being incrementally updatable, for example with traffic data from previously untracked areas. In this regard, an incremental scheme was implemented, to further train the model with new data without the need of training from scratch every time. At the end of training the model is saved to a file and can be loaded later for further training. In this way the proposed architecture is able to seamlessly scale up without the need of training from the beginning in case of new data. The model can automatically cope with class imbalance, by taking into account class weights if appropriate. At this regards, it was preferred not to implement strategies to mitigate class imbalance (e.g., SMOTE [16]), since such approaches are computationally expensive, increase the size of the dataset and consequently the training time, and most importantly add additional noise to the dataset. The described solution supports using parallel computing frameworks (e.g. Dask) to leverage multi-nodes and multi-GPUs for faster training.

5 Experiments and Discussion

Training was performed for various time slots (i.e., [1, 2, 3, 4, 6, 8, 12, 24] h). A *n-hour* time slot is defined as a time period spanning all the day with time slots of *n-hours* each. For example, 8-h includes time slots [00:00-08:00), [08:00-16:00), [16:00-00). Relevant training parameters for gradient boosting include learning rate (0.3), maximum number of discrete bins to bucket continuous features (256), grow policy (depthwise), L2 regularization term on weights (1), minimum loss

reduction required to make a further partition on a leaf node of the tree (0), maximum depth of a tree (6), maximum number of nodes to be added (6). Random Forest model was trained defining the fraction of randomly selected features used to train each tree (80%), learning rate (1), maximum depth (5), number of parallel trees (100), the fraction of the randomly selected training samples used to train each tree (80%). For each training, performance metrics were calculated for the best iteration number, and they are reported in Table 3. By averaging each model's accuracy, calculated for each time slot, we get an accuracy of 94.78%, with the XGBoost model, which generally outperforms other machine learning or deep learning approaches (e.g., CNN, RNN) [17–19], and 94.81% with the Random Forest model. Training times for each model highlight the efficiency of the model, which is able to scale easily with large amounts of real-time data. (31.1 s for 1H model, 13.5 s for 2H model, 10.4 s for 3H model, 7.06 s for 4H model, 5.32 s for 6H model, 4.18 s for 8H model, 3.05 s for 12H model, 1.74 s for 24H model). Classification metrics for the worst and best cases are reported in Table 4. Confusion matrixes for the worst and best cases are reported respectively in Fig. 2. In the worst and best cases, training times were respectively 31.1 s and 1.74 s, with the traffic dataset of the whole area of San Francisco (4,771,410 rows in the training set and 1,192,853 rows in the test set).

These metrics are reported only for the XGBoost model, since Random Forest showed comparable results. Features' importances, calculated for 1-h and 24-h models (XGBoost), show the features' impact on the model's output magnitude, with road segment's by far the most important feature). The importance was calculated using the cover strategy, thus calculating the average coverage of splits that use the feature, where coverage is defined as the number of samples affected by the split. Results indicate that *segment_id* has the biggest impact on the model's output, and *hour* is of course useless for the 24-h model (i.e., the bigger the model's time slots, the lesser the importance of time-dependent features). The impact on city traffic is thus affected more by the structure of arterial roads than by the time related features, which can exacerbate critical issues arising from bottlenecks generated by poorly sized roads. Among the temporal components, the greater importance of *hour* is highlighted, as opposed to *day* or *weekday*. In general, temporal features with fine-grained resolution (i.e., hour, minutes) have a greater impact on the traffic predictions, but models with such features are noisier and have worse performance in terms of accuracy. Despite its relative importance, *weekday* is an important feature that deserves consideration. In the present experiments, the relevance of this feature is limited, having considered one month, whereas for larger time intervals it is an indicator of the seasonality of traffic trends.

Table 3. Model's performance metrics XGBoost (Random Forest)

Accuracy Score	Balanced Accuracy Score	F1 Score	Geom. Mean Score	Cohen Kappa Score	Matth. Corr. Coeff.	Model
0.971 (0.971)	0.931 (0.931)	0.940 (0.940)	0.930 (0.930)	0.902 (0.903)	0.902 (0.903)	24H
0.961 (0.962)	0.936 (0.937)	0.937 (0.937)	0.934 (0.936)	0.880 (0.881)	0.880 (0.881)	12H
0.954 (0.955)	0.922 (0.922)	0.924 (0.925)	0.920 (0.921)	0.865 (0.866)	0.865 (0.866)	8H
0.948 (0.949)	0.917 (0.919)	0.917 (0.919)	0.915 (0.918)	0.849 (0.852)	0.849 (0.852)	6H
0.944 (0.945)	0.915 (0.915)	0.915 (0.917)	0.914 (0.914)	0.846 (0.848)	0.846 (0.848)	4H
0.941 (0.941)	0.909 (0.909)	0.913 (0.913)	0.908 (0.908)	0.842 (0.842)	0.842 (0.842)	3H
0.936 (0.936)	0.910 (0.909)	0.909 (0.909)	0.908 (0.907)	0.831 (0.832)	0.832 (0.832)	2H
0.926 (0.927)	0.902 (0.902)	0.900 (0.901)	0.900 (0.900)	0.813 (0.813)	0.813 (0.813)	1H

Table 4. Model's Classification report 1-h (24-h) (XGBoost)

	precision	recall	f1-score	support
low speed	0.94 (0.98)	0.96 (0.99)	0.95 (0.98)	901925 (63914)
medium speed	0.84 (0.92)	0.78 (0.90)	0.81 (0.91)	257554 (12414)
high speed	0.89 (0.95)	0.93 (0.91)	0.91 (0.93)	33374 (1457)
accuracy			0.92 (0.97)	1192853 (77785)
macro avg	0.89 (0.95)	0.89 (0.94)	0.89	1192853 (77785)
weighted avg	0.92 (0.97)	0.92 (0.97)	0.92	1192853 (77785)

6 Web Map

To visualize traffic predictions in an easy way, we developed a web application by converting the dataset with predictions in a GeoJSON format. GeoJSON is a text-based format that uses a simple data structure and is ideal both for accessing and creating geospatial data. The conversion process was automated with a Python script that reads the original dataset and builds the GeoJSON structure, for each date (i.e., day, month, year). The dataset was built using the features *day*, *hour*, *weekday*, *segment_id*, *geometry*, with three class labels (0: *low traffic*, 1: *medium traffic*, 2: *high traffic*). Predictions provide an array with [*y_prob*_0, *y_prob*_1, *y_prob*_2], where *prob_x* represents the probability of class x. Each traffic layer is represented as a GeoJSON *FeatureCollection*, with *LineString* geometries representing the roads. Speed related classes were converted to colors, *dark red* for low speed (*high traffic*), *red* for medium speed (*medium traffic*), *orange* for high speed (*low traffic*). For the purpose of visualizing traffic data on the map, the road's geometries were converted to the EPSG:4326 geodetic system. The web application is structured as an interactive map, built with OpenStreetMap and the Javascript library Leaflet, allowing to zoom and drag. The user is able to select a date and then the traffic data is asynchronously loaded and shown on the map, using a color for each road segment, depending on the traffic intensity. In this way, it is possible to give an insight into traffic conditions

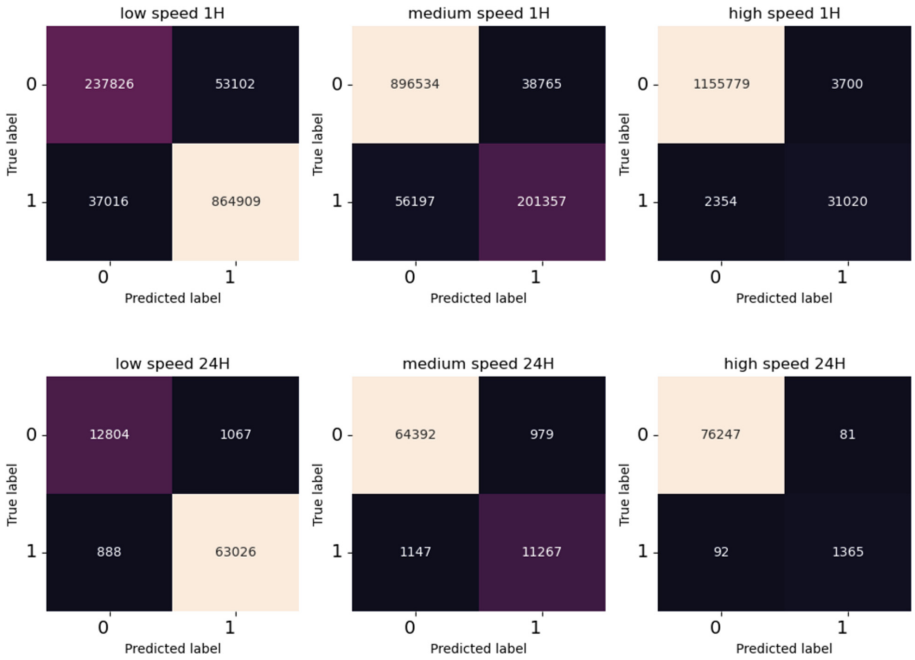


Fig. 2. Confusion Matrixes [1, 24 h] (XGBoost)

at first glance. Users can select and switch between dates and time resolutions with their respective select menus, and browse different traffic maps in a seamless way, by clicking on appropriate arrow buttons. Since large amounts of data can be handled with GeoJSON, depending on the portion of the displayed map, a spinning wheel is shown as the layers with traffic data are loaded, to warn the user to wait for the completion of asynchronous data loading. Clicking of each road segment will pop up the probabilities of each predicted class. Predictions can be obtained from a Flask REST API that exposes the traffic model, or are read from a batch generated file. The use of a traffic heatmap is a valuable aid in getting an immediate perception of traffic conditions, and allows for easy identification of traffic jams.

7 Conclusion

In this paper we introduced a solution for processing and managing traffic related data, with the goal of providing accurate predictions of city-related traffic intensities. In general, traffic predictions can be derived from statistical, machine learning or deep learning based models, but dealing with geographical features is often a complex task that prevents the model's architecture to scale up efficiently. Statistical approaches allow to identify traffic patterns at different time scales, and are easier to implement, though they are less accurate and are not best

suited to deal with multivariate data. Deep learning models are more complex, but they are difficult to handle when the size of the dataset grows exponentially, which is the case with real-time traffic data covering large geographical areas. With the goal of mitigating the curse of dimensionality, without reducing the information content of the traffic datasets, this paper implemented a strategy for creating a compact predictive model that can be trained on clusters of GPUs, even incrementally so as to update as new traffic data arrives (both historical and real-time traffic data can be used to generate valuable traffic related predictions with our platform). The proposed solution turns out to be easily scalable, upgradeable with new data to take into account larger geographical areas, and has an average accuracy of 94.7% (i.e., calculated by averaging the models' accuracies at various time slots). These results outperform traditional machine learning or deep learning approaches, with the clear advantage of constituting a streamlined and efficient solution that can be easily implemented at large scales with reasonable hardware resources. Cities can make use of this platform to curb critical traffic event issues, with a clear benefit from a pollution and energy-saving perspective, benefiting the productivity and economic development of urban settings. Future directions include taking into account external data (e.g., weather data, social media) and using transfer learning techniques to perform spatio-temporal predictions.

Acknowledgment. Traffic data used for this study were retrieved from Uber Movement, (c) 2023 Uber Technologies, Inc. This work is supported in part by the NSF project CNS-1932482.

References

1. Google Maps Platform (2023). https://developers.google.com/maps/documentation/javascript/trafficlayer. Accessed 25 June 2023
2. Waze (2023). https://support.google.com/waze/partners/answer/10618035?hl=en. Accessed 25 June 2023
3. HERE (2023). https://www.here.com/platform/traffic-solutions/real-time-traffic-information. Accessed 25 June 2023
4. ArcGIS (2023). https://developers.arcgis.com/rest/network/api-reference/traffic-service.htm. Accessed 25 June 2023
5. PTV (2023). https://www.ptvgroup.com/en/solutions/products/ptv-maps-data/traffic/real-time-traffic-data/. Accessed 25 June 2023
6. Jiang, J., Han, C., Zhao, W.X., Wang, J.: PDFormer: propagation delay-aware dynamic long-range transformer for traffic flow prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, no. 4, pp. 4365–4373 (2023). https://ojs.aaai.org/index.php/AAAI/article/view/25556
7. Huang, F., Yi, P., Wang, J., Li, M., Peng, J., Xiong, X.: A dynamical spatial-temporal graph neural network for traffic demand prediction. Inf. Sci. **594**, 286–304 (2022). https://doi.org/10.1016/j.ins.2022.02.031
8. Chen, R., Liang, C.-Y., Hong, W.-C., Gu, D.-X.: Forecasting holiday daily tourist flow based on seasonal support vector regression with adaptive genetic algorithm. Appl. Soft Comput. **26**, 435–443 (2015). https://doi.org/10.1016/j.asoc.2014.10.022

9. Johansson, U., Boström, H., Löfström, T., Linusson, H.: Regression conformal prediction with random forests. Mach. Learn. **97**(1), 155–176 (2014). https://doi.org/10.1007/s10994-014-5453-0
10. Uber Movement (2023). https://movement.uber.com. Accessed 25 June 2023
11. OpenStreetMap. OpenStreetMap API (2023). https://wiki.openstreetmap.org/wiki/API. Accessed 25 June 2023
12. Micci-Barreca, D.: A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems. SIGKDD Explor. Newsl. **3**(1), 27–32 (2001). https://doi.org/10.1145/507533.507538
13. Pargent, F., Pfisterer, F., Thomas, J., Bischl, B.: Regularized target encoding outperforms traditional methods in supervised machine learning with high cardinality features. Comput. Stat. **37**(5), 2671–2692 (2022). https://doi.org/10.1007/s00180-022-01207-6
14. Chen, T., Guestrin, C.: XGBoost: a scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2016, pp. 785–794. Association for Computing Machinery, New York (2016). https://doi.org/10.1145/2939672.2939785
15. Grinsztajn, L., Oyallon, E., Varoquaux, G.: Why do tree-based models still outperform deep learning on typical tabular data? In: Thirty-Sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (2022). https://openreview.net/forum?id=Fp7_phQszn
16. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: synthetic minority over-sampling technique. J. Artif. Int. Res. **16**(1), 321–357 (2002). https://doi.org/10.5555/1622407.1622416
17. Liu, Y., Wu, H.: Prediction of road traffic congestion based on random forest. In: 2017 10th International Symposium on Computational Intelligence and Design (ISCID), vol. 2, pp. 361–364 (2017). https://doi.org/10.1109/ISCID.2017.216
18. Kurniawan, J., Syahra, S.G., Dewa, C.K., Afiahayati: Traffic congestion detection: learning from CCTV monitoring images using convolutional neural network. Procedia Comput. Sci. **144**, 291–297 (2018). iNNS Conference on Big Data and Deep Learning. https://doi.org/10.1016/j.procs.2018.10.530
19. Ma, X., Yu, H., Wang, Y., Wang, Y.: Large-scale transportation network congestion evolution prediction using deep learning theory. PLOS ONE **10**(3), 1–17 (2015). https://doi.org/10.1371/journal.pone.0119044

Fingertip Air-Writing with Ambient Light

Hao Liu, Hanting Ye, Xiangxie Zhang, Jie Yang, and Qing Wang(✉)

Delft University of Technology, Delft, The Netherlands
{h.liu-8,h.ye-1,j.yang-3,qing.wang}@tudelft.nl

Abstract. Contact-free interaction with public devices could become popular. To achieve this purpose, state-of-the-art methods mainly use cameras to capture mid-air hand gestures, which are power-hungry and can raise privacy issues. In this work, we design **LightDigit**, a system for fingertip air-writing of digits with ambient light and photodiodes, to enable contact-free interactions with public devices. The key enabler is detecting and interpreting dynamic shadows on photodiodes introduced by fingertip movements. We design an embedded deep learning model **LightConvRNN** –customized ConvRNN with attention pooling– to capture spatial and temporal patterns in the dynamic shadows. We evaluate LightDigit through extensive experiments under different light conditions. Evaluation results show that our model can achieve an accuracy of up to 98%. Through model compression, the model size is reduced by 92% with less than a 5% drop in the performance. LightDigit is robust to ambient light positions (60°) and ambient light intensity (5000 lx).

Keywords: Air-writing · Ambient light · Embedded AI

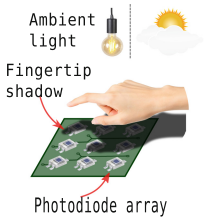
1 Introduction

Touch-free interaction with public devices might play an important role in our future lives, because it provides a means for people to avoid touching public devices shared among many people. This could reduce the potential spread of virus. Recently, a mid-air virtual touchscreen has been created to help people avoid touching public screens [1]. This is done by deploying a *camera* to capture and recognize mid-air hand gestures such as sliding up and sliding down. Although promising, using cameras leads to high energy consumption and raises privacy issues. Also, the Analog Devices company introduces a new generation of infrared-based dynamic optical sensors to sense a broader range of gestures [16]. This sensor can recognize simple gestures such as swipes and mid-air click by modulating several infrared LEDs and detecting the reflected infrared light.

Meanwhile, the ubiquitous visible light is gaining significant interest as a medium to sense/connect things (objects). Besides *active* visible light systems where LEDs are modulated to send information and objects leverage light sensors to receive that information, researchers are also studying *passive* visible light systems for various objects sensing [12,13,19,20].

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 179–191, 2024.
https://doi.org/10.1007/978-3-031-63992-0_11

(a) Using toothpicks during the pandemic



(b) The LightDigit system

Fig. 1. The motivation (a) and illustration (b) of our LightDigit system, a device-free fingertip air-writing system with ambient light and simple photodiodes.

Motivated by these rising fields, in this paper, we design *LightDigit*, an embedded and real-time system to enable touch-free fingertip writing of the digits 0–9 with *ambient light*. As illustrated in Fig. 1, with LightDigit people can use a finger to write digits in front of but not touching the photodiodes. The key enabler is the detection and embedded-learning-based decoding of the dynamic shadow cast by the fingertip movement.

Challenges. To achieve fingertip air-writing of digits with only ambient light and simple photodiode, several challenges must be tackled.

- *Human impact: Different people usually write certain digits differently.* Taking the digit '7' as an example, it is commonly written without an additional horizontal line crossing the middle, as depicted in Fig. 2. However, many people have alternative writing styles. Another example is '4': people typically stop the horizontal line at the vertical line, whilst others make them cross. Such systematical features of handwritten digits can also be observed from popular handwritten digit datasets collected from different regions, e.g., MNIST/EMNIST [6,7] (US), [3] (Europe), CASIA/HIT-OR3C [14,21] (Asia), etc. These regional variations in handwritten digits, as well as *writing speed* and *sizes of handwritten digits*, bring us challenges in recognizing air-writing digits with ambient light.
- *Environment impact: The shadow area –detected and interpreted for digit recognition– is affected by the intensity levels of ambient light, varying angles of light incidence, and passersby.* With different ambient light like the illumination and sunlight, the position of the ambient light source can be different, leading to different shadow angles. Even with the same shadow angles, the various heights and sizes of fingertips will also generate different shadow areas. Besides, the system's stability may be weakened by the fact that many light sources cast different shadows; passersby also affect the shadows.
- *Device impacts: Embedded devices are resource-constrained.* To handle the complex digit recognition task, deep learning models usually have a significant amount of parameters, which do not fit in embedded devices that have limited computing and storage resources. It is challenging to design light-weighted deep learning models that can run in real-time on resource-limited embedded devices with reduced inference time and memory usage.

Fig. 2. Regional digit writing styles: *left)* mostly adopted in western countries; *right)* mostly adopted in Asian countries.

Our Contributions. To the best of our knowledge, LightDigit is the first embedded system that exploits deep learning models in resource-constrained devices for *real-time* recognition of air-writing digits with light. Our work builds the first-of-its-kind ambient light dataset of air-writing digits and investigates state-of-the-art model compression techniques for efficient model inference on embedded devices. We summarize our contributions as follows:

- We propose, design, and prototype the LightDigit system. It is the first embedded system that enables fingertip air-writing of digits with ubiquitous ambient light and simple photodiodes. To improve the system's robustness, we also propose a calibration method to actively adapt to the shadow area variations caused by changes in ambient light and fingertip height.
- We analyze the attributes of air-written digits and build the first *ambient light dataset of air-written digits*, a dataset including time-series information. Traditional data-gathering methods need to recruit hundreds or thousands of volunteers. To avoid such time-consuming and labor-intensive processes, we analyze the properties of air-written digits to create our own training dataset and evaluate it by existing datasets. *Our LightDigit dataset has 20880 samples of air-writing digits, which represent 174 different types of the ten digits 0–9.* Due to the space limit, please refer to [10] for the dataset details.
- We thoroughly evaluate our system in different experimental scenarios and progressively construct the LightDigit system with the best performance in terms of accuracy, inference time, and the number of parameters. The testbed evaluation results show that the proposed embedded deep learning models are effective, with LightConvRNN achieving 98% accuracy in the Intra-Subjects scenario. With knowledge distillation, we can reduce its model size by more than 92% while keeping less than a 5% reduction in Accuracy. We also demonstrate that LightDigit is robust to ambient light positions (e.g., 60°) and different ambient light intensity levels (up to 5000 lx).

2 System Design

2.1 System Architecture

We design LightDigit as an embedded sensing system to enable touch-free fingertip writing of the digits 0–9 with ambient light (unmodulated), such as indoor illuminating light and sunlight. The overall system mainly has three components (cf. Fig. 3): *moving fingertip*, *photodiode (PD) array*, and *embedded deep learning algorithms*. The key enabler is the detection and interpretation of the dynamic shadow brought by the fingertip movements when a person uses one of his/her fingers to write digits in the air (on the top of the PD array).

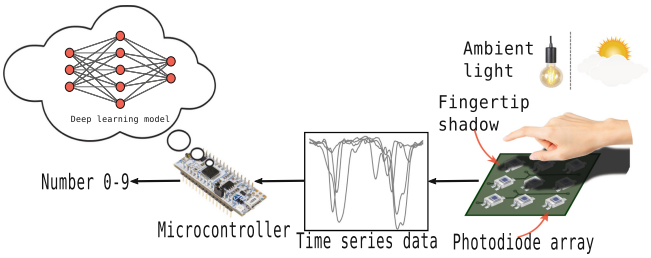


Fig. 3. System architecture.

Moving fingertip. It is the object whose trajectory will be sensed directly. When drawing a digit with finger in the front of the PD array, the movement of the finger will reflect the surrounding ambient light towards the PD array. The corresponding dynamic shadow, implicitly carrying the shape of the digit, is cast on the PD array.

Photodiode (PD) array. We use a group of PDs to detect dynamic shadows. The PDs are sampled sequentially with a fixed frequency, and the sampled data is sent to the embedded DL algorithm as the input for recognizing the digit.

Embedded deep learning algorithm. In this paper, we design an embedded deep learning algorithm LightConvRNN –customized ConvRNN with Attention– to recognize the air-written digits. LightConvRNN can be *trained and run on embedded devices* such as Raspberry Pi or BeagleBone AI and Black. We also propose a method to pre-process the captured data. To train the model, we built our own dataset LightDigit. Due to the space limit, please refer to our technical report [10] for the details of the LightDigit dataset.

2.2 Robustness Design

Complicated location distribution of ambient light sources and dynamic changes in handwriting height greatly impact the distribution of shadow areas, which also raises great challenges in recognizing handwritten digits. There is an intuitive method which we can calculate the angle from the light source to the sensing PD array by comparing the output values of the PDs from different columns or rows of PDs. However, the influence from the different heights of fingertips is not taken into account. For example, when there is a 45° illumination light for the sensing area, the shadow area on the sensing area varies according to the different heights of the fingertip to the sensing area. Thus, we propose an adaptive shadow recognition method to expand the original sensing area and automatically select a new sensing area.

We define two areas: (i) *writing area*, perpendicularly to which a person moves his/her finger to write the digits, and the M PDs contained in the writing area are marked as PD_1, PD_2, $\cdots$, PD_M. (ii) *sensing area,* that host M PDs from where the system will capture the shadow for further processing (Refer to Fig. 4 for details). The available writing area was also considered to be an important factor in this regard.

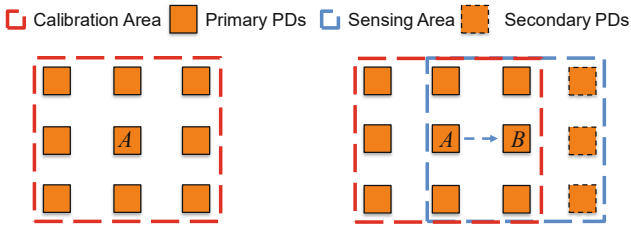


Fig. 4. Proposed calibration for better robustness.

The adaptive recognition method consists of four steps:

(1) *Reset calibration area.* We first need to set an area as the calibration area. As shown in Fig. 4, we mark an area as the calibration area, which contains M primary PDs that can output M channel values. Note that we only selected M PDs as the primary PDs. If more primary PDs are picked for the calibration area, calibration mode can provide a higher resolution in shadow projection.

(2) *Ambient light measurement.* Since the indoor light intensity does not change much in a short time, we first measure the ambient light intensity through primary PD i under the current environment, denoted as $p_i, i \in \{1, 2, \cdots, M\}$.

(3) *Handwriting position measurement.* Users need to put a fingertip on the center PD A of the calibration area. According to the actual light source distribution and the position distribution of the user's handwriting, the new output values of the M PDs are different, denoted as $q_i, i \in \{1, 2, \cdots, M\}$.

(4) *Shadow area determination.* Through the operation of step 1 to step 3, we can calculate the percentage change of the output value of each PD caused by the user operation, denoted as $\alpha_i = \frac{|p_i - q_i|}{p_i}, i \in \{1, 2, \cdots, M\}$. The largest α_i means that the shadow of the finger almost falls in this area, and the shadow of subsequent writing of numbers falls in the direction from the center PD of the calibration area to PD i. As shown in Fig. 4, we can obtain shadow destination $B = \arg\max_{i \in \{1,2,\cdots,M\}} \alpha_i$, and then activate the secondary PDs in the corresponding sensing area on the extension line of the shadow direction $\overrightarrow{AB}$ to obtain the correct handwritten digital shadow data. If $A = B$, the shadow is completely projected on the calibration area. Thus, we regard the calibration area as the sensing area.

3 Embedded Deep Learning Algorithm

We continue to introduce the embedded Deep Learning (DL) algorithm designed for the recognition of the air-writing digits. Our embedded DL algorithm takes as input the pre-processed data and then classifies the input into one of the ten digits 0–9. The key requirement for an effective embedded DL model is

the ability to capture both the spatial pattern among the channels and the temporal pattern across different time steps while accounting for the variations of writing styles within and between subjects. To this end, we design a customized architecture LightConvRNN –ConvRNN with Attention– for better control of the learning process for embedded machine learning. Next, we first present the data preprocessing and then describe the proposed embedded DL algorithm.

3.1 Data Preprocessing

We denote the raw data stream from channel/PD m as a vector $\boldsymbol{x}_m = [x_{m,1}, x_{m,2}, \cdots, x_{m,L}]$. Due to the noise interference and the limited on-device computing capability, $\boldsymbol{x}_m$ cannot be directly fed to the embedded deep learning models. Thus, we propose a procedure to preprocess the raw data.

Step 1) Outliers reset. The sampled abnormal value appears in $\boldsymbol{x}_m$ when the user unconsciously touches the sensing area with fingertips and causes the PDs array to move violently. Thus, we take the values at 99^{th} and 1^{st} percentile of the data $\boldsymbol{x}_m$ as the upper limit x_U and lower limit x_L, respectively. The filtered data $x_{m,l}$, $\forall l \in \{1, 2, \cdots, L\}$ can then be calculated as

$$x_{m,l} = \begin{cases} x_U, & x_{m,l} > x_U \\ x_L, & x_{m,l} < x_L \end{cases}. \tag{1}$$

Step 2) Data smoothing. There are many methods to reduce the output fluctuates caused by noise interference, including low-pass filters, high-pass filters, and Kalman filters [11] [5]. It is not suitable to use high-pass and low-pass filters in our case since the characteristics of the visible light signal we collect are distributed in various frequency bands. Also, Kalman filters can filter and smooth the data and predict the next data value. However, Kalman filters are overpowerful. To solve this issue, we deploy the lightweight Savitzk-Golay filter [17] that is designed for smoothing the data without distorting the signal tendency in our system. The filtered data $\tilde{\boldsymbol{x}}_{m,l}$ can be calculated as $\tilde{x}_{m,l} = \sum_{i=-w}^{+w} k_i x_{l+i}$, where w represents the window size, and k_i is the smoothing coefficient.

Step 3) Signal synchronization and time series compression. To reduce the computational overhead in embedded devices, we use energy detection to synchronously extract the effective shadow signal segment from the data stream $\tilde{\boldsymbol{x}}_{m,l}$. Firstly, we use *minmaxscaler* tool to obtain normalized $\hat{\boldsymbol{x}}_{m,l}$. Next, we set a threshold to extract the action signal $\bar{\boldsymbol{x}}_m = [\hat{x}_{m,a}, \hat{x}_{m,a+1}, \cdots, \hat{x}_{m,b}]$, where the starting and ending index of action signal is calculated as $a = \arg\min_l (x_{m,l} < \gamma)$ and $b = \arg\max_l (x_{m,l} < \gamma)$, respectively. The threshold is set as $\gamma = 0.7 x_{max}$, where $x_{max} = \max \hat{\boldsymbol{x}}_{m,l}$. The difference $(b - a)$ may vary as channel m changes. Thus, the length of processed data is shortened to the fixed number of time steps N by using interpolation to make the data better adapt to our deep learning models, denoted as $\bar{\boldsymbol{x}}_m = [\hat{x}_{m,1}, \hat{x}_{m,2}, \cdots, \hat{x}_{m,N}]$.

Finally, denoting the number of channels as M and integrating the data of channel $m \in \{1, 2, \cdots, M\}$, the input data format to the learning models is $D_{M \times N} \triangleq [\bar{\boldsymbol{x}}_1, \bar{\boldsymbol{x}}_2, \cdots, \bar{\boldsymbol{x}}_M]$.

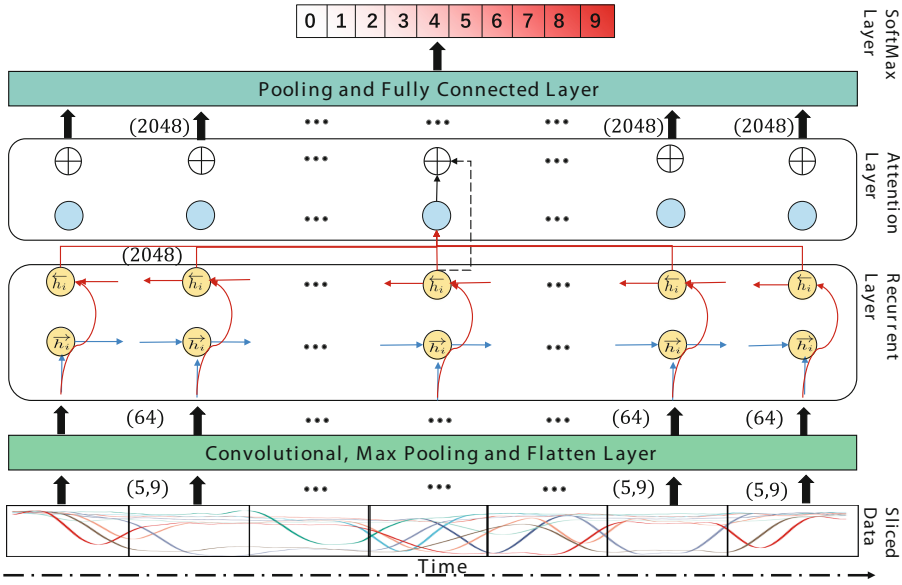


Fig. 5. The architecture of the proposed LightConvRNN: a customized *ConvRNN with Attention* model.

3.2 LightConvRNN: Customized ConvRNN with Attention

We explicitly model the spatial and temporal patterns to better control the learning process. To do so, we first segment the multi-dimensional time series data $D_{M\times N}$ into k smaller chunks $k\times D_{M\times \frac{N}{k}}$. Our network starts with a convolutional layer on each of the chunks, so as to explicitly model the spatial pattern among channels. Note that it also helps to capture short-term temporal patterns. Afterwards, we apply the pooling operator for information aggregation and obtain a new data representation of size $D_{M'\times k}$. We then build bi-directional recurrent layers, i.e., bi-directional GRU [4], to capture longer-term temporal patterns. GRU [9] uses the gating mechanism to balance the information flow from previous time steps in encoding the time series, and the bi-directional GRU [18] contains a forward and a backward layer to encode information flow from both directions of the time series for better information preservation. Even with the gating mechanism, it is still challenging for GRU to deal with long range dependencies in the times series. We therefore, add the attention pooling mechanism to help the model preserve relevant information from all time steps: the output is a weighted sum, i.e., a selective summary, of the bi-directional GRU's hidden units at all time steps. Output of the attention pooling layer is then fed to the fully-connected layers and finally to the softmax layer to generate the classification result, a vector of size ten, representing the predicted probability of the digit being 0–9. The overall model structure is depicted in Fig. 5.

To reduce the complexity of LightConvRNN, we compress it using knowledge distillation [2], a technique that is also used by state-of-the-art studies on visible light sensing such as [15]. Knowledge distillation trains a large model (i.e., the teacher model) and then transfers the learned patterns to a smaller model (i.e., the student model), by training the smaller model to learn from both the original labels and the output from the larger model (referred to as pseudo labels, as in contrast to the original labels).

4 Performance Evaluation

4.1 Testbed Implementation

The prototype of LightDigit is shown in Fig. 6. We use Raspberry Pi 4 as our controller due to its compatibility with deep learning platforms like TensorflowLite, Tensorflow, Keras, and Pytorch. In addition, OPT101 is selected since it is an integrated combination of PDs and trans-impedance amplifier on a single chip, which is very stable to sensing light. The sensing board consists of 16 OPT101 PDs with fixed equal intervals, which is shown in Fig. 6. Note that we only use M=9 out of the 16 PDs for detecting the shadow. The 9 PDs are dynamically selected during the calibration, cf. Sect. 2.2 for the details. The size of the square sensing area is set to 41 mm × 41 mm by the tradeoff between a convenient writing area for people and fewer PDs. We use two Analog-to-Digital Converters (ADCs) to sample the 16 PDs. In our implementation, the sampling frequency of the PD is fixed to 100 Hz because we find that it is already enough for handling the fastest digit writing (<0.5 s) in our test.

4.2 Experimental Setup

Testing Environments. The data is collected in two typical indoor environments: a student apartment and a university lab. We use a reading lamp and fluorescent lamps as the ambient light source (the light is not modulated). The average ambient light intensities at the sensing board can be changed between

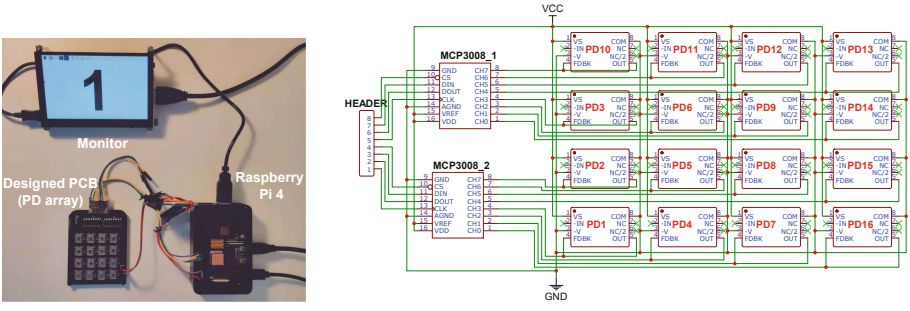


Fig. 6. LightDigit prototype: *left)* hardware, *right)* schematic.

Table 1. Performance of our LightConvRNN model (including comparisons with state-of-the-art works). Results are averaged over ten runs in each configuration.

Scenarios	RNN [1,8]			ConvRNN [15]			LightConvRNN		
	ACC	#Param	Itime(s)	ACC	#Param	Itime(s)	ACC	#Param	Itime(s)
Intra-Subjects	85.9%	808458	0.5531	91.4%	1369994	0.4494	97.9%	1754059	0.5147
Inter-Subjects	77.2%	808458	0.5569	79.2%	1369994	0.4406	90.9%	1754059	0.5189

143 lx and 5000 lx. The incident angle of the light emitted by the lamps to the sensing board can be adjusted between 0° and 60°.

Hyperparameter Setting. Optimal hyperparameters are searched for LightConvRNN in each scenario. We empirically set optimal parameters based on a hold-out validation set that contains 10% of the test data on Intra-Subjects and Inter-Subjects scenarios. For the learning rate, we apply a grid search in $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$. The number of filters in CNN and the number of hidden units in GRU are explored from the set {8, 16, 32, 64, 128, 256} and the set {32, 64, 128, 256, 512, 1024}, respectively. Model training is performed using Adagrad with mini-batches of size 32. The dropout rate is selected from the set {0.0, 0.1, 0.2, 0.3}. Model compression is performed as follows. After the optimal hyperparameters are found, we reduce the hidden units in the GRU layer.

We consider two *test scenarios* to evaluate our system: (1) **Intra-Subjects**: When LightDigit is only used by a fixed population, such as in schools, the capacity to recognize similar data is crucial. Thus, we shuffle each participant's data in the real dataset and divide it into training (80%) and test (20%). The training set is enriched with our simulated dataset. The training set is enriched with our simulated dataset. (2) **Inter-Subjects**: If LightDigit is deployed in a public place, anyone from various geographic regions has the ability to use it. In this instance, the system's ability to recognize data from unknown domains, i.e. its ability to generalize, is crucial. Using cross-validations, we utilized 2/3 of the real data, or 18 individuals, for training and the remaining 6 for testing.

Metrics. We use the metrics Accuracy (ACC), number of parameters (Param), and inference time (Itime). ACC quantifies the proportion of correct categorization across all instances of data. Param measures the model size, a key component in deciding the amount of storage space in an embedded device. The model's inference time running on the embedded device is impacted by the number of parameters, input size, and CPU operating modes (serial/parallel) (Table 1).

4.3 Evaluation Results

Proposed Model vs. State of the Art. We first compare LightConvRNN with the state-of-the-art RNN [1,8] and ConvRNN [15]. We evaluate these models on our LightDigit dataset. Table 2 demonstrates their performance and model size in both scenarios. Our LightConvRNN surpassed RNN and ConvRNN by 5% to 15% in terms of accuracy. In terms of model size, LightConvRNN is

Table 2. Performance under different light angles.

Scenarios	Angle	Accuracy	
		With calibration	Without calibration
Intra-Subjects	0°	94.67%	95.91%
	30°	85.12%	15.38%
	60°	75.14%	8.65%
Inter-Subjects	0°	90.63%	90.92%
	30°	84.09%	24.79%
	60°	69.89%	10.58%

approximately 54% larger than RNN and 22% larger than ConvRNN. As for inference time, ConvRNN and LightConvRNN have comparable inference times since they require a similar amount of serial processing. We can observe that the inference time of RNN is longer than ConvRNN and LightConvRNN because the architecture of ConvRNN separates input data into smaller chunks, hence reducing the number of serial operations. LightConvRNN requires more time for inference than ConvRNN since it has an additional attention layer.

We can observe that the optimal performance is under the Intra-Subjects scenario, where LightConvRNN achieves almost 98% accuracy. Performance in the Inter-Subjects scenario is lower: 91%. The result confirms the larger variation of air-writing styles across subjects as compared to that within the same subject, thus posing a bigger challenge for digit recognition. In the Inter-Subjects scenario, our system achieves over 90% accuracy with LightConvRNN. This result demonstrates the effectiveness of our system in recognizing air-writing digits, especially by subjects who have been historically seen by the system.

Shadow Shifting. We compare the performance of LightConvRNN on calibrated and non-calibrated data with a fixed fingertip height between 0.5 cm and 1.5 cm. Results are given in Table 2 on the three incident light angles (0°, 30°, and 60°). We observe that LightDigit performance under 0° incident angle is equally well in calibrated and non-calibrated modes across all scenarios (less than 2% difference) due to the overlapping drawing and sensing areas. When it comes to the 30° incident angle, the performance in non-calibration mode shows a heavy drop from over 90% accuracy to less than 30% in all scenarios. By contrast, LightDigit performance in the calibration mode remains high: 85% and 84% in Intra-Subjects and Inter-Subjects scenarios, respectively. When the incident angle is 60°, the system performs poorly in non-calibration mode with less than 20% accuracy in all scenarios, while keeping around or over 70% accuracy in the different scenarios. Those results demonstrate the strong effectiveness of the proposed calibration for robustness improvement. We note that there is still a performance drop in high-incident angle situations. It happens for several reasons: fingertip shadow under high incident angles can cover multiple PDs, as compared to an individual PD in normal situations; slight changes of the ver-

Table 3. Accuracy under different ambient light intensities.

Light intensities (lux)	500	1000	5000
Accuracy	91%	91.5%	99.1%

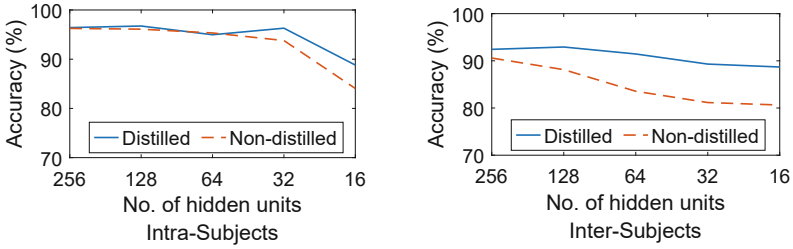


Fig. 7. Distilled vs. non-distilled LightConvRNN. Number of parameters under different number of hidden units are 1754k, 484k, 144k, 48k, 18k, 8k, respectively.

tical distance between the fingertip and the sensor – which frequently happens for complex digits like '6' and '8' – can easily move the shadow from one PD to another. We leave it to future work to tackle those issues.

Different Ambient Light Intensities. To test the robustness of our system under different ambient light intensities, we place an indoor light source directly above our sensing board and vary the light source's height to achieve different light intensities perceived by the PDs. The evaluation results are given in Table 3. As the light intensity increases, we can see that the accuracy rate remains above 90%. This demonstrates the stability of our system up to an ambient light intensity of 5000 lx. In addition, it can be observed that the accuracy is higher at 5000 lx than under other light intensities. This is primarily because, at 5000 lx, the light source is very close to the system. As a result, the projection of the finger is deeper and more compact, resulting in cleaner signals, and thus, a slightly higher accuracy.

Model Compression. To make our model run on embedded devices, we use knowledge distillation to reduce the model size and inference time. To evaluate the performance, we compare the distilled model (i.e., student model) with the non-distilled model that has the same number of parameters trained from scratch. Figure 7 shows the result for LightConvRNN, with the number of hidden units decreasing 256 (Intra- and Inter-Subjects) to 16, halved in each step; note that the teacher model has 512 hidden units in the Intra- and Inter-Subjects scenarios. We observe a similar result that distilled LightConvRNN consistently outperforms non-distilled LightConvRNN. In the Intra- and Inter-Subjects scenarios, the number of hidden units can be reduced to 128 with performance kept almost unchanged, amounting to 92.27% reduction of parameters. In all scenarios, the number of hidden units can be reduced to 32, while keeping less than 5% drop in performance. This amounts to 98.95% (Intra-Subjects and

Table 4. Inference time for different numbers of hidden units in LightConvRNN.

Number of units	512	256	128	64	32	16
Inference time (s)	0.518	0.508	0.497	0.495	0.495	0.494

Inter-Subjects) parameter reduction, respectively. Table 4 shows the changes in inference time that accompany changes in the model parameters. In summary, we have observed a strong effect of knowledge distillation in compressing the model. This result indicates that hidden units in the recurrent layers of LightConvRNN contain a large amount of redundant information that can be removed without affecting the performance.

5 Conclusion

We have designed LightDigit, an air-writing digit recognition system by exploiting the dynamic shadow caused by the fingertip movement with a low-cost PD array and ambient light. We offer a calibration method and utilize data processing techniques to enhance the system's robustness. To execute the recognition, we designed an embedded deep-learning model LightConvRNN. We created a reliable and trustworthy LightDigit dataset for model training. We prototyped LightDigit and validated that it can achieve superior performance. We envision it can motivate more follow-up research on passive sensing with ambient light.

References

1. Alam, M.S., et al: Trajectory-based air-writing recognition using deep neural network and depth sensor. Sensors (2020)
2. Ba, L.J., Caruana, R.: Do deep nets really need to be deep? arXiv:1312.6184 (2013)
3. van Breukelen, M., et al: Handwritten digit recognition by combined classifiers. Kybernetika (1998)
4. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv:1412.3555 (2014)
5. Cipra, T., Romera, R.: Robust kalman filter and its application in time series analysis. Kybernetika (1991)
6. Cohen, G., Afshar, S., Tapson, J., Van Schaik, A.: EMNIST: extending MNIST to handwritten letters. In: IEEE IJCNN (2017)
7. Deng, L.: The MNIST database of handwritten digit images for machine learning research. IEEE Signal Process. Mag. (2012)
8. Duan, H., Huang, M., Yang, Y., Hao, J., Chen, L.: Ambient light based hand gesture recognition enabled by recurrent neural network. IEEE Access (2020)
9. Graves, A., et al: Speech recognition with deep RNN. In: IEEE ICASSP (2013)
10. Hao, L.: LightDigit Dataset (2021). https://www.dropbox.com/s/lbllt66mnunj22g
11. Li, Q., Li, R., Ji, K., Dai, W.: Kalman filter and its application. In: ICINIS (2015)
12. Li, T., An, C., Tian, Z., Campbell, A.T., Zhou, X.: Human sensing using visible light communication. In: ACM MobiCom (2015)

13. Li, T., Zhou, X.: Battery-free eye tracker on glasses. In: ACM MobiCom (2018)
14. Liu, C.L., Yin, F., Wang, D.H., Wang, Q.F.: CASIA online and offline Chinese handwriting databases. In: IEEE ICDAR (2011)
15. Liu, H., Ye, H., Yang, J., Wang, Q.: Through-screen visible light sensing empowered by embedded deep learning. In: AIChallengeIoT Workshop (2021)
16. Maxim integrated: Max25405, April 2021. https://datasheets.maximintegrated.com/en/ds/MAX25405EVKIT.pdf
17. Menon, S.V., Seelamantula, C.S.: Robust Savitzky-Golay filters. In: ICDSP (2014)
18. Schuster, M., et al: Bidirectional recurrent neural networks. IEEE TSP (1997)
19. Yang, Y., Hao, J., Luo, J., Pan, S.J.: CeilingSee: device-free occupancy inference through lighting infrastructure based led sensing. In: IEEE PerCom (2017)
20. Zhang, C., Tabor, J., Zhang, J., Zhang, X.: Extending mobile interaction through near-field visible light sensing. In: ACM MobiCom (2015)
21. Zhou, S., et al: An empirical evaluation on HIT-OR3C database. In: ICDAR (2011)

Global Optimal PnP Algorithm with Reverse Projection Error

Xuhao Jin(✉)

North China University of Technology, Beijing 100043, China
jxh2453020228@163.com

Abstract. A non-iterative global optimal PnP (perspective-n-point) algorithm with time complexity O(n) is proposed. The basic idea is to use camera internal parameters to reverse project 2D points into space, use the relationship between 3D points in world coordinates and 2D points to transform PnP problem into minimization problem, and use non-unit quaternion parameterization rotation matrix to transform PnP problems into unconstrained minimization problem, and finally solve the polynomial equation system composed of its first-order partial derivative. And the proposed method can deal with the minimization problem of PnP problem, P3P problem. Experimental results show that the proposed method can accurately process the all configuration of 3D points, and compared with the state-of-the-art PnP algorithm, the proposed method can provide comparable or better accuracy, and the computational efficiency is higher.

Keywords: PnP algorithm · Global optimal · Reverse projection · Non-unit quaternion · Minimization

1 Introduction

Pose estimation is a fundamental problem in computer vision, with wide applications in areas such as navigation [1], robotics [2], and virtual reality [3]. Pose refers to the combination of position and orientation, and pose estimation aims to find the transformation relationship between the target coordinate system and the world coordinate system. This transformation is typically represented by a rotation matrix R (a 3×3 matrix satisfying orthogonality constraints $RR^T = I$ and determinant constraints $\det(R) = 1$) and a translation matrix T. The most commonly used sensors for pose estimation include IMU (Inertial Measurement Unit), LiDAR, and cameras. IMU can provide high-frequency motion information but suffers from significant drift over time. LiDAR can effectively detect the surrounding environment but is expensive and sensitive to weather conditions. Cameras, on the other hand, have the advantages of low cost and rich sensing information, making visual-based pose estimation highly advantageous.

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 192–203, 2024.
https://doi.org/10.1007/978-3-031-63992-0_12

Pose estimation is a core problem in many theoretical research and practical applications. For example, in the front-end visual odometry of a visual SLAM system [4], the camera is used as the sole or primary sensor. The feature-based approach is employed, where feature points are first extracted from the images, then feature matching is performed to find corresponding features between two images, and finally, the camera pose is estimated using the matched feature points. There are three methods for camera pose estimation based on feature points: 1) 2D-2D: When the camera is monocular, only 2D point coordinates are available. The camera motion is estimated by solving the epipolar geometry problem using two sets of matched 2D points [5]. 2) 3D-3D: When the camera is binocular or RGB-D, or when distance information is obtained through some means, the camera motion is estimated based on two sets of 3D points using the ICP algorithm [6]. 3) 3D-2D: When both the 3D points and their 2D projections on the image are known, the camera pose is estimated by solving the PnP problem. In this case, a minimum of three point correspondences is required to estimate the camera motion.

Perspective-n-Point (PnP) problem for target pose estimation based on feature points. In this problem, given the camera intrinsic parameters, the coordinates of n ($n \geq 3$) feature points in the target coordinate system (often referred to as the world coordinate system) and their corresponding pixel coordinates in the image pixel coordinate system, we aim to compute the target coordinates in the camera coordinate system and thus determine the transformation between the camera coordinate system and the world coordinate system using the perspective projection relationship.

In this paper, we propose a non-iterative PnP algorithm. Our basic idea is to use the perspective projection relationship to reverse project 2D points into 3D points (3D points in the camera coordinate system). At this point, the depth of the reverse projected 3D points is unknown. However, the depth can be represented by the unknown rotation matrix and its translation, as well as the known 3D point coordinates in the world coordinate system. Due to the presence of noise, the 3D points in the two coordinate systems are not exactly equal. We exploit this error to transform the PnP problem into a minimization problem, which can be solved using the method of Gröbner bases.

Our method has the following advantages: (1) It uses non-unit quaternion to represent the rotation matrix, which eliminates the constraints on the rotation matrix. As a result, the PnP problem is formulated as an unconstrained optimization problem. Compared to the polynomials formed by unit quaternion with constraints, our method is simpler to solve when dealing with polynomials. (2) Unlike iterative methods that may converge to local optima, our method can obtain the global optimum. (3) It can handle the minimization problem in PnP (P3P). (4) Unlike OPnP and UpnP that return multiple solutions, our method only returns one solution.

2 Related Work

The PnP algorithm has been widely used in fields such as robot localization and virtual reality. Over the past few decades, researchers have proposed many methods to solve the PnP problem more accurately and efficiently. These methods can be roughly divided into iterative methods and non-iterative methods.

Iterative methods transform the PnP problem into a nonlinear least squares problem, typically using reprojection error as the error function, and solve it using iterative optimization methods such as Gauss-Newton or Levenberg-Marquardt [7]. Compared to non-iterative methods, iterative methods achieve higher accuracy but require longer computation time and better initial values, otherwise they may converge to local minima. Lu et al. [8] proposed a method that uses 3D error as the error function and alternately optimizes the rotation matrix and translation matrix T to enhance the orthogonality of the rotation matrix. Although it converges quickly, it still suffers from the problem of getting stuck in local minima. In this paper, a non-iterative method is used, so only a brief description of the iterative methods is provided here.

Direct Linear Transform (DLT) [9,10] is a traditional approach in non-iterative PnP algorithms. The DLT method solves the pose by exploiting the linear relationship between at least six sets of corresponding points. However, in the solution process, the pose is directly treated as twelve unknowns, ignoring the constraints between them (the rotation matrix R needs to satisfy $RR^T = I$ and $\det(R) = 1$). The rotation matrix obtained using DLT, denoted as R, is a general 3×3 matrix that may not satisfy the constraints of a rotation matrix, resulting in lower accuracy. Quan et al. [11] proposed a linear solution to the PnP problem, which can obtain a unique solution when n = 4 or 5. However, it is still not accurate enough when n is small, and it becomes time-consuming when n is large.

To solve these problems, Lepetit et al. [12] proposed EPnP, which uses four control points to represent the coordinates of 3D points in world coordinates. By solving for the coordinates of the four control points in camera coordinates, the coordinates of other 3D points in camera coordinates can be obtained. EPnP is the first non-iterative method with a time complexity of O(n) and has been widely used. However, EPnP is not accurate for n = 4 or 5, and it cannot handle the case of n = 3.

To improve accuracy, Li et al. [13] proposed RPnP, a non-iterative method with a time complexity of O(n). The basic idea is to divide the n 3D points into n-2 sets and construct a quartic polynomial. The PnP problem is then solved by minimizing the sum of squares of the quartic polynomial. RPnP can still obtain accurate solutions for n = 4 or 5.

Hesch et al. [14] proposed the DLS method with a time complexity of O(n). They formulated the PnP problem as a nonlinear least squares function represented by a multivariate polynomial. By using multiplication matrices, all solutions can be obtained. However, due to the use of Cayley parameterization, the accuracy of DLS is unstable and it becomes singular when the rotation is 180°. To address the singularity issue, Zheng et al. [15] used non-unit quaternion to

represent the rotation matrix and solved the PnP problem using the Gröbner basis method [16]. They also considered the symmetry of the solutions. However, in the solving process, the polynomials were treated as unconstrained general polynomials, neglecting the geometric relationship of the PnP problem. Kneip et al. [17] proposed the UPnP method, which takes into account the geometric relationship between known variables in the polynomials when solving with the Gröbner basis. Compared to OPnP, UPnP can obtain fewer solutions while ensuring global optimality of the solutions.

In practical applications, noise is inevitably present, which introduces errors in the projection equations. Some methods, such as EPnP, ignore this error during computation, while our method takes it into account. Unlike other global optimization methods like OpnP and UPnP, our method only returns a single global optimal solution, making it more convenient for practical use. Experimental results demonstrate that our method can robustly handle all experimental scenarios and achieve accuracy comparable to state-of-the-art algorithms.

3 Proposed Method

3.1 Preliminaries of the PnP Problem

The objective of the PnP problem is to determine the rotation matrix R and translation matrix t of a camera, given the known camera intrinsic matrix K, the 3D coordinates $P_i = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}^T$ of n points in world coordinates, and their corresponding 2D projection points $\begin{bmatrix} u_i^c & v_i^c & 1 \end{bmatrix}^T$. The relationship between the 3D points and 2D points can be expressed by the following equation:

$$\lambda_i \begin{bmatrix} u_i^c \\ v_i^c \\ 1 \end{bmatrix} = K \left(RP_i + t \right), i = 1, 2, \ldots, n \tag{1}$$

where λ_i denotes the depth factor of the i-th point.

3.2 The Minimization Problem

When we have the coordinates of 3D points in world coordinates, the coordinates of their corresponding 2D projected points in camera coordinates, and the camera's intrinsic matrix, we can leverage the projection relationship between the 3D points in camera coordinates and the 2D projected points. This allows us to convert the known coordinates of the 2D projected points into the coordinates of the 3D points in camera coordinates.By solving this projection relationship, we can estimate the camera's pose, which includes its position and orientation in the world coordinate system. we can leverage the projection relationship between the 3D points in camera coordinates and the 2D projected points to transform the known coordinates of the 2D projected points into the coordinates of the 3D

points in camera coordinates.

$$\lambda_i K^{-1} \begin{bmatrix} u_i^c \\ v_i^c \\ 1 \end{bmatrix} = RP_i + t, i = 1, 2, \ldots, n \tag{2}$$

The camera intrinsic matrix K is known.

$$K = \begin{bmatrix} fx & 0 & cx \\ 0 & fy & cy \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

K^{-1} can be obtained by taking the inverse of (3).

$$K^{-1} = \begin{bmatrix} 1/fx & 0 & -cx/fx \\ 0 & 1/fy & -cy/fy \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

When the variables K^{-1} and $\begin{bmatrix} u_i^c & v_i^c & 1 \end{bmatrix}^T$ on the left-hand side of (2) are known, the resulting vector after multiplication still has a last dimension of 1. Therefore, the multiplied form can be expressed as:

$$\lambda_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} r_1^T \\ r_2^T \\ r_3^T \end{bmatrix} P_i + \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}, i = 1, 2, \ldots, n \tag{5}$$

The left-hand side of (5) represents the coordinates of the 3D point in camera coordinates.The (5) can be used to obtain $\lambda_i = r_3^T p_i + t_3$. By substituting λ_i into (5), we can obtain the following expression.

$$\begin{matrix} u_i r_3^T p_i + u_i t_3 = r_1^T p_i + t_1 \\ v_i r_3^T p_i + u_i t_3 = r_2^T p_i + t_2 \end{matrix}, i = 1, 2, \ldots, n \tag{6}$$

Make the right side of the (6) equal to 0.

$$\begin{matrix} r_1^T p_i - u_i r_3^T p_i + t_1 - u_i t_3 = 0 \\ r_2^T p_i - v_i r_3^T p_i + t_2 - u_i t_3 = 0 \end{matrix}, i = 1, 2, \ldots, n \tag{7}$$

To better illustrate the relationship between the unknown variables and the known point coordinates, express (7) in the form of a Kronecker product and write the 3×3 rotation matrix as vector.

$$\left(\begin{bmatrix} 1 & 0 & -u_i \end{bmatrix} \otimes p_i^T \right) \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} + \begin{bmatrix} 1 & 0 & -u_i \end{bmatrix} \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} = 0, i = 1, 2, \ldots, n \tag{8}$$

$$\left(\begin{bmatrix} 1 & 0 & -v_i \end{bmatrix} \otimes p_i^T \right) \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} + \begin{bmatrix} 1 & 0 & -v_i \end{bmatrix} \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} = 0, i = 1, 2, \ldots, n \tag{9}$$

(8) and (9) involve only the unknown variables $\begin{bmatrix} r_1^T & r_2^T & r_3^T \end{bmatrix}^T$ and $\begin{bmatrix} t_1 & t_2 & t_3 \end{bmatrix}^T$, while the rest are known variables. Therefore, we can combine the two equations and explain them in simpler terms.

$$\begin{bmatrix} 1 & 0 & -u_i \\ 0 & 1 & -v_i \end{bmatrix} \otimes p_i^T \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} + \begin{bmatrix} 1 & 0 & -u_i \\ 0 & 1 & -v_i \end{bmatrix} \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} = 0, i = 1, 2, \ldots, n \tag{10}$$

Let's denote the coefficient matrices of $\begin{bmatrix} r_1^T & r_2^T & r_3^T \end{bmatrix}^T$ and $\begin{bmatrix} t_1 & t_2 & t_3 \end{bmatrix}^T$ as M and N, respectively. We can denote X as the $\begin{bmatrix} r_1^T & r_2^T & r_3^T \end{bmatrix}^T$ and t as the $\begin{bmatrix} t_1 & t_2 & t_3 \end{bmatrix}^T$. Equation (10) can be written as follows:

$$MX + Nt = 0 \tag{11}$$

In the previous derivations, we have been working with (1), which represents an ideal case where the equation holds exactly. However, in practical situations, there will always be some level of error or noise present, which means that the equation is not strictly equal. In previous PnP algorithms proposed by researchers, some methods are based on calculations in the ideal case without considering the presence of errors. In our calculations, we take into account the existence of errors and represent the error function as:

$$\begin{aligned} \varepsilon = & \ (MX + Nt)^T (MX + Nt) \\ = & \ X^T M^T MX + \ X^T M^T Nt + \ t^T N^T MX + \ t^T N^T Nt \end{aligned} \tag{12}$$

The equation above has X and t as unknowns. We can consider the error ε as a function of X and t. Taking the partial derivative of the equation with respect to t, $\frac{D_\varepsilon}{D_t} = 2N^T MX + 2N^T Nt$. The expression of t in terms of X can be obtained by setting the derivative to zero.

$$t = \ -\left(N^T N\right)^{-1} N^T MX \tag{13}$$

Substituting Eq. (13) into Eq. (12)

$$\varepsilon = \ X^T \left(M^T M - M^T N \left(N^T N\right)^{-1} N^T M\right) X \tag{14}$$

To facilitate representation, letting $L = M^T M - M^T N \left(N^T N\right)^{-1} N^T M$. (14) can be represented as

$$\varepsilon = X^T LX \tag{15}$$

3.3 Rotation Parameterization

For a rotation matrix R, it must satisfy the orthogonality constraint $RR^T = I$ and the determinant constraint $\det(R) = 1$. There are several parameterization methods for rotation matrices, such as Euler angles, rotation axis-angle, and

quaternion. Here, we adopt the representation of rotation matrices using non-unit quaternion. For regular quaternion$q = [a, b, c, d]^T$, they must satisfy the constraint $q^T q = 1$, while non-unit quaternion are unconstrained and can satisfy the constraints of a regular rotation matrix. The representation of rotation matrices using non-unit quaternion is as follows:

$$R = \frac{1}{s}\begin{bmatrix} a^2+b^2-c^2-d^2 & 2bc-2ad & 2bd+2ac \\ 2bc+2ad & a^2-b^2+c^2-d^2 & 2cd-2ab \\ 2bd-2ac & 2cd+2ab & a^2-b^2-c^2+d^2 \end{bmatrix} \tag{16}$$

In (16), where $s = a^2+b^2+c^2+d^2$ is a four-dimensional vector of unknowns without any constraints. Here,X is a nine-dimensional vector, and it can be observed that each term in is quadratic. We can make the substitution , letting $\tilde{X} = \left[a^2, ab, ac, ad, b^2, bc, bd, c^2, cd, d^2\right]^T$, and perform appropriate row and column transformations, resulting in a 10×10 matrix. By using non-unit quaternions, we transform (15) from a constrained solution to an unconstrained one. (15) can be expressed as follows:

$$\min_{a,b,c,d} \varepsilon = \tilde{X}^T L \tilde{X} \tag{17}$$

3.4 Global Optimization Solution

The error ε is always positive. When ε is minimized, the corresponding solution $\tilde{X}$ to the PnP problem can be obtained by taking partial derivatives of ε with respect to a, b, c, d, resulting in a system of polynomial equations.

$$\frac{\partial \varepsilon}{\partial a} = 0, \frac{\partial \varepsilon}{\partial b} = 0, \frac{\partial \varepsilon}{\partial c} = 0, \frac{\partial \varepsilon}{\partial d} = 0 \tag{18}$$

Solving the system of polynomial equations can yield all the solutions. When solving (18), the same method as UPnP [17] is used, resulting in eight solutions, similar to UPnP. However, we ultimately want to obtain only one globally optimal solution and need to filter the solutions. By substituting the solutions into (17), it is observed that the values obtained from the first four solutions are much smaller than those obtained from the last four solutions. When $n > 3$, the solution with the minimum reprojection error among the first four solutions can be selected as the optimal solution. When $n = 3$, it is noticed that in some cases, there are two different solutions with reprojection errors close to zero. In such cases, we return the solution with a reprojection error close to zero, allowing the user to choose the correct solution based on the actual tracking scenario.

3.5 Root Polishing

To further improve the accuracy, we can optimize the obtained solution using a first-order Newton method. In this case, we calculate the rotation matrix R, replace the original 3D points in world coordinates P_i with RP_i, and recompute

the matrix L. The solution obtained at this point is close to the identity matrix. Therefore, we can use the Cayley parameterization $c = \begin{bmatrix} c_1 & c_2 & c_3 \end{bmatrix}^T$ to represent the rotation matrix. The Cayley parameterization has singularity when the rotation is 180°, but in this case, the rotation angle is close to zero. We can represent $\tilde{X}$ as follows:

$$\tilde{X} = \frac{1}{1 + c_1^2 + c_2^2 + c_3^2} \left[1, c_1^2, c_2^2, c_3^2, c_1, c_2, c_3, c_1c_2, c_1c_3, c_2c_3\right]^T \in R^{10} \tag{19}$$

It is almost trivial and very efficient to compute the 3×1 Jacobian J_ε and the 3×3 Hessian H_ε of (17) around c = 0. Optimizing R through $R \leftarrow R - \delta R$ is obtained by calculating the Jacobian matrix and Hessian matrix through $\delta R = (H_\varepsilon)^{-1} J_\varepsilon$.

4 Experiment Results

In this section, we compare our method, referred to as "RPPnP," with several other PnP algorithms using simulated experiments. We compare our method with the following approaches: EPnP+GN [12] (EPnP when 3D points lie on a plane), RPnP [13], OPnP [15], UPnP [17], LHM [8] (SP+LHM when 3D points lie on a plane or near-singular configurations), and DLS [14].

4.1 Simulated Data

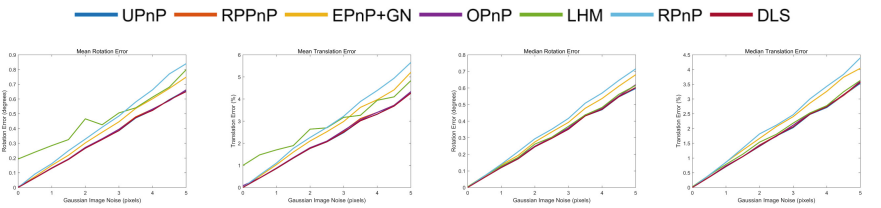
Assuming the focal length of the virtual camera is 800, we generate the coordinates of n 3D points in the world coordinate system. These points are then transformed into the camera coordinate system using randomly generated rotation and translation matrices(R_{true} and T_{true}). Finally, these 3D points are projected onto a two-dimensional plane to obtain their 2D coordinates. Gaussian noise is added to the 2D coordinates based on specific experimental requirements.

We use three configurations for selecting 3D points in world coordinates when evaluating the PnP algorithm:

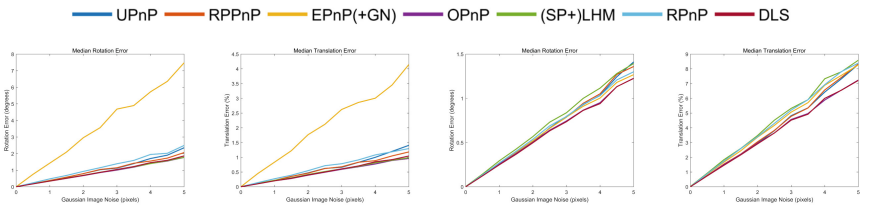
(1) Ordinary Case: $rank(W^TW) = 3$ and the smallest eigenvalue is not close to 0. The range of 3D points is [–2, 2] × [–2, 2] × [4, 8].
(2) Planar Case: $rank(W^TW) = 2$, the range of 3D points is [–2, 2] × [–2, 2] × [0, 0].
(3) Quasi-Singular Case:$rank(W^TW) = 3$ and the largest eigenvalue is close to 0. The range of 3D points is [1, 2] × [1, 2] × [4, 8].

The error between the true values (R_{true} and t_{true})and the estimated values (R and t) of the rotation matrix and translation matrix is calculated as follows:

$$\begin{aligned} e_{rot}(degrees) &= cos^{-1}\left(R_{true}^T R\right) \times \frac{180}{\pi} \\ e_{trans}(\%) &= \frac{||T_{true} - T||}{||T_{true}||} \times 100 \end{aligned} \tag{20}$$

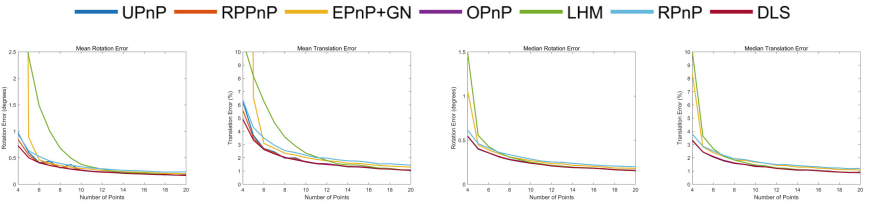


(a) The mean error of R, the mean error of t, the median error of R, and the median error of t when the 3D points are in the ordinary case

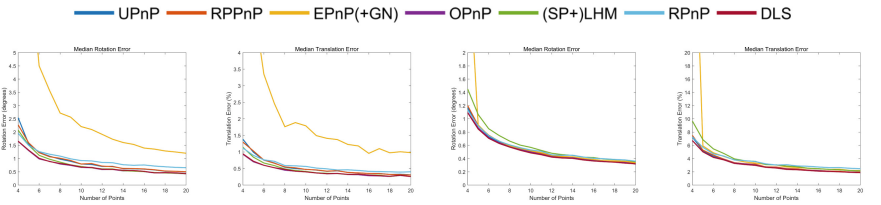


(b) The median error of R and t when 3D points are within the planar case, and the median error of R and t when 3D points are in the quasi-singular case.

Fig. 1. Experiment results with respect to varying noise levels (n = 10) in case of ordinary-3D, quasi-singular and planar point configura-tion, shown in (a) and (b) respectively.



(a) The mean error of R, the mean error of t, the median error of R, and the median error of t when the 3D points are in the ordinary case



(b) The median error of R and t when 3D points are within the planar case, and the median error of R and t when 3D points are in the quasi-singular case.

Fig. 2. Experiment results with respect to varying point numbers (Gaussian Image Noise is 2) in case of ordinary-3D, quasi-singular and planar point configuration, shown in (a) and (b) respectively.

4.2 The Effect with the Varying Noise

This experiment tested the effects of noise on the accuracy of all methods.We fixed $n = 10$ and the noise level added to the 2D point coordinates gradually increased from 0 to 5 (with an increment of 0.5 for each iteration). Figure 1 shows the mean and median errors of the rotation and translation matrices corresponding to the 3D points in the ordinary, quasi-singular, and planar cases. As shown in Fig. 1, as the noise increases, the accuracy of all PnP algorithms decreases, following a roughly linear distribution. Even for $n = 10$, EPnP+GN and RPnP are not accurate enough. Due to the presence of local optima, LHM and SP+LHM have larger mean errors in the general case. RPPnP and UPnP have similar errors in the planar and quasi-singular cases, but they exhibit slightly lower accuracy compared to OPnP and LHM. In the ordinary case, RPPnP and OPnP have similar errors, with a clear advantage in terms of mean error.

4.3 The Effect with the Varying Number of Points

This experiment investigated the performance of all methods with the varying number of points.We add zero-mean Gaussian noise with fixed deviation 2 pixels onto the image projections and gradually increasing the number of generated 3D points from 4 to 20 (increasing by 1 point per iteration). Figure 2 shows the mean and median errors of the rotation and translation matrices corresponding to the 3D points in the general, quasi-singular, and planar cases. As shown in Fig. 2, EPnP is sensitive to the change in the number of points and cannot achieve accurate poses when the number of points is small. LHM has lower accuracy when there are fewer points, possibly due to local minima. RPnP has relatively stable results for both small and large numbers of points, but with lower accuracy. UPnP, as a nonlinear optimization method, achieves high accuracy. RPnP and UPnP have similar errors in the general, planar, and quasi-singular cases.

In the mentioned experiments, only the median error is reported in the planar case and quasi-singular case. This is because in special cases, the error can fluctuate significantly. Additionally, both the PnP problem and its minimization problem, such as the P3P problem, can be handled by RPPnP. This inevitably leads to a decrease in robustness in special cases. Moreover, by looking at the median error, we can observe that RPPnP can achieve high accuracy solutions in most cases.

4.4 Computational Efficiency

In this experiment, the number of points, n, was gradually increased from 6 to 1006 in increments of 100. Each value of n was tested 500 times, and the average time was calculated. Figure 3 shows the computational efficiency of different PnP algorithms, and it is evident that our method has a significant advantage in terms of computational efficiency. Our method is 10 times faster than OPnP. Both our method and UPnP use mex files for implementation.

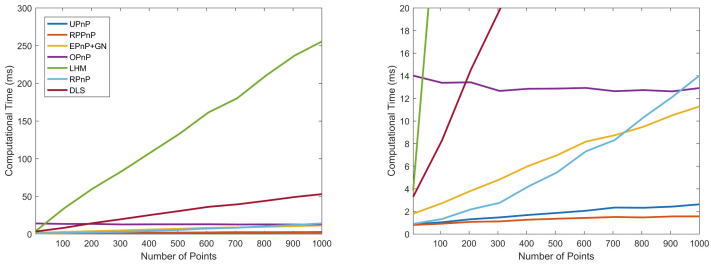


Fig. 3. Running time with respect to varying number of points.

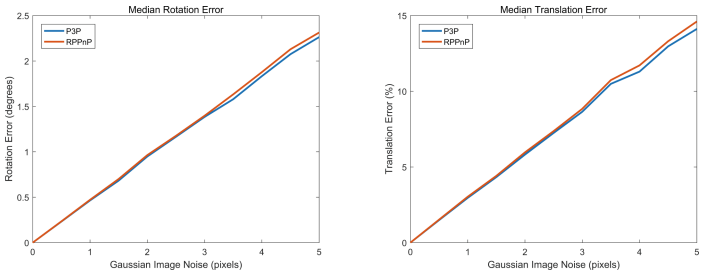


Fig. 4. The median rotation error and the median translation error with respect to varying noise levels in the minimal n = 3.

4.5 Comparison with the P3P Solver

We compared RPnP with the P3P [18] method. Figure 4 shows the median errors of R and t when the noise is increased from 0 to 5. It can be observed that RPnP and the compared P3P method achieve similar accuracy.

5 Conclusion

In this paper, we propose a non-iterative PnP algorithm that establishes a minimization problem based on the relationship between 3D points and their 2D projections. We parameterize the rotation matrix using non-unit quaternions and solve the resulting system of polynomial equations composed of the first-order partial derivatives to obtain the pose. Experimental results demonstrate that our method maintains high accuracy even with a small number of points and remains computationally efficient as the number of points increases.

References

1. Wei, W., Tan, L., Jin, G., Lu, L., Sun, C.: A survey of UAV visual navigation based on monocular SLAM. In: 2018 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC), pp. 1849–1853. Chongqing, China (2018). https://doi.org/10.1109/ITOEC.2018.8740355

2. O'Mahony, N., et al.: Adaptive multimodal localisation techniques for mobile robots in unstructured environments : a review. In: 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), pp. 799–804. Limerick, Ireland (2019). https://doi.org/10.1109/WF-IoT.2019.8767330
3. Marchand, E., Uchiyama, H., Spindler, F.: Pose estimation for augmented reality: a hands-on survey. IEEE Trans. Vis. Comput. Graph. **22**(12), 2633–2651 (2016)
4. Mur-Artal, R., Tardos, J.D.: ORB-SLAM2: an open-source SLAM system for monocular, stereo, and RGB-D cameras. IEEE Trans. Robot. **33**(5), 1255–1262 (2017)
5. Kneip, L., Sweeney, C., Hartley, R.: The generalized relative pose and scale problem: view-graph fusion via 2D-2D registration. In: 2016 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 1–9. Lake Placid, NY, USA (2016). https://doi.org/10.1109/WACV.2016.7477656
6. Yang, J., Li, H., Campbell, D., Jia, Y.: Go-ICP: a globally optimal solution to 3D ICP point-set registration. IEEE Trans. Pattern Anal. Mach. Intell. **38**(11), 2241–2254 (2016)
7. Pedregal, P.: Introduction to Optimization. TAM, vol. 46. Springer, New York (2004). https://doi.org/10.1007/b97412
8. Lu, C.P., Hager, G.D., Mjolsness, E.: Fast and globally convergent pose estimation from video images. IEEE Trans. Pattern Anal. Mach. Intell. **22**(6), 610–622 (2000)
9. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, UK (2003)
10. Abdel-Aziz, Y.I., Karara, H.M., Hauck, M.: Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry. Photogramm. Eng. Remote Sens. **81**(2), 103–107 (2015)
11. Quan, L., Lan, Z.: Linear N-point camera pose determination. IEEE Trans. Pattern Anal. Mach. Intell. **21**(8), 774–780 (1999)
12. Lepetit, V., Moreno-Noguer, F., Fua, P.: EPnP: an accurate O(n) solution to the PnP problem. Int. J. Comput. Vis. **81**, 155–166 (2009)
13. Li, S., Xu, C., Xie, M.: A robust O(n) solution to the perspective-n-point problem. IEEE Trans. Pattern Anal. Mach. Intell. **34**(7), 1444–1450 (2012)
14. Hesch, J.A., Roumeliotis, S.I.: A direct least-squares (DLS) method for PnP. In: 2011 International Conference on Computer Vision, pp. 383–390. Barcelona, Spain (2011). https://doi.org/10.1109/ICCV.2011.6126266
15. Zheng, Y., Kuang, Y., Sugimoto, S., Astrom, K., Okutomi, M.: Revisiting the PnP problem: a fast, general and optimal solution. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 2344–2351. Sydney, Australia (2013). https://doi.org/10.1109/ICCV.2013.291
16. Kukelova, Z., Bujnak, M., Pajdla, T.: Automatic generator of minimal problem solvers. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008. LNCS, vol. 5304, pp. 302–315. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88690-7_23
17. Kneip, L., Li, H., Seo, Y.: UPnP: an optimal $O(n)$ solution to the absolute pose problem with universal applicability. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8689, pp. 127–142. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10590-1_9
18. Banno, A.: A P3P problem solver representing all parameters as a linear combination. Image Vis. Comput. **70**, 55–62 (2018)

Proposal of Word Prediction Method for Gaze Swipe Text Entry

Hisanori Fujii(✉) and Ryo Katsuma

Osaka Metropolitan University, Sakai, Osaka 599-8531, Japan
sb22651s@st.omu.ac.jp, katsuma@omu.ac.jp

Abstract. Gaze swipe, one of the text entry methods that do not require physical interaction with a device, involves sequentially tracing the characters of the desired word on a displayed keyboard using the user's gaze. This approach holds the promise of enabling quick word entry. However, previous gaze swipe method presents an issue where the time required for typing increases with longer words, consequently burdening the user. As a solution, there is a demand for an input word prediction method that presents the desired word for typing before the user completes tracing all the characters with their gaze. Thus, this paper proposes a gaze swipe typing method incorporating word prediction functionality. Through experimentation, the base gaze swipe typing method's typing speed was measured at 8.30 words per minute, with an accuracy rate of 91% for the top 3 estimated words. Additionally, for the proposed method, when 70% of the gaze path length was typed, the accuracy rate for the top 3 predicted words was confirmed to be 63%.

Keywords: Gaze typing · Word prediction · Text entry

1 Introduction

The popular method for text entry on computers has typically been through keyboards, while in recent years, mobile devices like smartphones have introduced touch screen keyboards as well as alternative entry methods such as toggle typing and swipe typing. Most of these methods share the characteristic of typing using hands and are not suitable for users with dirty hands, hand injuries, or physical disabilities that make it difficult to handle key input devices. Gaze typing, which is an input method that does not use the hands, is expected to facilitate communication with others via a computer in various situations.

Gaze typing involves utilizing devices like eye trackers to convert the user's gaze information into input signals. The typical gaze-based text entry involves placing a virtual keyboard on the screen and repeatedly gazing steadily at the desired characters to type in text. While this method ensures stability, it comes with the drawback of consuming time for text entry. The necessity to strictly gaze at characters one by one adds a time-consuming aspect to the operation,

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 204–216, 2024.
https://doi.org/10.1007/978-3-031-63992-0_13

becoming a bottleneck in quick text entry. The fixed amount of time it takes to gaze at a character and make a decision is called **dwell time**, and keyboard text entry using dwell time is called **dwell typing**. Dwell typing requires the dwell time to be set to a sufficiently longer duration. The shorter the dwell time, the more likely it is that a character will be entered unintentionally. Because of these characteristics, dwell typing is considered to have a limit to the typing speed. In order to facilitate text entry, it is necessary to establish other methods that allow text to be entered in a short time with fewer errors.

Gaze swipe is expected to be a fast, gaze-based text entry method. In gaze swipe, users track the characters of the intended word using their gaze. By matching the resultant, the path of the user's gaze (hereafter referred to as **gaze path**) with the paths of pre-established words in a dictionary, the typing word is estimated. This method does not require the user to strictly gaze at each character as dwell typing, but rather, the input is accomplished by moving the user's gaze in a flowing way along the string of characters that make up the word. Therefore, instead of dwell typing, which required strictly typing one character at a time, gaze swipe is expected to enable faster typing. In particular, research has been conducted on English words, and this paper also covers them.

There is one thing to keep in mind when implementing gaze swipe. The computation required to evaluate the gaze path against words in a dictionary is complex, and it is necessary to reduce the computational load. A previous study of gaze swipe is EyeSwipe [4] by Kurauchi et al. EyeSwipe uses gaze gestures on the first key of a word to signal the start of word, and on the last key of a word to signal the end of word. This method determines the first and last letters of the word to be entered, reducing the number of candidate words, increasing accuracy and reducing the amount of computation. However, there is a problem with this method. It is impossible to estimate a word unless the user tracks the last letter of the word with the user's gaze. This makes text entry time-consuming and increases the burden on the user. Therefore, a method that enables text entry without tracking all the letters that make up a word is required.

In this paper, we propose a new gaze swipe method with a mechanism for predicting English words during the typing process. Although word prediction methods that predict the most likely word to be typed next based on the context of the immediately preceding typing string are possible, in this study, the goal is to predict words without dynamic information other than during the typing of a single word as a preliminary step. On a PC, typing one character at a time on the keyboard is easy to predict because the input is determined one character at a time, but gaze swipe, which uses path matching, is difficult because the character string is not determined in the middle of typing. The proposed method enables prediction by matching partial paths of words in a dictionary and ranks predicted words using the frequency of occurrence of English words on the Web.

2 Related Work

Gaze-based text input methods can be divided into three main categories: methods that improve dwell typing [7,8], methods that use characteristic interfaces

[1,10–12], and gaze swipe [4,9]. Table 1 describes the characteristics of each gaze-based text input.

Table 1. The characteristics of each gaze-based text entry

	typing speed	familiarity	robustness	ease of implementing prediction function
characteristic interfaces [1,10–12]	○	×	○	○
dwell typing' [7,8]	×	◎	×	○
gaze swipe [4,9]	○	○	○	×

Approaches that use characteristic interfaces are expected to optimize typing speed and accuracy and reduce the burden of user's entry. However, there is a problem of time-consuming learning, such as memorizing key layouts, and it would be better to use key layouts that many users are familiar with. This is one of the reasons why many gaze-based studies use a virtual QWERTY keyboard. QWERTY keyboard-based dwell typing including its advanced versions (denoted as dwell typing' in Table 1) are expected to be easy to master, the presence of at least some dwell time limits the improvement in typing speed. The study by Kristensson et al. [3] shows the potential of gaze swipe. They hypothesized a perfect interface in which typing word is completed by looking near the key on the virtual keyboard corresponding to the letter of the word and conducted simulations using this interface. After 40 min of practice, the subjects' average typing speed reached 46 wpm, showing that there is room for improvement in gaze swipe and high promise for research to make gaze swipe speed up. Therefore, this study aims to further improve usability by introducing word prediction for gaze swipe, which has a large room for growth. By design, widely used word prediction mechanisms cannot be simply applied to gaze swipe, and there is no research on predictive entry for gaze swipe. This study challenges that issue.

Conventional input prediction methods are designed for typing a sentence or word one character at a time from the front, and present candidates by performing a forward matching from the letters the user is typing. Existing "studies" on gaze gestures and dwell typing realizes the input prediction by this kind of prediction method [5,6]. On the other hand, dwell-free typing like gaze swipe is not a form of typing one letter at a time, so the letters required for forward matching is indeterminate, making it impossible to predict a word. For example, if the user tries to type "admini", the gaze path may pass through "asdfgbnmkijnji" in that order. Since it is difficult to accurately determine the user's input letters from this path, it is impossible to extract the letters necessary for forward matching. The proposed method can efficiently select candidates of the prefixes for forward matching and uses these prefixes to estimate words. Specifically, the proposed method selects possible prefixes in typing by using a chronologically ordered letters in the neighborhood of the gaze path, and then efficiently selects prefixes for prediction by assigning a preference to them. The proposed word

prediction method then performs a forward matching from the prefixes that are most likely to be in the typing state, in that order.

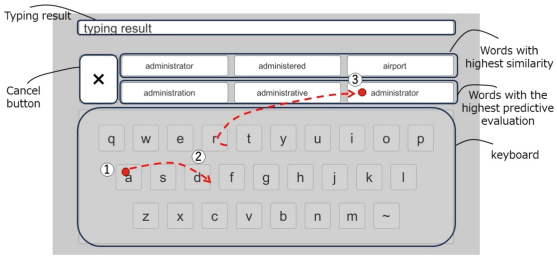


Fig. 1. Overview of the proposed system

3 Proposed System

The interface of the system targeted by this research is shown in Fig. 1. As shown in Fig. 1, the system has a virtual keyboard with 27 keys, including 26 alphabetic characters in a QWERTY layout and an "∼" for consecutive letters. The reason for using the key "∼" is described later in Sect. 4.1. It is assumed that the user types a word composed only of alphabetical characters. A non-contact eye tracker is attached to the monitor, and the user moves their gaze on the screen to type words. The movement of their gaze is measured by the eye tracker. The eye tracker can acquire the coordinates of both eyes' gaze on the monitor and the time at which the gaze is captured.

Describes the word input procedure. First, word input is initiated by gazing at the first key of the word for dwell time. Subsequently, the user traces each letter of the word using their gaze. While the user continues to move user's eyes on the virtual keyboard for typing, the system successively estimates the most similar words and additionally predicts the words to be typed by input prediction. These words are then listed and displayed in descending order of their respective evaluations. When the desired word is displayed, the user selects it to finish inputting the word. The input up to this point is regarded as a single word typing, and sentences can be composed by repeating this process.

4 Method

4.1 Base Gaze Swipe Typing Method

Acquisition of Gaze Path. Eye tracker can acquire the coordinates of the gaze in real time. The user tracks the letters of a word in sequence with their gaze as input for gaze swipe, and the coordinate time-series data of the gaze

path at that time is denoted by $\{\boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_t, ..., \boldsymbol{a}_{\mathrm{t_{end}}}\}$. Currently, the gaze coordinates obtained from a non-contact eye tracker are too noisy for using as a gaze path. This is because light from sunlight or illumination reflects off the pupil or glasses, adversely affecting accuracy. It is necessary to format the gaze path data from the perspective of removing noise and accurately determining the degree of similarity to each word. First, average filtering is performed to remove noise, as in the following Eq. (1).

$$\boldsymbol{a}'_t = \begin{cases} \boldsymbol{a}_t & t = 1, t_{\mathrm{end}} \\ \frac{\boldsymbol{a}_{t-1}+\boldsymbol{a}_t+\boldsymbol{a}_{t+1}}{3} & o.w \end{cases} \tag{1}$$

To accurately determine the degree of similarity, the gaze coordinates above a certain speed are removed from the gaze path. The formatted coordinates are then treated as the coordinate data of the gaze path.

Selection of Candidate Words. Gaze swipe requires efficient search of a large number of words in order to estimate which words are appropriate based on the gaze path. One of the most commonly used data structures for text is the trie, which is used as a dictionary in the proposed method. The coordinates of each key on the virtual keyboard are denoted as $\{\boldsymbol{\lambda}_{\mathrm{a}}, \boldsymbol{\lambda}_{\mathrm{b}}, ..., \boldsymbol{\lambda}_{\mathrm{z}}, \boldsymbol{\lambda}_{\sim}\}$. The initial letter of a typing word is denoted as c_{start}.

Let the set of words contained in the trie-structured dictionary be denoted as D. Since calculating the similarity for all words in dictionary D would be impractical in terms of computation time, it's necessary to select candidate words before calculating the similarity. In this paper, we refer to the selected words as **candidate words**. When selecting candidate words, letters present in the proximity of the gaze path are arranged in chronological order. Words that are similar to this sequence are chosen. Let the coordinate of the gaze at time t be represented as $\boldsymbol{x_t}$ and the distance between $\boldsymbol{x_t}$ and the coordinates of each letter be calculated. Then, k elements are obtained in order of shortest distance and the corresponding characters $C_t = \{c_{(1,t)}, c_{(2,t)}, ..., c_{(k,t)}\}$ are obtained. The character chosen for time t is represented as $c'_t \in (C_t \cup \mathrm{NULL})$. Note that "NULL" represents an empty string. The sequence of characters selected at each time up to time t forms a character string, which can be expressed using the following Eq. (2).

$$w_t = c_{\mathrm{start}} + c'_1 + c'_2 + ... + c'_t \tag{2}$$

The + sign in Eq. (2) represents the concatenation of strings. It follows that the text string represented by Eq. (2) takes a form close to the prefix of the word to be typed. Therefore, among the strings expressed in Eq. (2), we call one corresponding to the prefix of a word the **candidate prefix** at time t. A candidate word obtained a string of characters that form a word among the candidate prefixes of the end time. Among the candidate prefixes at the ending time t_{end}, only those that can form words are obtained as candidate words.

Calculation of Similarity. With a gaze swipe, the words with the highest similarity to the gaze path are presented on the screen among the candidate words. The DTW (Dynamic Time Warping) method is used for similarity. DTW calculates the DTW distance, which represents the distance between two time series data, and the shorter the DTW distance, the higher the similarity between the two data. In the DTW method for gaze swipe, one data is the coordinates time series of the gaze path, and the other is the center coordinates of letters of a word, arranged in order (hereafter referred to as **ideal path**). For example, the ideal path for "nice" is $\{\boldsymbol{\lambda}_{\mathrm{n}}, \boldsymbol{\lambda}_{\mathrm{i}}, \boldsymbol{\lambda}_{\mathrm{c}}, \boldsymbol{\lambda}_{\mathrm{e}}\}$.

While the user is inputting a gaze path, the system generates candidate words and calculates the similarity between all of them and displays the top words in terms of similarity on the interface. The user selects the desired word from the list to complete the input.

There is one drawback to using the DTW method to calculate word similarity. Two words like "meet" and "met" that differ only in whether or not they contain consecutive letters have equal paths and thus equal similarity. To solve this problem, different paths are provided by mapping the " " key on the keyboard to consecutive letters. In this way, "meet" and "aaa" are treated as "me t" and "a a" on the keyboard, respectively.

4.2 Input Prediction

Input prediction is a feature that predicts what words will be entered while typing. It is highly likely that the word to be entered is somewhere among the derivatives of the candidate prefixes. Thus, the similarity between the gaze path during typing and part of the ideal path is expected to be high. Use statistical information to determine the words that are most likely to be typed among them. Let $s \in D$ be a word in the dictionary and $n(s)$ be the frequency of occurrence in the 1-gram model and let $n_w(u)$ be the frequency of the prefix u, expressed as the sum of the frequencies of the derivations. The probability of transition from a prefix u to its child node v in trie is defined as follows.

$$p(v|u) = \frac{n_w(v)}{n_w(u)} \tag{3}$$

The probability of transition from a prefix u to a word s whose letters matches the prefix is defined as follows.

$$p(s|u) = \frac{n(s)}{n_w(u)} \tag{4}$$

As a concrete example, Fig. 2 shows the probability distribution of five words {ad, adapt, adopt, advice, advise}. In Fig. 2, arrows indicate the probability of transition. A black node indicates the existence of a word terminating at that node, and the probability of transition to that word is indicated by an arrow returning to its own node. In gaze swipe, the above probability distribution is used as a measure for input prediction. First, we calculate the similarity of the

candidate prefixes obtained from the base gaze swipe typing method. Then, up to m words with high transition probability are calculated among the words derived from each candidate prefix. It means that similarity is used as the first priority factor, and probability distribution is used as the second.

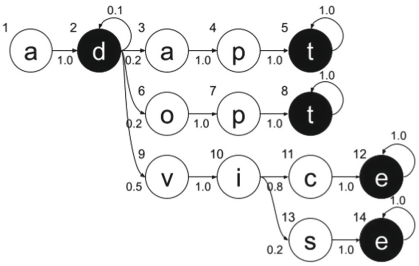


Fig. 2. Example of probability distribution

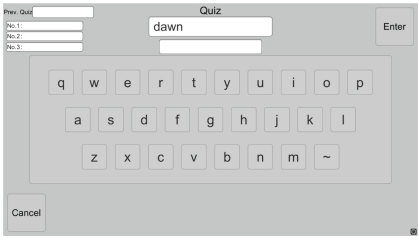


Fig. 3. Interfaces for experiment

5 Experiments

In this chapter, we describe two kinds of experiments. The first one is a comparison in which multiple subjects type words using both the proposed method and dwell typing interfaces. The purpose of this experiment is to evaluate the accuracy of word estimation and typing speed of the two methods. The second one evaluates the accuracy of the proposed method's input prediction under various conditions using the input data from the first one.

5.1 Experiment Environment

In the first experiment, we used the interface shown in Fig. 3, which implements the proposed method. This interface was implemented as a prototype of the proposed system to obtain the gaze path from the start to the end of typing, and is designed according to the proposed system, except that only the typing completion procedure differs slightly from the one shown in Fig. 1. The proposed system is designed to complete typing by selecting an estimated or predicted word. On the other hand, in the experimental interface, typing is completed by moving the user's gaze to the area in the upper right corner of the interface.

The interface randomly assigns a word from the dictionary to a question. After a word has been typed, the next question is presented, regardless of whether the typing was correct or incorrect. A non-contact eye tracker was attached to the monitor, and the user moved their gaze on the screen to type words, which was measured by the eye tracker. The eye tracker can acquire the coordinates of both gazes on the screen and the time at which the gazes were acquired. We used the Tobii Pro Nano eye tracker, which was a non-contact eye tracker. Its sampling rate was 60 Hz and was mounted on a 27-in. Full HD monitor.

5.2 Comparison Experiment

In the experiment, six subjects typed with both interface of the dwell typing and the proposed method. Calibration for each user is necessary to obtain highly accurate eye tracking information with a non-contact eye tracker. Even for a single user, the accuracy of calibration is not stable due to the fact that the distance and direction between the eye tracker and the user's eyes are not constant during the experiment and the light environment changes. Therefore, calibration was performed before the start of the experiment, as well as during the experiment.

Table 2. The typing speed, correct typing speed and top-i accuracy for each method

		typing speed [wpm]	correct typing speed [wpm]	top-i accuracy [%]
dwell typing		5.63	5.47	97.0
gaze swipe	top-1	8.30	6.92	83.4
	top2		7.41	89.3
	top3		7.56	91.0
	top4		7.67	92.4
	top5		7.69	92.6

The subjects were novice users with little experience using eye tracker. Before the data acquisition, the subjects watched a video explaining the typing procedure and practiced typing a few words as preparation. Each subject was given 6 sets of 5 min to type one word after another. Three sets consisted of dwell typing and the other three sets consisted of the proposed method. Subjects were asked to type as many correct answers as possible within a time limit.

The following is a description of the parameter values set for each typing method. In dwell typing, the dwell time, which is the time it takes for a letter to be typed, must be set. In this experiment, the dwell time was set to 0.4 s. The proposed method also uses dwell time for typing the first letter of a word, so we also set one to 0.4 s.

The list of dictionary words used in the proposed method consists of the 10,000 words with highest frequency of occurrence extracted by Kaufman from the Google Web Trillion Word Corpus [2]. The parameter k, the number of neighboring characters for selecting candidate words, was set to 3.

The evaluation items in this experiment were typing speed, correct typing speed, and top-i accuracy for both typing methods. Typing speed is the speed of typing regardless of correct or incorrect answers, and correct typing speed is the speed at which the correct word is typed. The top-i accuracy is the rate at which the correct word was among the top i estimated candidates. The top-i accuracy for the base gaze swipe was measured for word lengths. The top-i accuracy indicates the proportion of correctly estimated words among the top-i

ranked candidate words. In addition, the typing time for both input methods was measured for word lengths.

Result for Top-i Accuracy in Both Methods. Dwell typing yielded 507 typing data, while gaze swipe yielded 747 ones. Table 2 shows the typing speed, correct typing speed and top-i accuracy for each method.

However, only the top-1 accuracy is shown for dwell typing. On the other hand, the proposed gaze swipe had the typing speed 1.47 times faster than that of dwell typing, with a lower accuracy but a superior correct typing speed. Therefore, the proposed gaze swipe demonstrated better performance compared to dwell typing.

Results for Word Length and Accuracy in Gaze Swipe. Figure 4 shows the relationship between word length and top-i accuracy in gaze swipe. The shorter the typing word, the more stable the top-i accuracy became when the number of nearest neighbor characters k was increased. On the other hand, the top-i accuracy for longer ones was not stable. One possible reason for this is that the number of long words input was small. In addition, the subjects did not fully remember the spelling of long words, which may have negatively affected the accuracy.

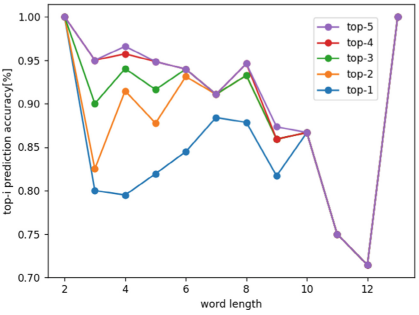


Fig. 4. The relationship between word length and top-i accuracy in gaze swipe

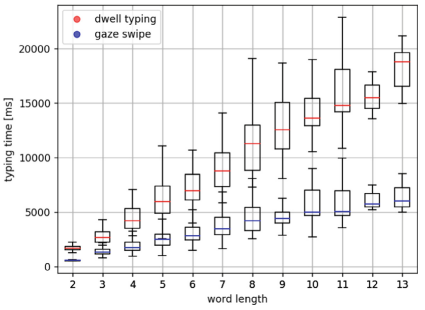


Fig. 5. The relationship between word length and typing time for both methods

Results for Word Length and Typing Time for both Methods. A box plot of the relationship between word length and the time it takes to typing word for both methods is shown in Fig. 5. However, since the start time of typing was not clearly known, the time measurement started when the first letter of the word was determined. Figure 5 shows that the time required to type a single word was shorter and the typing speed was faster with gaze swipe. Figure 5 also shows that both dwell typing and gaze swipe are close to proportional. A linear approximation yielded slopes of 1566.8 ms and 579.03 ms for dwell typing

and gaze swipe, respectively. By comparison, the typing speed of gaze swipe is about 2.71 ($\simeq$ 1566.8/579.03) times faster than that of dwell typing, regardless of the length of the word. This value is significantly different from the value of 1.47 obtained in the comparison of typing speeds for the entire experiment. The reason for this seems that the number of times the user canceled typing for gaze swipe because the desired word was not likely to be obtained during the deciding stage of the input. Therefore, if gaze swipe enables more reliable typing and requires fewer cancellations, the theoretical typing speed advantage of the eye-swiping method is approximately 2.71 times greater.

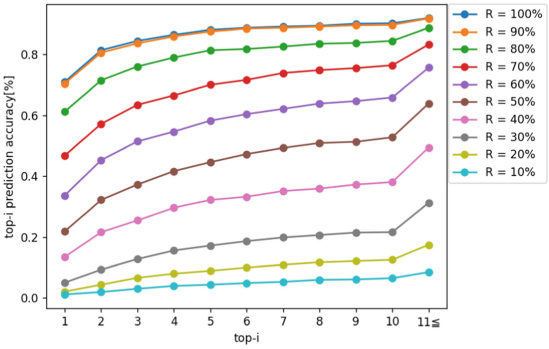


Fig. 6. Top-i accuracy for each split rate R

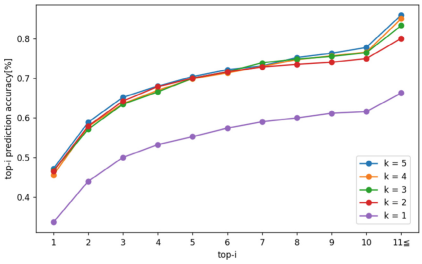


Fig. 7. Top-i accuracy for each number of nearest neighbor characters k

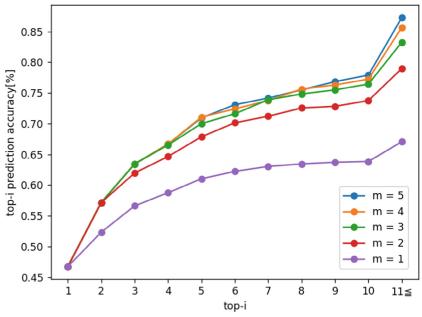


Fig. 8. Top-i accuracy for each number of derived words m

5.3 Analysis of Input Prediction

We analyzed the input prediction of the proposed method using the gaze paths obtained from the experiments in Sect. 5.2. Specifically, we calculated the predicted rank of the desired words and the top-i accuracy when predicting which words were going to be typing using a path that was clipped by R% from the

beginning of the gaze path. The calculations were performed for various values of the parameters k, m, and R.

The measured top-i accuracy when the split rate R of the gaze path was varied from 10% to 100% is shown in Fig. 6. Here, $k = 3$ and $m = 3$. The $11 \leqq$ in the horizontal axis represents the 11th or later word in the word priority. Figure 6 shows that the higher the split rate R, the higher the top-i accuracy, and the split rate almost saturated at $R = 90\%$.

The measured top-i accuracy when number of nearest neighbor characters k was varied from 1 to 5 is shown in Fig. 7. Here, $m = 3$ and $R = 70\%$. Figure 7 shows that there was a large difference in the top-i accuracy between cases where the number of nearest neighbor characters k was 1 and other cases, but there was almost no difference in the top-i accuracy between cases where k was between 2 and 5. Therefore, once there was no difference in the accuracy rate, it is reasonable to use the smaller value of $k = 2$ or $k = 3$ in order to reduce the computational load.

The measured top-i accuracy when number of derived words m was varied from 1 to 5 is shown in Fig. 8. Here, $k = 3$ and $R = 70\%$. Figure 8 shows that the higher the number of derived words m, the higher the top-i accuracy. Therefore, it is reasonable to adopt $m = 5$, which has a high top-i accuracy, in the actual typing system. The top-3 accuracy was 0.63 when the parameters were set to reasonable values $k = 3$, $m = 5$ and 70% of the gaze paths were typed.

Considerations on the Framework of Input Prediction Methods. In this experiment, we examine an input prediction method for typing in response to gaze swipe using the frequency of occurrence of each word. The frequency of occurrence in a 1-gram model was used as statistical information to determine the derivation of candidate prefixes. That is very simple statistical information, and input prediction using only that is limited in its accuracy. Therefore, instead of a 1-gram model, more complex statistical information such as user input history and language corpora could be used to improve prediction accuracy. However, the use of complex statistics is a trade-off for computation time, since gaze swiping requires multiple forward matching searches.

6 Conclusion and Future Work

In this paper, we proposed a new typing method for gaze swipe with a mechanism for input prediction of English words to be typed during typing. The proposed method enabled input prediction by matching the prefix paths of words in a dictionary and ranking words by using their frequency of occurrence. Experimental results show that the proposed method is 1.47 times faster than dwell typing. When the system presented three words in response to typing that traced all of the letters of a word, percentage of correct word existence in the presented words was 91%. Furthermore, when the system presented three words for a gaze typing in which the user had typed 70% of the gaze path length, the accuracy was 63%, thus demonstrating the relationship between the input prediction accuracy and

the ratio of the input gaze path length to the ideal path length. In this paper, as a preliminary step, we used the frequency of occurrence of words to rank the words to be predicted. In addition, we expect to improve the accuracy of prediction by taking into account various factors such as the context of the preceding and following words. The ranking of words to be predicted can be expressed by adjusting the probability distribution of word derivation, and this will be a subject for future work.

References

1. Bee, N., André, E.: Writing with your eye: a dwell time free writing system adapted to the nature of human eye gaze. In: Proceedings of the 4th IEEE Tutorial and Research Workshop on Perception and Interactive Technologies for Speech-Based Systems: Perception in Multimodal Dialogue Systems, pp. 111–122. Springer-Verlag, Kloster Irsee, Germany (2008). https://doi.org/10.1007/978-3-540-69369-7_13
2. Kaufman, J.: google-10000-English. https://github.com/first20hours/google-10000-english. Accessed 16 Aug 2023
3. Kristensson, P.O., Vertanen, K.: The potential of dwell-free eye-typing for fast assistive gaze communication. In: Proceedings of the Symposium on Eye Tracking Research and Applications, pp. 241–244. Association for Computing Machinery, Santa Barbara, California (2012). https://doi.org/10.1145/2168556.2168605
4. Kurauchi, A., Feng, W., Joshi, A., Morimoto, C., Betke, M.: EyeSwipe: dwell-free text entry using gaze paths. In: Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, pp. 1952–1956. Association for Computing Machinery, San Jose, California, USA (2016). https://doi.org/10.1145/2858036.2858335
5. MacKenzie, I.S., Chen, J., Oniszczak, A.: Unipad: single stroke text entry with language-based acceleration. In: Proceedings of the 4th Nordic Conference on Human-Computer Interaction: Changing Roles, pp. 78–85. Association for Computing Machinery, Oslo, Norway (2006). https://doi.org/10.1145/1182475.1182484
6. MacKenzie, I.S., Zhang, X.: Eye typing using word and letter prediction and a fixation algorithm. In: Proceedings of the 2008 Symposium on Eye Tracking Research & Applications, pp. 55–58. Association for Computing Machinery, Savannah, Georgia (2008). https://doi.org/10.1145/1344471.1344484
7. Morimoto, C.H., Amir, A.: Context switching for fast key selection in text entry applications. In: Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications, pp. 271–274. Association for Computing Machinery, Austin, Texas (2010). https://doi.org/10.1145/1743666.1743730
8. Mott, M.E., Williams, S., Wobbrock, J.O., Morris, M.R.: Improving dwell-based gaze typing with dynamic, cascading dwell times. In: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, pp. 2558–2570. Association for Computing Machinery, Denver, Colorado, USA (2017). https://doi.org/10.1145/3025453.3025517
9. Pedrosa, D., Pimentel, M.D.G., Wright, A., Truong, K.N.: Filteryedping: design challenges and user performance of dwell-free eye typing. ACM Trans. Access. Comput. **6**(1) (2015). https://doi.org/10.1145/2724728

10. Rough, D., Vertanen, K., Kristensson, P.O.: An evaluation of dasher with a high-performance language model as a gaze communication method. In: Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces, pp. 169–176. Association for Computing Machinery, Como, Italy (2014). https://doi.org/10.1145/2598153.2598157
11. Ward, D.J., Blackwell, A.F., MacKay, D.J.C.: Dasher-a data entry interface using continuous gestures and language models. In: Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology, pp. 129–137. Association for Computing Machinery, San Diego, California, USA (2000). https://doi.org/10.1145/354401.354427
12. Wobbrock, J.O., Rubinstein, J., Sawyer, M.W., Duchowski, A.T.: Longitudinal evaluation of discrete consecutive gaze gestures for text entry. In: Proceedings of the 2008 Symposium on Eye Tracking Research & Applications, pp. 11–18. Association for Computing Machinery, Savannah, Georgia (2008). https://doi.org/10.1145/1344471.1344475

AI and Machine Learning

Integrating Social Environment in Machine Learning Model for Debiased Recommendation

Yihong Zhang[1(✉)], Lina Yao[2], and Takahiro Hara[1]

[1] Graduate School of Information Science and Technology, Osaka University, Osaka, Japan
yhzhang7@gmail.com, hara@ist.osaka-u.ac.jp

[2] School of Computer Science and Engineering, The University of New South Wales, Sydney, Australia
lina.yao@unsw.edu.au

Abstract. Social data provided by social media platforms contains rich social environment information, which has been used in many ubiquitous tasks such as disaster monitoring, epidemic tracking, or stock market movement prediction. In this paper, we show that social environment information can be used to debias recommendation system. Recommendation systems are used to extract the preference of e-commerce platform users for predicting what would they see or buy next. Most recommendation systems that rely on machine learning models are currently facing the problem of bias, which occurs when the system is isolated in a platform and trained solely on past data. While various debiasing methods have been proposed, the problem remains largely unsolved. In response, we propose a recommendation model that uses real-time social data to reduce recommendation bias while also improving accuracy. The proposed model integrates social data, represented as embeddings generated by language models, into a traditional machine learning model called single value decomposition (SVD). Empirical evaluations on two pairs of real-world e-commerce plus social data datasets show that our model is superior in both recommendation accuracy and bias reduction compared to state-of-the-art debiased recommendation methods.

Keywords: environmental factor extraction · social media mining · debiased recommendation system

1 Introduction

Social media platforms facilitate user-generated content, outputting hundreds of millions of text messages and other data every day, covering almost any topic in social conversation [27]. These social data can be seen as a representation of the

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 219–230, 2024.
https://doi.org/10.1007/978-3-031-63992-0_14

whole social environment and have been used to help many ubiquitous computational tasks [17]. For example, it has been shown that tweets from Twitter can be used for detecting earthquakes [24] and monitoring epidemic spreads [2]. In some tasks, the environmental factor has been extracted from social data to improve an existing model. Example of these tasks include crime prediction, product sales forecasting, stock market movement prediction, and traffic monitoring.

Recommendation systems have now been widely used in many internet platforms, such as Amazon, eBay, and YouTube, providing personalized information to hundreds of millions of users worldwide [9]. Currently, recommendation system research is facing a critical situation that the system, consisting of a recommendation model, data, and users, is usually isolated in a specific platform. As a result, recommendation systems are developing so-called *bias*, due to the fact that the model is trained only on the past, static data [8]. Even though many different recommendation models have been proposed, bias is prevalent in almost all models [15]. An example impact of the bias is that the item popular in the past will get more recommendations, and will become even more popular in the future, while unpopular items get fewer recommendations and become even more unpopular. This is called *popularity bias* [1], one of the known biases in an isolated recommendation system, and is known as being connected to fairness and diversity.

We argue that environmental information extracted from social data can also be helpful in improving the recommendation of e-commerce products. For example, a festival discussed on social media can invoke wine purchases [31]. A natural disaster that is happening and discussed on social media can be followed by emergency goods purchases. Furthermore, seasonal information discussed on social data can also be hints for selling certain products [28]. Such social context may not be found within the recommendation system, but is present in social data. The problem is how to incorporate useful social data into existing recommendation systems.

Our insight of studying recommendation bias is that one of the causes of such bias is the fact that most recommendation systems are built on a machine learning model, which is isolated and trained only on past data within the system. This causes *feedback loop* which leads to bias. As such, if we can bring new information and out-of-system contexts to such an isolated system, illustrated in Fig. 1, we can expect less bias and even better recommendation accuracy. We use social data from social media platforms as the new context because they contain the latest information and can be easily obtained. A challenge, though, is what data should we use and extract from the massive corpus.

Social data are largely unstructured and noisy. It has been a challenging problem to represent real-time social data [11]. The emergence of pre-trained language models [3], however, gives us a convenient means for this task. Pre-trained language models such as GPT [12] and BART [20] let us input unstructured text and obtain embeddings that contain subtle information. In this paper we use a pre-trained language model to obtain social data embeddings, and then integrated them as the environmental factor to the well-known single value decomposition

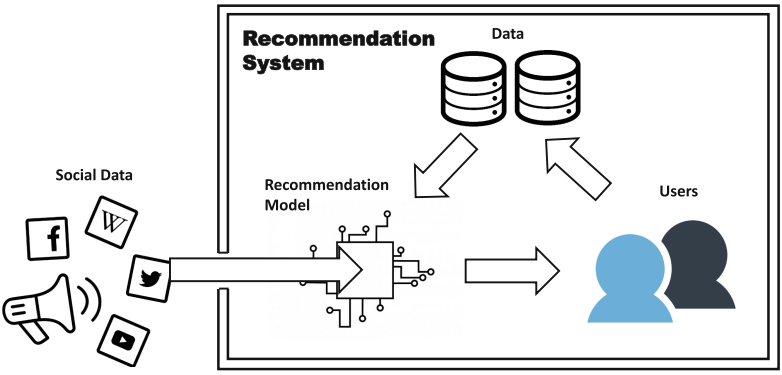


Fig. 1. Enhancing isolated recommendation system with social data

(SVD) learning model, similar to other latent factor models [18,19]. Testing on two pairs of real-world e-commerce and social data datasets, we show that our model is superior in terms of both the recommendation accuracy and the bias reduction, compared to state-of-the-art debiased recommendation methods. To summarize, our main contributions with this paper are:

- We identify the problem of recommendation bias as the result of an isolating learning process. In contrast to other hypotheses, our direction is to reduce the isolating state of the learning model.
- We propose to use the environmental factor extracted from social data to reduce the isolated state of recommendation models and improve the recommendation diversity. To the best of our knowledge, this is the first work that proposes to use real-time social data to reduce biases in a recommendation system.
- We conduct comprehensive empirical evaluations on two pairs of e-commerce and social media datasets. The evaluation results show that our approach not only reduces the bias, but improves recommendation accuracy in general.

2 Related Work

Some research efforts have already explored methods of using social data to enhance the recommendation system of another domain [7,10]. However, many of these methods have strict assumptions, such as shared users between the social media platform and the recommendation domain, or famous items that can be linked to entities in social data [26,33]. Such assumption significantly limits their applicability. In our work, we do not assume any shared data between the social media platform and the recommendation domain. The social data in our work represents the whole social environment, from which the recommendation system can borrow information as needed. In this way, we do not require the item to be famous due to the coverage and the details of the social data.

Our work also takes inspiration from recommendation debiasing works. For example, Zhang et al. proposed that the popularity bias is partially caused by the temporary item bias [30]. They used a deconfounding method to reduce the bias. Misztal-Radecka and Indurkhya proposed to analyze recommendation disparity for detected biased attributes [22]. Following the finds, they proposed a debiased model selection method. Chaney et al. investigated how algorithmic confounding in recommendation systems increases homogeneity and decreases utility [6]. These studies, however, restrain themselves within the system, which makes their effectiveness limited.

3 Debiasing Recommendation with Social Data

We deal with the traditional implicit recommendation problem, with the exception that we allow the addition of out-of-system social data. In this section, we will first define the problem. Then we will present our method for integrating the real-time massive social data to a recommendation system.

3.1 Problem Formulation

The base problem we are dealing with is making dynamic personalized recommendations according to user preference and time. We follow the approach of collaborative filtering with implicit feedback. In a dynamic setting, the model should make a time-aware prediction $\hat{y}_{uit} = g(u, i, t)$, where $\hat{y}_{uit}$ is the predicted preference of user u towards item i at time t.

The main point of this paper is to incorporate social data into a dynamic recommendation model to improve recommendation performance. We assume we are monitoring social media in real-time, so that we have the data for the current timeframe and some past timeframes. As such, for each timeframe t, we have a collection of short text documents D_t. This collection of documents contains rich data. To use them in computational tasks, however, there may be two issues. First, the volume of data is large, with each D_t potentially containing millions of short texts. Second, the length of text documents in D_t is different. This will cause a problem for computational models that rely on fixed-length inputs. To address the issues, we need to have a transformation function

$$R(t) = \rho(D_t) \tag{1}$$

to transform the data into a representation that is much smaller than the data and has a fixed length for each timeframe t, while still preserving the most important information for the timeframe.

To use social data to support the model is to incorporate the social data representation R as an extra input to the model, so that the prediction becomes

$$\hat{y}_{uti} = g(u, i, t, R(t)). \tag{2}$$

Our task, then, is to devise the social data representation R and the integrated recommendation model g.

3.2 Social Data Representation

Traditional text stream techniques that can be used to represent social data include bag-of-words and topic modeling [4]. One common deficiency of these techniques is that they treat words as independent entities, not considering the sequence or context in the text. Natural language processing (NLP) research has since then produced techniques such as BERT [29] that can represent text while taking into account the context. Recently, pre-trained language models such as GPT [5] have shown superiority in many NLP tasks such as text summarization and question answering. One reason they have this superiority is due to their highly sophisticated text representation. In this paper, we propose to take advantage of the superior text representation capability of a pre-trained language model to help our task. Specifically, we use a language model to provide a mapping from text to real-value embeddings:

$$E(t) = \mathrm{LM}(D_t) \tag{3}$$

There are several choices for the pre-trained language models. In this paper we use BART [20] due to its efficiency when running on local computers. We use the *facebook/bart-base* model[1], which is a median size language model with 139M parameters. This can be run on our local computer with one NVIDIA GeForce GTX 1070 GPU with 8 GB graph memory. We can expect that larger and more complex models may provide better results. But these larger models are either too slow to run locally or too expensive to run as an online service, as we have potentially millions of text to process. Thus given our experimental environment, the BART model is a suitable choice.

Furthermore, we can calculate embeddings for a smaller time unit, such as an hour, and stack them together for a larger time unit, such as a day:

$$R(t) = \{E(t_s) : \forall t_s \in t\} \tag{4}$$

In this way, we essentially produce for each timeframe t a matrix of $n \times m$, where n is the number of small time units in a larger time unit, e.g., 24 in the case of hours in a day, and m is the text embedding size.

3.3 Recommendation Model

Our base recommendation model is a latent factor model [18]. In a latent factor model, users and items have corresponding latent factor embeddings, denoted as p_u and q_i. When we combine a pair of user-item embeddings through multiplication, i.e., $p_u \cdot q_i$, we can reveal user u's preference towards item i. And there are global latent factors for each user and item, denoted as b_u and b_i, also called user and item biases. The complete latent factor model is thus

$$\hat{y}_{uit} = p_u \cdot q_i + b_i + b_u, \tag{5}$$

[1] https://huggingface.co/facebook/bart-base.

where the parameters can be learned through gradient descent techniques. We note that in a complete implementation, traditional regularizers are to be used, but are not shown here for simplicity.

As we mentioned in the introduction, an e-commerce item may suddenly become important due to some social media discussions at the time. As such, on the base recommendation model, we would like to add a social factor, denoted as $s_{i,t}$, that indicates the item's significance with regard to the social media discussion at the time. So the extended model becomes:

$$\hat{y}_{uit} = p_u \cdot q_i + b_i + b_u + s_{i,t}. \tag{6}$$

We call the model Language Model-enhanced Social Recommender (LMSocRec). Now we deal with how to generate $s_{i,t}$ given a social media representation $R(t)$. We need a method to make this latent factor cover both the social data and the item characteristics, and we find that the *attention* [25] is suitable for this task. Following the approach of attention, we can make a social context encoder for each item at a time. The module is depicted in Fig. 2.

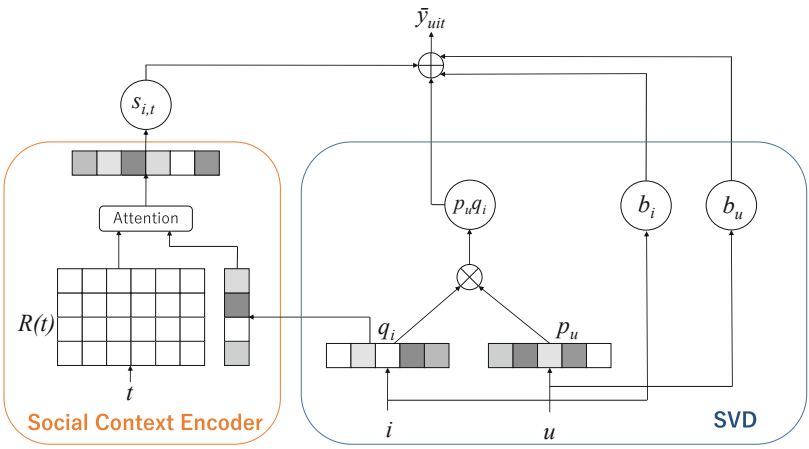


Fig. 2. Overview of the LMSocRec model

Since we assume no shared information between social data and the recommendation system, the item embedding size may be different from dimensions of $R(t)$. We use a linear transformation on the item embedding:

$$r_i = \sigma(\mathbf{h}_i^\top \cdot q_i) \tag{7}$$

where σ is a linear transformation layer, and $\mathbf{h}_i$ is the layer weights. Now the dimension of r_i is the same as embedding dimension m in $R(t)$.

We used r_i to calculate attention weights using what is called *general attention* [21]:

$$\mathbf{a}_i = \text{softmax}(r_i^\top \mathbf{W} R(t)_j) \tag{8}$$

where $\mathbf{W}$ is a learnable weighting matrix, $R(t)_j$ is the j-th column of the social data matrix, and $\mathbf{a}_i = \{a_{i1}, ..., a_{im}\}$.

Then we calculate the output of the social context encoder module, a context vector c_i for item i

$$\mathbf{c}_i = \sum_j a_{ij} R(t)_j. \tag{9}$$

In this way, the context vector c_i contains both the social data information and the item information. Finally, we use linear regression to convert all contained information into the social latent factor:

$$s_{i,t} = w^\top c_i + b, \tag{10}$$

where w and b are learnable parameters.

4 Experimental Evaluation

We conduct experimental evaluations on two pairs of real-world datasets, in order to test the effect on recommendation accuracy and bias reduction, when applying our method and compared methods. In this section, we first introduce the datasets used and the experimental setup. Then we discuss the evaluation results.

4.1 Datasets

We test our approach intently on e-commerce items that are not famous or are not linkable to famous entities. Therefore we prepare two such e-commerce datasets. The first dataset is provided by our industry partner, which is a flash sales (FS) dataset generated in a Japanese e-commerce platform. It includes daily-life items such as food, cosmetics, and home appliances. The second dataset is a subset of the well-known Amazon dataset [23], containing only the category of grocery, which includes food, drinks, and other daily-life items. Both datasets include a number of users, items, and timestamped purchase records. The FS dataset is of a four-month period between June and September 2017. The Amazon dataset is of a six-month period between January and June 2018.

In the same period as the e-commerce datasets, we prepare corresponding social data datasets. For the FS dataset, we collect tweets from a number of Japanese Twitter users through the timeline API[2]. For the Amazon dataset, we use the publicly available Reddit dataset[3] of the same period. The number of users, items, interactions, and tweets for each pair of the datasets are shown in Table 1.

We divide the datasets into training and test datasets. For the FS+Twitter dataset, we use the first three months as the training data and the last month as the test data. For the Amazon+Reddit dataset, we use the first four months as the training data, and the last two months as the test data.

[2] https://developer.twitter.com/en/docs/twitter-api/tweets/timelines/introduction.
[3] https://files.pushshift.io/reddit/submissions/.

Table 1. Dataset sizes of e-commerce and social data

	User	Item	Interaction	Tweets
FS+Twitter	40,895	20,542	125,968	2.4M
Amazon+Reddit	18,659	11,291	57,489	69M

4.2 Experimental Setup

We test our model with several other recommendation models, including a state-of-the-art debiased model and a social data-based model.

- MF [14]. The well-known matrix factorization model that produces two lower-rank matrices that represent latent user factor and item factor. It lacks user and item biases compared to SVD.
- Popularity-bias Deconfounding (PopDecon) [30]. This is a state-of-the-art debiased recommendation model. We implement it with BPR-MF according to the description in the paper.
- SVD. The base recommendation model.
- TimeSVD [19]. This is a time-sensitive SVD model. We implement the period-based version of the model since there is no time overlapping between training and test datasets in our evaluation.
- Item Social Trend Encoder (ISTE) [32]. This is a state-of-the-art recommendation model that integrates real-time social data. It differs from our model in that it uses NeuMF [13] architecture and concatenates social data as a vector in the output layer.

We implement our model and all compared models as neural networks using Python and Tensorflow[4]. We use 20 as the embeddings size for all MF and SVD models.

We follow the established method and measure recommendation accuracy as Recall and Normalized Discounted Cumulative Gain (NDCG) [16]. Both measure the ranking quality when given a positive test case and a number of negative candidates. Recall is based on hit or not-hit, while NDCG is based on the position of the positive item in the ranking.

For measuring bias reduction, we follow an existing approach [30]. We first divide the items into 10 groups according to the order of their interaction number in the training data. Group 1 contains the most popular items, group 10 contains the least popular items, and so on. At the same time, we keep the total number of training interactions in each group about the same, which means group 1 has fewer items and more interactions per item, etc. If a recommendation model is not biased, the recommended items should have the same distribution over groups as in the training data, which is 0.1 for all groups. Deviation from this distribution indicates recommendation bias. A lower deviation generally means lower recommendation bias. Like [30], we measure the deviation as the standard deviation.

[4] The source code will be made available soon.

4.3 Results Analysis

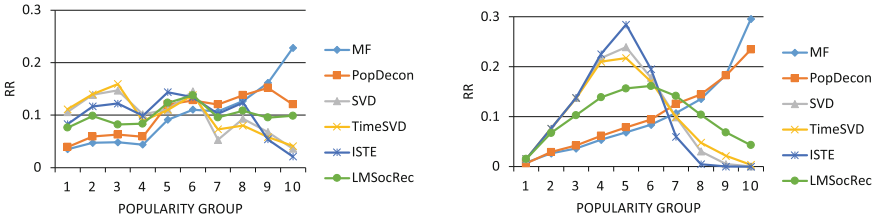
First we look at the impact of different models on recommendation accuracy, which is shown in Table 2. We can see that SVD models are generally better than MF models (MF and PopDecon) after considering user and item biases. Our method of integrating social data improves NDCG@5 by 25% and 10% for two datasets, compared to the original SVD. TimeSVD gives time context to the item like our method but not the social context, and it turns out this is not helpful for improving recommendation accuracy. ISTE outperforms other baselines by considering social context, which other baselines do not have. Our method, though, is better than ISTE, even though the latter uses a more complex architecture. So our model is successful in terms of improving recommendation accuracy.

Table 2. Recommendation accuracy of different models

	FS+Twitter		Amazon+Reddit	
	Recall@5	NDCG@5	Recall@5	NDCG@5
MF	0.154	0.108	0.112	0.075
PopDecon	0.237	0.165	0.145	0.104
SVD	0.335	0.243	0.269	0.178
TimeSVD	0.315	0.228	0.263	0.174
ISTE	0.334	0.245	0.269	0.173
LMSocRec	**0.342**	**0.265**	**0.351**	**0.256**

Next, we look at the impact on bias reduction. Figure 3 shows the distribution of item recommendations over popularity groups. We can see for both datasets, the methods operate in two modes. The MF-based methods tend to recommend the least popular items, because these are the majority of the items in the test candidates. The SVD-based methods tend to recommend items with popularity somewhere in the middle, especially popularity group 5. That is because these methods learned popular item bias based on the training data. Both types of methods do not recommend the most popular items in the training data because the market is changing for both datasets, and popular items in the training period do not appear in the test period. Within the SVD-based methods, we can see the proposed LMSocRec furthermore recommends more diversified items, and the distribution over popularity groups is closer to the training distribution, which is constant at 0.1 over groups.

We can also look at the single value summary, i.e., the deviation score. Table 3 shows the standard deviation of the recommendation distribution for different models. We can see PopDecon method, specialized in debiasing, achieves lower deviation, especially for the Amazon dataset. However, the lowest deviation achieved is by our proposed method, indicating its effectiveness in bias reduction and recommendation diversification.



(a) Flash Sales Dataset (b) Amazon Dataset

Fig. 3. Recommendation rate over popularity groups

Table 3. Standard deviation of recommendation distribution over popularity groups

	MF	PopDecon	SVD	TimeSVD	ISTE	LMSocRec
FS	0.062	0.040	0.038	0.038	0.038	**0.019**
Amazon	0.087	0.072	0.090	0.080	0.105	**0.050**

5 Conclusion

Social data that represents the whole social environment has been aiding many ubiquitous tasks. In this paper, we attempt to use social data to debias recommendation models. One major cause for a recommendation system to generate biased recommendations is that the system is isolated in a platform, and the machine learning model is trained on a feedback loop. In contrast to existing debiasing studies, we propose a method to bring out-of-system social data into the recommendation system to improve recommendation performance. The method integrates social data, represented as embeddings generated by a pre-trained language model, as a latent factor into the traditional SVD model. Experimental evaluations with two pairs of e-commerce plus social data datasets show that our method is successful, outperforming state-of-the-art debiasing methods in terms of both recommendation accuracy and bias reduction. In the future, we will continue to investigate more effective ways of social data integration.

References

1. Abdollahpouri, H.: Popularity bias in ranking and recommendation. In: Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society, pp. 529–530 (2019)
2. Amin, S., Uddin, M.I., AlSaeed, D.H., Khan, A., Adnan, M.: Early detection of seasonal outbreaks from twitter data using machine learning approaches. Complexity **2021** (2021)
3. Birhane, A., Kasirzadeh, A., Leslie, D., Wachter, S.: Science in the age of large language models. Nat. Rev. Phys. 1–4 (2023)
4. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. J. Mach. Learn. Res. **3**(Jan), 993–1022 (2003)

5. Brown, T., et al.: Language models are few-shot learners. Adv. Neural Inf. Process. Syst. **33**, 1877–1901 (2020)
6. Chaney, A.J., Stewart, B.M., Engelhardt, B.E.: How algorithmic confounding in recommendation systems increases homogeneity and decreases utility. In: Proceedings of the 12th ACM Conference on Recommender Systems, pp. 224–232 (2018)
7. Chen, C., et al.: An efficient adaptive transfer neural network for social-aware recommendation. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 225–234 (2019)
8. Chen, J., Dong, H., Wang, X., Feng, F., Wang, M., He, X.: Bias and debias in recommender system: a survey and future directions. ACM Trans. Inf. Syst. (2020)
9. Davidson, J., et al.: The youtube video recommendation system. In: Proceedings of the fourth ACM Conference on Recommender Systems, pp. 293–296 (2010)
10. Deng, Z., Yan, M., Sang, J., Xu, C.: Twitter is faster: personalized time-aware video recommendation from twitter to youtube. ACM Trans. Multimed. Comput. Commun. Appl. (TOMM) **11**(2), 1–23 (2015)
11. Dhingra, B., Zhou, Z., Fitzpatrick, D., Muehl, M., Cohen, W.W.: Tweet2Vec: character-based distributed representations for social media. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Volume 2: Short Papers (2016)
12. Floridi, L., Chiriatti, M.: GPT-3: its nature, scope, limits, and consequences. Mind. Mach. **30**, 681–694 (2020)
13. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.-S.: Neural collaborative filtering. In: Proceedings of the 26th International Conference on World Wide Web, pp. 173–182, Perth, Australia, 2017. ACM (2017)
14. He, X., Zhang, H., Kan, M.Y., Chua, T.S.: Fast matrix factorization for online recommendation with implicit feedback. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 549–558 (2016)
15. Jannach, D., Lerche, L., Kamehkhosh, I., Jugovac, M.: What recommenders recommend: an analysis of recommendation biases and possible countermeasures. User Model. User-Adap. Inter. **25**(5), 427–491 (2015)
16. Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of IR techniques. ACM Trans. Inf. Syst. (TOIS) **20**(4), 422–446 (2002)
17. Kalampokis, E., Tambouris, E., Tarabanis, K.: Understanding the predictive power of social media. Internet Res. (2013)
18. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 426–434 (2008)
19. Koren, Y.: Collaborative filtering with temporal dynamics. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 447–456 (2009)
20. Lewis, M., et al.: Bart: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 7871–7880 (2020)
21. Luong, M.-T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 1412–1421 (2015)
22. Misztal-Radecka, J., Indurkhya, B.: A bias detection tree approach for detecting disparities in a recommendation model's errors. User Model. User-Adapt. Interact. 1–37 (2022)

23. Ni, J., Li, J., McAuley, J.: Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 188–197 (2019)
24. Sakaki, T., Okazaki, M., Matsuo, Y.: Earthquake shakes twitter users: real-time event detection by social sensors. In: Proceedings of the 19th International World Wide Web Conference, pp. 851–860 (2010)
25. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
26. Wang, X., He, X., Cao, Y., Liu, M., Chua, T.S.: KGAT: knowledge graph attention network for recommendation. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 950–958 (2019)
27. Wu, S., Hofman, J.M., Mason, W.A., Watts, D.J.: Who says what to whom on twitter. In: Proceedings of the 20th International World Wide Web Conference, pp. 705–714 (2011)
28. Yamada, K.: Detecting two types of seasonal words using simple autocorrelation analysis. In: 2017 IEEE International Conference on Big Data (Big Data), pp. 3162–3167. IEEE (2017)
29. Yates, A., Nogueira, R., Lin, J.: Pretrained transformers for text ranking: bert and beyond. In: Proceedings of the 14th ACM International Conference on Web Search and Data Mining, pp. 1154–1156 (2021)
30. Zhang, Y., et al.: Causal intervention for leveraging popularity bias in recommendation. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 11–20 (2021)
31. Zhang, Y., Hara, T.: Explainable integration of social media background in a dynamic neural recommender. ACM Trans. Knowl. Discov. Data (TKDD) **17**(3), 1–14 (2023)
32. Zhang, Y., Maekawa, T., Hara, T.: Using social media background to improve cold-start recommendation deep models. In: Proceedings of 2021 IEEE International Joint Conference on Neural Networks IJCNN, pp. 1–8 (2021)
33. Zhao, W.X., Li, S., He, Y., Chang, E.Y., Wen, J.-R., Li, X.: Connecting social media to e-commerce: cold-start product recommendation using microblogging information. IEEE Trans. Knowl. Data Eng. **28**(5), 1147–1159 (2015)

A Novel Deep Learning Model for Smartphone-Based Human Activity Recognition

Nadia Agti[1,2(✉)], Lyazid Sabri[1,3], Okba Kazar[4], and Abdelghani Chibani[3]

[1] Maths and Informatics Faculty, Mohamed El Bachir El Ibrahimi University, Bordj Bou Arreridj, Algeria
nadia.agti@univ-bba.dz
[2] LMSE, The Laboratory of Materials and Electronic Systems, Mohamed El Bachir El Ibrahimi University, Bordj Bou Arreridj, Algeria
[3] LISSI-The Laboratory of Images, Signals and Intelligent Systems, University Paris-Est Vitry-sur-Seine, Vitry-sur-Seine, Île-de-France, France
[4] University of Kalba, Sharjah, United Arab Emirates

Abstract. Traditional pattern recognition methods have shown improvements in recent years. However, these techniques relied on human intervention to identify crucial information within the data. Deep learning has revolutionized this scenario by enabling computers to learn from data autonomously. This capability is precious for comprehending how individuals interact with mobile and wearable technology. The growing popularity of this method stems from its capacity to function effectively with minimal or no human involvement. This study introduces a novel hybrid deep learning network that merges Convolutional Neural Network (CNN) and Multi-Layer Perceptron (MLP) layers. While maintaining the ability to perform accurate activity identification, this CNN-MLP approach captures the temporal features from sensors, facilitating, therefore, multi-class classification. We also explore various machine learning models, such as Random Forests (RF), K-Nearest Neighbors (KNN), Logistic Regression (LR), Long Short-Term Memory (LSTM), and CNN-LSTM, on two well-established datasets: the WISDM and UCI HAR datasets. Through extensive testing on these datasets, we showcase the superior accuracy of our proposed CNN-MLP model compared to other competing machine learning and deep learning models. Our research opens up new possibilities for precise and effective human activity recognition on smartphones.

Keywords: Human activity recognition · convolution neural network · multiLayer perceptron · smartphone · spatial-temporal knowledge

1 Introduction and Related Work

Human activity recognition (HAR) is a critical task in various domains, such as health [16], behavior analysis [20], transportation [6], and security [14].

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 231–243, 2024.
https://doi.org/10.1007/978-3-031-63992-0_15

Improving human-machine interactions and raising quality of life are significantly impacted by the capacity to identify and comprehend human behaviors automatically. Since they can develop hierarchical representations from unprocessed sensor readings, deep learning models have demonstrated impressive performance in tackling HAR issues. One of the most significant drawbacks of conventional methods employed in human activity recognition is their dependence on manual feature extraction. This approach demands substantial human labor and introduces subjectivity, potentially leading to the omission of crucial details. Moreover, these methods often need help to generalize effectively across diverse contexts, limiting their practical applicability.

Smartphone sensor-based human activity recognition harnesses the capabilities of diverse built-in sensors within cell phones to capture various facets of human activity, such as movements and gestures. Many researchers use accelerometers [5,8] to record acceleration across multiple directions. Gyroscopes [3] assess angular velocity, aiding in the detection of rotation and orientation changes. Magnetometers [7] gauge magnetic field intensity, facilitating orientation determination. Barometers [4] measure atmospheric pressure and enable the identification of elevation changes, such as ascending or descending stairs or navigating uphill. The Global Positioning System (GPS) [17] tracks outdoor activities and locations.

Numerous research endeavors have been conducted to explore diverse deep-learning approaches for HAR. Wan et al. [21] focused on comparing five algorithms for human behavior recognition: CNN, LSTM, BLSTM, MLP, and SVM. They underscored CNN's sustained significance as a premier classification and recognition system by presenting promising outcomes on two datasets. In a separate study, Moradi et al. [11] undertook a comprehensive evaluation of multiple algorithms for HAR, encompassing CNN, LSTM, Bidirectional LSTM, CNN-LSTM, as well as Logistic Regression and SVM with RBF kernel. Among these, CNN-LSTM and Bidirectional LSTM emerged as the most accurate techniques, while SVM with RBF kernel exhibited superior training efficiency. A balanced compromise between accuracy and training duration was identified in LSTM. Oluwalade et al. [15] used artificial neural networks (LSTM, Bi-LSTM, CNN, and Convolutional LSTM) to classify accelerometer and gyroscope data from smartphones and smartwatches, achieving over 91% average accuracy across 15 diverse activities, signifying substantial generalization improvements. Addressing real-time applications, Mutegeki et al. [12] introduced a CNN-LSTM approach for real-time Human Activity Recognition (HAR). Their integration of CNN and LSTM models resulted in superior performance compared to alternatives, with a notable accuracy increase of over 1% and a nearly 2% reduction in Softmax loss, demonstrating significant computational efficiency improvements.

To cater to the demand for wearable sensor-based HAR, Xu et al. [23] introduced the InnoHAR model, featuring max-pooling layers and concatenated convolution kernels. Their research consistently outperformed competitors across three datasets, highlighting its potential for real-time applications. In their paper, Xia et al. [22] proposed a unique LSTM-CNN model, combining convolu-

tional layers with LSTM. It achieved robust F1 scores on UC-HAR, WISDM, and OPPORTUNITY datasets, demonstrating superior performance and generalization. Mekruksavanich et al. [10] introduced a 4-layer CNN-LSTM network for HAR, surpassing basic LSTM networks with remarkable accuracy. Their study highlights the potential of CNN-LSTM models in human behavior identification tasks.

Our study represents a substantial contribution to the field of deep learning-based human activity identification. We harness the potential of deep learning to autonomously decipher how individuals interact with mobile and wearable devices, eliminating the need for human intervention in data pattern analysis. Our hybrid model, skillfully amalgamating Convolutional Neural Network (CNN) and Multi-Layer Perceptron (MLP) architectures, not only elevates activity identification accuracy but also establishes a new benchmark for hybrid models in this domain. By outperforming well-established machine learning and deep learning models on widely recognized datasets like WISDM and UCI HAR, our work underscores the ongoing progress in deep learning-based activity identification models.

The rest of this paper is organized as follows: Sect. 2 introduces our methodology, including dataset details, data preprocessing, and our proposed model. Section 3 presents experimental results, and Sect. 4 concludes the paper and outlines future work.

2 Methodology

In this section, we introduce the methodological framework utilized in this study, covering data collection and preprocessing, as well as our proposed architecture.

2.1 Data Collection and Preprocessing

We utilize in our study, two widely recognized and openly accessible datasets for sensor-based human activity recognition: the UCI HAR dataset [1] and WISDM dataset [2] for human activity recognition using smartphones.

Table 1 provides a comparison of the two datasets in terms of instances, attributes, subjects, activities, characteristics, tasks, sampling rate, sensors used, challenges, and file type.

We applied the data preparation principle to the WISDM dataset, such as label encoding, linear interpolation, normalization, segmentation, and one-hot encoding data preparation techniques. We incorporate encoded labels from the "activity" column into an "activity label" column. The labels are converted into numerical equivalents as detailed in Table 2.

We employ linear interpolation with missing data points (NaN). This technique fills in gaps by estimating values between existing data points, even though our dataset has few NaN values. Then, we normalize the characteristics of the training data to a range of 0 to 1 using the normalization equation:

$$Y^{(i)}_{\text{normalized}} = \frac{Y^{(i)} - Y^{(i)}_{\text{min}}}{Y^{(i)}_{\text{max}} - Y^{(i)}_{\text{min}}} \quad \text{for} \quad i = 1, 2, \ldots, n \tag{1}$$

Table 1. Comparison between UCI HAR Dataset and WISDM Dataset [13]

Factors	Datasets	
	UCI HAR Dataset	WISDM Dataset
# examples	10299	1098209
# attributes	561	6
# subjects	30	36
# activity types	6	6
Characteristics	Multivariate/Time series	Multivariate/Time series
Tasks	Classification/Clustering	Classification
Sampling Rate	50 Hz	20 Hz
Sensors types	Accelerometer, Gyroscope	Accelerometer
Challenges	Multimodal	Class Imbalance
File Type	csv	txt

Table 2. Activity labels for WISDM datset

Activity	Label
Downstairs	0
Jogging	1
Sitting	2
Standing	3
Upstairs	4
Walking	5

where n represents the number of channels, and $Y_{\max}$ and $Y_{\min}$ correspond to the maximum and minimum values of the i-th channel, respectively.

We also reshape the data frame; each record's length is equal to 80 steps. We define a function called *segments* that takes the data frame, label names, and three input parameters. The *x_train* and *y_train* functions then segregate features and labels into separate sets. We employ the *reshape* function to convert the two-dimensional data into a list, preparing it for model input.

Additionally, all elements are converted to the *float32* format to ensure compatibility with the model's requirements.

One-hot encoding enhances machine learning algorithms by transforming categorical variables into a suitable format. Each category is represented by a binary vector, where only one element is set to 1 (hot), and the rest are 0 (cold). The length of the binary vector matches the number of distinct categories. The labels undergo one-hot encoding and are stored as *y_train_hot*, concluding the data preparation process.

The UCI HAR dataset is also pre-processed. We start by looking for duplicates and empty fields. There are no duplicates or null values in the data collection. After receiving the train data from X_train.txt and subject_train.txt, and the test data from X_test.txt and subject_test.txt, respectively, we save the train and test data in two CSV files. Feature names are then stripped of any extra chanting.

The two datsets are split, with 80% for training, and 20% for testing the model.

2.2 Proposed Model Architecture

The architecture of our CNN-MLP model is designed to effectively capture temporal attributes from the input data, facilitating multi-class classification. The model consists of distinct layers, each serving a specific feature extraction and classification role. This structure is well-suited for tasks requiring concurrent sequential data processing, such as time-series sensor data. The complete structure of our approach is illustrated in Fig. 1, which visually portrays its crucial components.

Our CNN-MLP model, designed for smartphone-based human activity recognition, utilizes a series of three 1D Convolutional layers to efficiently extract temporal features. In the realm of time series analysis, 1D convolutions are well-known for identifying intricate temporal patterns. Within our model, these 1D convolutions serve a central role, progressively unveiling patterns within data segments and constructing a hierarchical representation that mirrors task complexity. The identified temporal features are then seamlessly fed into the fully connected Multi-Layer Perceptron (MLP) layers for classification.

Our MLP layers, consisting of two dense tiers enriched with Rectified Linear Unit (ReLU) activation functions, meticulously analyze dynamic temporal patterns within sensor data. This process enables the recognition of intricate interdependencies across successive time steps. We've incorporated Global Average Pooling (GAP) and Batch Normalization. GAP consolidates temporal dynamics, enhancing generalization and diminishing sensitivity to minor sensor fluctuations. Meanwhile, Batch Normalization ensures consistent activations within mini-batches, addressing gradient challenges during training.

The output layer, activated with softmax, predicts human behavior based on the extracted temporal features. Each neuron within this densely connected output layer represents a distinct activity class, generating a probability distribution signifying the likelihood of each activity.

To facilitate training convergence and optimization, our model implements learning rate scheduling techniques, adjusting the learning rate during training epochs. This dynamic adjustment enhances the model's efficiency and promotes effective convergence. The choice of the Adam optimizer and categorical cross-entropy loss function is aimed at multi-class classification, ensuring the model minimizes the loss while assessing performance through accuracy.

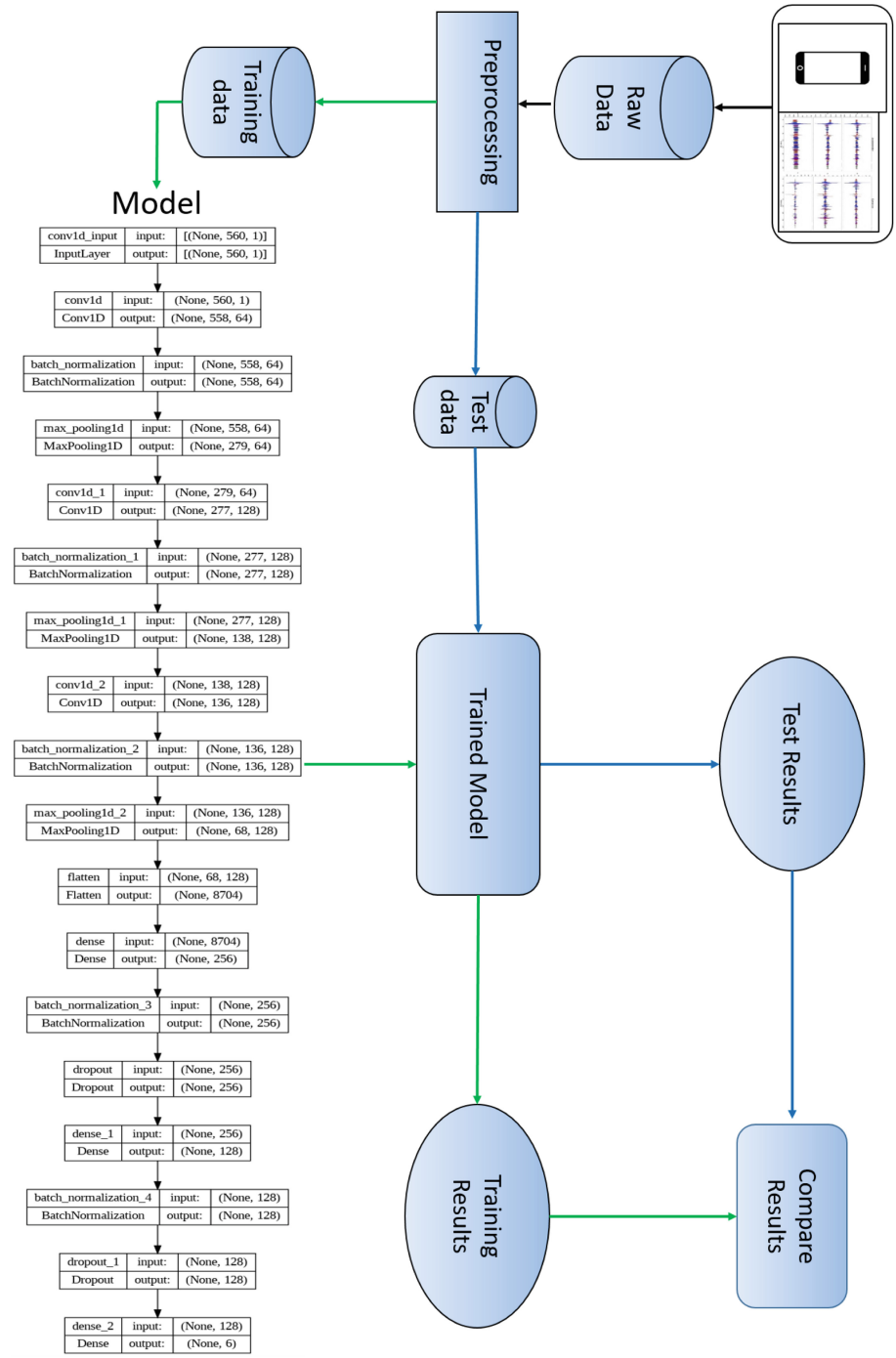


Fig. 1. Architecture of the proposed approach

3 Experimental Results and Discussion

This section showcases test outcomes and engages in insightful discussions. We commence by elucidating the hardware and software configurations employed to construct our model. Subsequently, we introduce the measurement metrics harnessed for evaluating our model's performance. Proceeding from there, we provide comprehensive explanations and discussions of the outcomes stemming from our two conducted experiments, and finally, we present a comparison of our model with state of the art machine learning models. The Code for our experiment is available online[1].

3.1 Model Implementation

We leveraged Python's Keras, a versatile neural network API, for architectural development. Keras seamlessly integrates Tensorflow or Theano as backends, and we deliberately chose Tensorflow for its capacity to accelerate computations and utilize GPU resources, greatly enhancing computational efficiency.

Our hardware setup was a robust server hosted on Google Colab, featuring a single-core 2.20 GHz Intel(R) Xeon(R) CPU with 16 GB of RAM. This powerful environment was instrumental in our experiments and analysis.

3.2 Evaluation Metrics

The confusion matrix and classification reports serve as the assessment tools within this study. The confusion matrix comprehensively evaluates the model's performance for each class by comparing predicted and actual values. Its components include True Positive, True Negative, False Positive, and False Negative. Table 3 depicts a generic representation of a confusion matrix's architecture.

Metrics such as precision (Eq. 2), recall (Eq. 3), F1 score (Eq. 4), and accuracy (Eq. 5) are featured in the classification report.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

Table 3. Simplified binary classification confusion matrix

	Positive Class	Negative Class
Predicted Positive	True Positive (TP)	False Positive (FP)
Predicted Negative	False Negative (FN)	True Negative (TN)

[1] https://github.com/NadiaAGTI/New-CNN-MLP-model-for-Human-Activity-Recognition.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{5}$$

3.3 Results and Discussion

We tested our model using four epoch values: 10, 50, 100, and 200 for both datasets. To ensure fairness and consistency in the subsequent comparison, we validated the outcomes from previous steps using the Accuracy metric.
The primary reason for the efficacy of our model lies in its thoughtful architecture. To achieve these remarkable results, the combination of a Convolutional Neural Network (CNN) and a Multi-Layer Perceptron (MLP) has proven to be essential. The CNN's ability to accurately identify human actions critically depends on its capacity to detect temporal patterns in sequential data, particularly time series data. This architecture excels in identifying intricate patterns, which is a fundamental requirement for robust activity recognition.

Furthermore, the excellent results are, in large part, due to careful data preparation. Methods like batch normalization and data normalization have been applied with care. These processes enhance the quality of the input data, reduce noise, and ensure consistency, all of which are essential for the model to successfully identify human activity.

The robust performance of our model is also a result of careful hyperparameter optimization. Parameters such as the number of filters, kernel sizes, activation functions, and dropout rates, among other factors, have all been finely tuned. Additionally, the inclusion of L2 regularization has played a crucial role in preventing overfitting and ensuring the model's effective generalization to new data.

Another essential element of the design involves the incorporation of batch normalization layers. They expedite both the model's convergence and training stability. Consequently, the training process becomes more resilient and efficient, and it is less dependent on parameter initialization.

Pooling layers, especially the max-pooling within the CNN component, effectively reduce the spatial dimensions of the data while retaining crucial features. This is highly beneficial when dealing with time series data, as it allows the model to focus on the most significant information. Finally, overfitting has been effectively mitigated through the cautious implementation of dropout layers within the MLP component. This technique randomly deactivates neurons during training, encouraging the model to capture more robust and transferable features, ultimately enhancing its overall performance.

Collectively, these design principles and components have yielded a model that consistently delivers outstanding performance on the datasets WISDM and UCI HAR. It achieves high recall, accuracy, F1-scores, precision, and overall

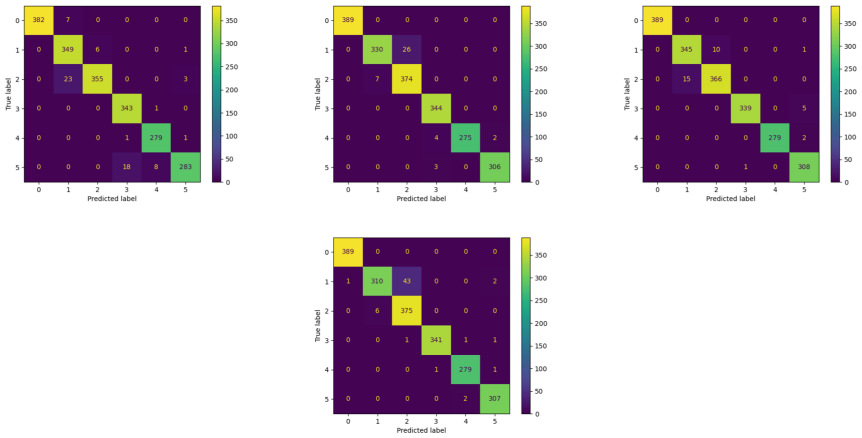


Fig. 2. Confusion matrix for the proposed CNN-MLP model with different epoch values with UCI dataset

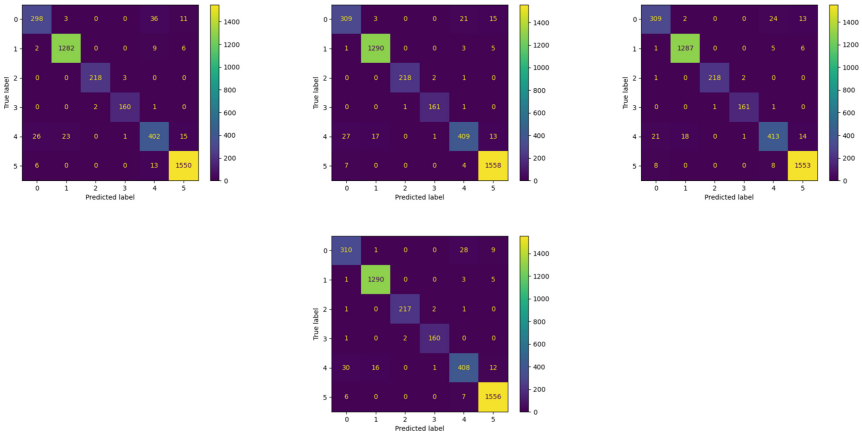


Fig. 3. Confusion matrix for the proposed CNN-MLP model with different epoch values with WISDM dataset

accuracy across various activity classes. Consequently, our model offers a versatile and adaptable solution for human activity recognition, with potential applications in diverse industries, including wearable technology and healthcare.

Figures 2 and 3 illustrates the confusion matrix of the model for four distinct epoch values using the UCI HAR dataset and WISDM dataset respectively.

The outcomes of the model with both datasets are presented in Table 4.

Table 4. The proposed CNN-MLP model results using different epochs values

Dataset	Metrics(%)				
	Epochs	Precision	Recall	F-Score	Accuracy
WISDM Dataset	10	96.09	96.14	96.11	96.14
	50	96.96	97.00	96.97	97.00
	100	96.86	96.90	96.88	96.90
	200	96.87	96.90	96.88	96.90
UCI HAR Dataset	10	89.81	85.92	85.37	85.92
	50	95.99	95.63	95.61	97.96
	100	97.24	97.18	97.18	97.18
	200	97.29	97.14	97.12	97.14

3.4 Comparison with Baselines and State-of-the-art

As illustrated in Table 5, the performance comparison results of the study provide compelling insights into the effectiveness of our proposed CNN-MLP model for human activity recognition. The comparison encompasses a range of established baseline models, including Random Forests, Logistic Regression, K-Nearest Neighbors, Multi-Layer Perceptron, Long Short-Term Memory, and CNN-LSTM, conducted on both the WISDM and UCI HAR datasets.
Our proposed CNN-MLP model demonstrated superior performance to all other baseline models, achieving an impressive accuracy of 96.9% on the WISDM dataset. This achievement stands out notably in contrast to RF, which attained an accuracy of 38.52%, highlighting the substantial distinction between conventional techniques and modern deep learning methodologies. Furthermore, the suggested model significantly outperformed both LR (90%) and KNN (75.04%), underscoring its ability to discern intricate temporal patterns within the sensor data.

Table 5. Comparison of our model with baseline models on both WISDM and UCI datasets(Accuracy in %)

Methods	WISDM dataset	UCI HAR dataset
RF [19]	38.52	86
LR [11,24]	90	96.1
KNN [18,24]	75.04	89.1
MLP [9,21]	91.7	86.83
LSTM [12,19]	87.7	91.28
CNN-LSTM [12,19]	86.3	92.13
Proposed model	**96.9**	**97.14**

Turning our attention to the UCI HAR dataset, a similar trend emerges. Our CNN-MLP model continues to outshine all other methods, boasting an exceptional accuracy of 97.14%. KNN achieved an accuracy of 89.1%, and LR reached 96.1%. Impressively, our model surpassed LSTM (91.28%), and CNN-LSTM (92.13%), which inherently consider sequential patterns. This notable advantage underscores our proposed architecture's effectiveness in effectively capturing local and global information.

4 Conclusion

The distinctive CNN-MLP model proposed in this research is tailored for smartphone sensor-based human activity recognition. This innovative approach integrates convolutional neural network (CNN) layers and multi-layer perceptron (MLP) layers, skillfully leveraging different convolutional kernel sizes and incorporating well-considered max-pooling layers. Our suggested approach consistently exhibits impressive performance advantages in comprehensive comparisons against baseline models and state-of-the-art techniques. Moreover, its robust generalization capabilities are demonstrated across two widely recognized public datasets, further affirming its potential for real-world applications.

In our forthcoming research, we aim to enhance the compatibility of our system with mobile contexts. This endeavor will involve exploring hardware-efficient implementations and robust cloud access mechanisms to ensure seamless usability on mobile devices.

References

1. Human activity recognition using smartphones dataset. https://archive.ics.uci.edu/dataset/240/human+activity+recognition+using+smartphones. Accessed 05 July 2023
2. Wisdm smartphone and smartwatch activity and biometrics dataset. https://www.cis.fordham.edu/wisdm/dataset.php. Accessed 05 July 2023
3. Biswas, D., et al.: Recognizing upper limb movements with wrist worn inertial sensors using k-means clustering classification. Hum. Mov. Sci. **40**, 59–76 (2015)
4. Chernbumroong, S., Cang, S., Atkins, A., Yu, H.: Elderly activities recognition and classification for applications in assisted living. Expert Syst. Appl. **40**(5), 1662–1674 (2013)
5. Chernbumroong, S., Cang, S., Yu, H.: A practical multi-sensor activity recognition system for home-based care. decision support systems **66**, 61–70 (2014)
6. Choujaa, D., Dulay, N.: Activity recognition from mobile phone data: state of the art, prospects and open problems. Imp. Coll. Lond. **5**, 32 (2009)

7. Gjoreski, H., Gams, M.: Activity/posture recognition using wearable sensors placed on different body locations. Proc. (738) Signal Image Process. Appl. Crete Greece **2224**, 716724 (2011)
8. Guo, J., Zhou, X., Sun, Y., Ping, G., Zhao, G., Li, Z.: Smartphone-based patients' activity recognition by using a self-learning scheme for medical monitoring. J. Med. Syst. **40**, 1–14 (2016)
9. Kwapisz, J.R., Weiss, G.M., Moore, S.A.: Activity recognition using cell phone accelerometers. ACM SIGKDD Explor. Newsl. **12**(2), 74–82 (2011)
10. Mekruksavanich, S., Jitpattanakul, A.: LSTM networks using smartphone data for sensor-based human activity recognition in smart homes. Sensors **21**(5), 1636 (2021)
11. Moradi, B., Aghapour, M., Shirbandi, A.: Compare of machine learning and deep learning approaches for human activity recognition. In: 2022 30th International Conference on Electrical Engineering (ICEE), pp. 592–596. IEEE (2022)
12. Mutegeki, R., Han, D.S.: A CNN-LSTM approach to human activity recognition. In: 2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), pp. 362–366. IEEE (2020)
13. Nayak, S., Panigrahi, C.R., Pati, B., Nanda, S., Hsieh, M.Y.: Comparative analysis of HAR datasets using classification algorithms. Comput. Sci. Inf. Syst. **19**(1), 47–63 (2022)
14. Niu, W., Long, J., Han, D., Wang, Y.F.: Human activity detection and recognition for video surveillance. In: 2004 IEEE International Conference on Multimedia and Expo (ICME) (IEEE Cat. No. 04TH8763), vol. 1, pp. 719–722. IEEE (2004)
15. Oluwalade, B., Neela, S., Wawira, J., Adejumo, T., Purkayastha, S.: Human activity recognition using deep learning models on smartphones and smartwatches sensor data. arXiv preprint arXiv:2103.03836 (2021)
16. Preuveneers, D., Berbers, Y.: Mobile phones assisting with health self-care: a diabetes case study. In: Proceedings of the 10th International Conference on Human Computer Interaction with Mobile Devices and Services, pp. 177–186 (2008)
17. Reddy, S., Mun, M., Burke, J., Estrin, D., Hansen, M., Srivastava, M.: Using mobile phones to determine transportation modes. ACM Trans. Sens. Netw. (TOSN) **6**(2), 1–27 (2010)
18. Semwal, V.B., Lalwani, P., Mishra, M.K., Bijalwan, V., Chadha, J.S.: An optimized feature selection using bio-geography optimization technique for human walking activities recognition. Computing **103**(12), 2893–2914 (2021)
19. Trabelsi, I., Françoise, J., Bellik, Y.: Sensor-based activity recognition using deep learning: a comparative study. In: Proceedings of the 8th International Conference on Movement and Computing, pp. 1–8 (2022)
20. Vepakomma, P., De, D., Das, S.K., Bhansali, S.: A-Wristocracy: deep learning on wrist-worn sensing for recognition of user complex activities. In: 2015 IEEE 12th International conference on wearable and implantable body sensor networks (BSN), pp. 1–6. IEEE (2015)
21. Wan, S., Qi, L., Xu, X., Tong, C., Gu, Z.: Deep learning models for real-time human activity recognition with smartphones. Mob. Netw. Appl. **25**, 743–755 (2020)

22. Xia, K., Huang, J., Wang, H.: LSTM-CNN architecture for human activity recognition. IEEE Access **8**, 56855–56866 (2020)
23. Xu, C., Chai, D., He, J., Zhang, X., Duan, S.: InnoHAR: a deep neural network for complex human activity recognition. IEEE Access **7**, 9893–9902 (2019)
24. Zaki, Z., Shah, M.A., Wakil, K., Sher, F.: Logistic regression based human activities recognition. J. Mech. Contin. Math. Sci. **15** (2020)

AIMSafe: EEG-Based Driver Behavior Understanding via Attention and Incremental Learning Mechanisms

Landu Jiang[1], Cheng Luo[1], Tao Gu[2], Kezhong Lu[1], and Dian Zhang[1(✉)]

[1] Shenzhen University, Shenzhen 518060, China
zhangd@szu.edu.cn
[2] Macquarie University, Sydney, NSW 2109, Australia

Abstract. In this paper, we propose *AIMSafe*, an electroencephalographic (EEG) based system that studies driver in-vehicle behaviors leveraging **A**ttention networks and **I**ncremental learning **M**echanism for road **Safe**ty. Instead of using predefined classes, we categorize driver in-vehicle activities into different risk levels - a stronger motion may have a higher chance of unsafe driving. More specifically, we first employ a CNN based model to distinguish two basic activities - 1. normal driving and 2. unsafe driving. Moreover, *AIMSafe* also leverages smartphone IMU sensors generating soft hints that helps automatically label EEG data on road. We then adopt class-incremental learning to rank other Out-of-Distribution (OOD) driver activities (safe to unsafe) based on the Mahalanobis distance. A modified Squeeze-and-Excitation (SE) block is also used to adaptively select effective EEG electrodes for improving the system efficiency. Evaluation results (involving 11 males and 4 females) show that *AIMSafe* could achieve a detection accuracy over 95% on unsafe driving activities using only 4 electrodes.

Keywords: Driving Safety · Driver Education · Wearable Sensing · EEG · Smart computing

1 Introduction

Driver inattention/distraction is a major factor causing a great number of traffic accidents every year. Driven by a pressing need to enhance road safety, many advanced systems have been developed as high-tech features in today's cars to monitor driver status by using specialized cameras or infrared sensors [1]. What's more, efforts have been made to explore smart sensing technologies for safe driving study and traffic congestion analysis [7,28]. For example, wrist-worn devices (e.g., smartwatch) are used for driver hand motion tracking and forearm posture recognition [6,14,25]. In addition, recent Wi-Fi CSI (channel state information)

L. Jiang and C. Luo—The first two authors contributed equally to this paper.

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 244–255, 2024.
https://doi.org/10.1007/978-3-031-63992-0_16

and acoustic (ultrasonic) based systems have seen great success in human in-vehicle activity recognition [20,24].

On the other hand, electroencephalogram (EEG) has been gaining great attention in various fields such as medical diagnosis, cognitive enhancement, entertainment and education. EEG devices have become more lightweight, wearable, affordable and efficient, which increased the popularity and acceptance in different applications. A number of EEG based driving safety systems have been proposed, they mainly focus on the detection of unsafe/inattentive driving periods like drowsy and non-drowsy [8,10,30], distracted and non-distracted [27]. Though some work used EEG signals to indentify visual distracting conditions such as reading smartphone [5], viewing navigation [26] as well as visual/cognitive [17]. However, there are still a great number of common in-vehicle activities that may cause unsafe driving, while existing work can not address them all due to limited types of predefined activities.

Moving along this direction, instead of classifying EEG signals elicited by predefined driver behaviors, we aim to provide a more general model that evaluates driver behaviors with different driving risk levels. It is based on the hypothesis that a driver will have stronger movements when encountering unsafe driving conditions (e.g., reaching the back seat), which may result a more noticeable stimuli in EEG patterns. While drivers may perform various motion styles (e.g., direction and speed) that imprint unique hallmarks in EEG signals. We are unable to label all possible types of events/behaviors for classification/recognition. Alerts based on risk levels could give drivers proper feedbacks and help them shape good driving habits, and thus avoid potential traffic accidents.

To achieve the goal, in this paper we propose *AIMSafe*, an EEG based driver **Safe**ty system that leverages **A**ttention networks and **I**ncremental learning **M**echanism. Specifically, we first use a Convolutional Neural Network (CNN) model to indentify two basic driver activities - 1. holding the steering wheel as normal driving - the least significant movement (safe), and 2. rotate the head and arm to reach the back seat - the most significant movement (unsafe). *AIMSafe* also utilizes smartphone IMU sensors that generates soft hints to detect vehicle turning events. We then adopt an incremental learning method using Mahalanobis distance to adaptively rank any other new in-vehicle activities without further model training [16]. Furthermore, *AIMSafe* employs a modified attentive squeeze-and-excitation (SE) block producing a collection of per-channel modulation weights. The main contributions of this paper are summarized as follows:

1. We propose *AIMSafe*, an EEG based approach for understanding driver behaviors in a cost-effective way. The system is able to detect and alert different types of most commonly occurring in-vehicle driving activities based on risk levels, which can be used in training courses to educate and help driver shape good driving habits.
2. We explore time-frequency features of EEG signals and propose a CNN model with modified attentive Squeeze-and-Excitation Networks to recognize basic in-vehicle activities and perform channel (electrode) selection. An adaptive

turning event labelling method is also provided by piggybacking on vehicle dynamic sensing using smartphone IMU sensors. We then employ the Mahalanobis distance-based incremental learning method to further indentify new OOD samples.

3. We design and build experimental environments involving 15 participants in stationary and moving vehicles. In the evaluation, AIMSafe achieved the accuracy of 100% using 4 electrodes for basic-activity classification, while the overall detection accuracy of OOD samples is over 95%. Which demonstrate that the proposed solution is robust and effective to indentify in-vehicle unsafe activities and has great potential to promote driving safety.

The reminder of the paper is organized as follows. Section 1 discusses the motivation and design considerations of implementing the proposed system. Section 2 gives a brief description of AIMSafe design and implementation details of the system. Sections 3 presents the experiments in real vehicles as well as the evaluation results, we further discusses how this work can be extended and how to improve the system performance. Section 4 reviews the related work on driver distraction study and Sect. 5 concludes the paper.

2 System Design

2.1 System Overview

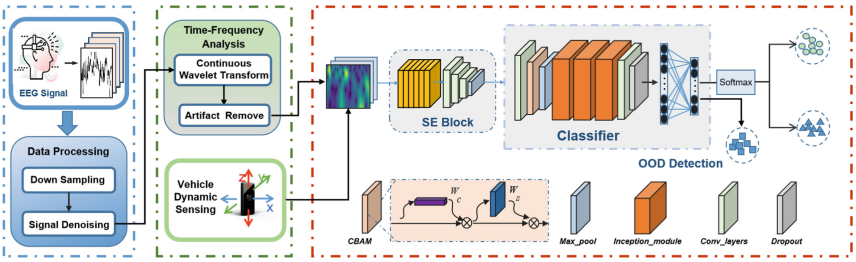


Fig. 1. System Overview

In this paper, instead of classifying a limited number of pre-defined driving event, we aim to help drivers to be aware of the risk levels of their in-vehicle behaviors. We divide these in-vehicle activities into 3 categories: 1. safe - normal driving, 2. potential dangerous - Secondary Tasks (unsafe when it takes more than 3 seconds), 3. dangerous - unsafe/strong motion. More specifically, AIMSafe is comprised of several key components including **Data collection and preprocessing**, **Time-frequency analysis and vehicle dynamic sensing**, **Human motion spotting and classification** and **Driver activity classification and OOD sample identification**. As illustrated in Fig. 1, these four components work together to continuously monitor the driver's in-vehicle activities and identify any instances of distracted or unsafe behavior. The system uses advanced signal processing and a lightweight version of Inception CNN mode to analyze EEG signals and provide real-time alerts to the driver if necessary.

2.2 EEG Data Collection and Preprocessing

Data Formatting. To improve the computational efficiency of driving distraction detection, we need to select the most representative channels from the 32 EEG electrodes available. Based on EEG studies of driver behavior analysis [2,21], we have selected 16 electrodes for this purpose, including FP1, F7, FZ, FC2, T7, C3, C4, CZ, CP1, CP2, P7, F5, PZ, O1, FC4, and O2. We will further reduce this number to 8 using a modified SE block in the classification module, and then further down to 4, and finally to 2. We will compare the performance of different sets of electrode selections.

To record the data, we have used a sliding window of 2 seconds with a 1-second stride at a sampling rate of 250 Hz. Moreover, we also leverage driver smartphone IMU sensors to detect turning events, while at the same time automatically label the EEG data during this time period. Then we are able to measure a baseline of target driver motion strength (turning the steering wheel is a necessary task to operate a vehicle).

Time-Frequency Analysis. EEG data is a collection of non-stationary time series corresponding to measurements of human brain activities across different frequency bands over each electrode, thus a transformation to the time-frequency domain is necessary for feature extraction. For the non-stationary time series, in signal processing, comprises a set of techniques studying signal representation in time-frequency domains. Among various methods, Wavelet transform (WT) outperformed others in recent CSI based human motion detection research [19]. WT methods (known as discrete or continuous WT) allow multi-resolution signal analysis at different scale components by using a mother wavelet function. It shows excellent advantages for the analysis of transient signals like EEG, Wi-Fi CSI, etc. In this paper, we use cgau8 as the kernel and set scale range 1–64, thus output a 64×500 $(250 \times 2) \times 16$ (channels) data matrix (spectrogram) of each window. In addition, since the eye states information is also very important for driver behavior analysis, we extract it as back up data/hints for the future study. After eye movements removal, the data size become 48 (frequency) $\times$ 500 (time/samples) $\times$ 16 (channels).

2.3 Human Motion Spotting

Driver Specific Feature Extraction with SE Block. With the increase in the depth and width of neural networks, the number of parameters and computational cost has also increased. To address this issue, we utilize the Squeeze-and-Excitation (SE) block [11] to address the issue of channel reduction. The SE block is a lightweight and efficient module that can be easily integrated into existing neural networks. The block consists of two operations: Squeeze and Excitation, as shown in Fig. 2. The Squeeze operation takes an input feature map and generates a channel descriptor by aggregating global spatial information using global average pooling. The Excitation operation then takes the channel descriptor as input and produces a collection of per-channel modulation weights.

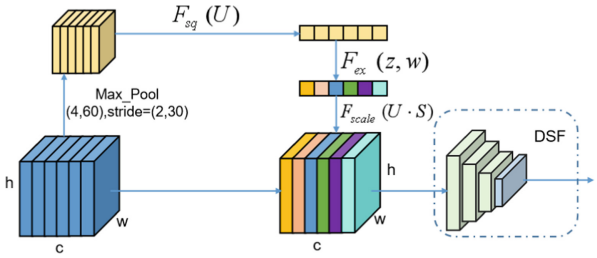


Fig. 2. SE BLock based Driver Specific Feature extraction

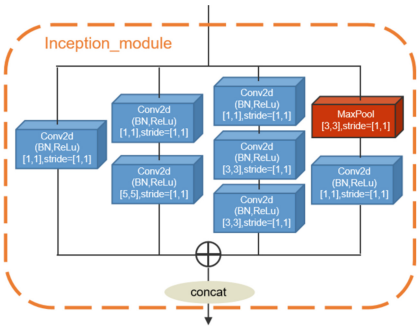


Fig. 3. Inception Network Structure

While traditional SE block has been successfully used in various deep learning models and EEG-based studies, it cannot be directly applied for EEG channel reduction as the existing SE block structure is designed to adjust the weights of feature maps (e.g., the output of a convolution layer), which erases the tags of EEG channels (electrodes). Although we can remove the transformation and directly connect the input matrix to SE block, the global pooling operation used to generate the channel descriptors in SE block may cause loss of spatial information (H×W), which is important for EEG analysis due to the distribution of brain signals following a Gaussian distribution with the global average value close to zero. To address this issue, we propose to use a Max Pooling layer to extract local maximums as features and then feed them to the SE block. The weights of each EEG channel can then be updated during training via backpropagation.

Inception Based Classification Module. We evaluated several networks with varying numbers of layers to draw conclusions, which were largely based on empirical observations. After thorough experimentation, we arrived at a modified, lightweight version of the Inception V3 model shown in Fig. 3. Which was achieved through modifications of the previous Inception Networks [22]. Additionally, it has been shown that the Squeeze-and-Excitation (SE) block can

improve performance when integrated with Inception models. Therefore, we plan to explore the use of this network in future work for EEG signal analysis.

2.4 Mahalanobis Distance Based Incremental Learning

During real driving scenario, the method of combining incremental learning with Mahalanobis distance can dynamically update the hazard rating to adapt to the constantly changing environment while accumulating new data. To further distinguish different driver behavior based on their motion levels, we consider measuring the Mahalanobis distance from the output of the penultimate layer of Neural Network. We get the Gaussian distribution of basic category from the pre-trained softmax neural classifier, Therefore, we could compute the empirical class mean $\hat{\mu}_c = \frac{1}{N} \sum_{i:y_i=c} f(x_i)$ and covariance of training samples as:

$$Cov_c = \sum_{i:y_i=c} (f(x_i) - \hat{\mu}_c)(f(x_i) - \hat{\mu}_c)^T \tag{1}$$

where $f(x_i)$ is the latent output of neural network, $\hat{\mu}_c$ is the mean of training samples with label c.

Using Gaussian distribution of basic categories, we define $M(x)$ between test sample x and existing label c to estimate the degree of danger.

$$M(x) = (f(x) - \hat{\mu}_c)^T Cov_c (f(x) - \hat{\mu}_c) \tag{2}$$

where N is the number of training samples of each class with label C. By using the mean vector and covariance matrix above, we are able to calculate the Mahalanobis distance of each OOD driver behavior to the basic classes. We then input the EEG data of driver doing steering wheel turning (automatically recorded) to calculate the Mahalanobis distance between normal driving and unsafe driving - feature extracted by the Inception model. The mean value of the distance can be used as the boundaries to categorize driver activities into different groups based on their motion level.

3 Evaluation

3.1 Experimental Setup and Data Collection

We use a wireless EEG headset from Guger Technologies (model name: g.Nautilus RESEARCH Headset) to collect brain activity data. We recruited 15 participants - 4 females and 11 males from age 19 to 50. Each participant was asked to perform 4 target types of activities in real vehicles. When recording the data of normal driving behavior, participants were asked to drive the vehicle at a lower speed (less than 30 KM/h) on the same selected driving route (2.0 KM). In contrast, when participants are required to perform some unsafe driving

behaviors, they will sit in the front passenger's seat (with a steering wheel of a simulator), record data on the driving vehicle, and simulate the driver's behavior in the real road as much as possible. In summary, one participant was asked to perform at least 200 times of each type of gesture (100 in the parking lot and 100 on real road). The cross validation (60% training and 40% testing) is also applicable due to the limited number of samples.

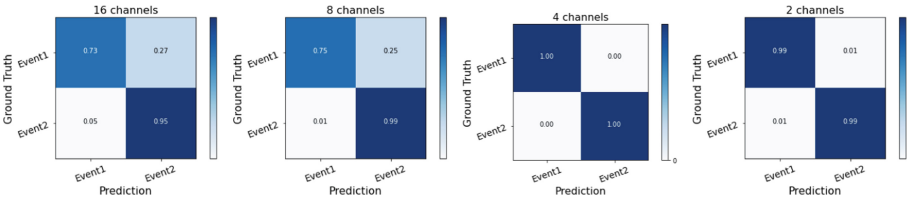


Fig. 4. Model performance on base classes with 16, 8, 4 and 2 channels

3.2 Unsafe Driving Detection

In this section, we provide a comprehensive analysis to examine the performance of proposed models. We also define the metrics including True Positive (number of driving activities that are correctly identified), False Positive (number of driving activities are wrongly classified) and detection rate/accuracy (number of correctly detected testing samples).

Base Class Recognition. Figure 4 shows that the proposed model using 4 ('FC2', 'CP1', 'CP2' and 'F5') and 2 ('FC2' and 'CP2') channels/electrodes achieved the very high average accuracy (over 99%) distinguishing two base classes. Though two base motion classes are significant different in real physical world, noise and diverse among drivers under different driving conditions (as we claimed) resulting complex EEG signal patterns - lower accuracy in 16 and 8 channels. According to these results, we select 4 channel (highest accuracy) as the default setting for later test since it can be more stable if there is a variety of new challenging scenarios (vehicles and drivers) in the future.

Incremental Learning Result Analysis. In Fig. 5, we present the outputs of Mahalanobis distance based method with 2 selected participants (1 male and 1 female). As we can see from 2 figures, the distance distributions of each participant are similar but at the same time the value patterns are quite different - one may have stronger motion and other may have more frequent movements. For instance, "Turnback" is consistently one of the actions with the highest magnitude and highest risk across all the charts. Meanwhile, we can observe that normal turning events, indicated by the baseline (blue), effectively differentiate between safe (holding the steering wheel and secondary tasks) and potentially

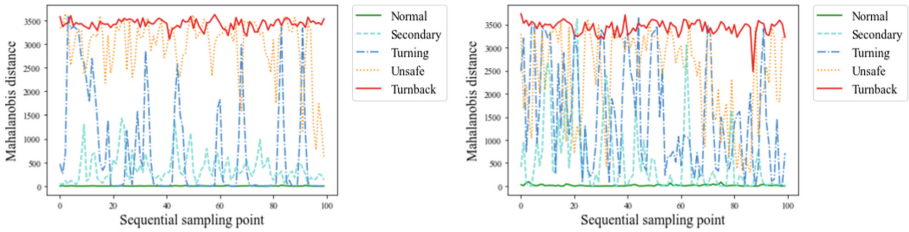


Fig. 5. Mahalanobis distance of each behavior from selected participants (Color figure online)

unsafe events. While some users exhibit clear data curves, while for others, the curves are blurry. Therefore, more effective machine learning methods are needed to differentiate them.

Table 1. 2-second Mahalanobis Distance

People	Event							
	Normal		Secondary		Unsafe		Turnback	
	P	D	P	D	P	D	P	D
user1	0.00	0.03	0.32	0.51	0.94	0.99	1.00	1.00
user2	0.00	0.00	0.64	0.75	0.96	0.99	1.00	1.00
user3	0.02	0.02	0.68	0.75	0.98	1.00	1.00	1.00
user4	0.05	0.05	0.46	0.48	0.93	0.96	1.00	1.00
user5	0.00	0.00	0.02	0.08	0.96	0.99	1.00	1.00
user6	0.00	0.00	0.21	0.43	0.99	1.00	1.00	1.00
Average	0.01	0.02	0.39	0.50	0.96	0.99	1.00	1.00

What's more, we test and compare the performance of learning model using Mahalanobis Distance and Euclidean Distance for each user. As illustrated in Table 1 and Table 2, we choose a 2-second time window for driving event detection, as the shorter time window could alert driver earlier (3 seconds rule). Where P and D in tables mean potential dangerous (counting 3 seconds) and dangerous. We can see that the Normal driving events never exceed the line of dangerous and sometimes detected as potential dangerous. For secondary tasks, the detection rate is from 30% to 90% as a potential dangerous task. It is because that drivers has to move their body and arms back and forth due to different tasks, the Mahalanobis distance are not always beyond the threshold and over the dangerous line occasionally (may be False Positive). And for unsafe behaviors and turnback events, our method could achieve a high detection rate (over 95% on average).

Table 2. 2-second Euclidean Distance

People	Event							
	Normal		Secondary		Unsafe		Turnback	
	P	D	P	D	P	D	P	D
user1	0.00	0.00	0.22	0.17	0.88	0.86	1.00	1.00
user2	0.00	0.00	0.57	0.50	0.98	0.98	1.00	1.00
user3	0.00	0.00	0.43	0.28	0.93	0.81	1.00	1.00
user4	0.01	0.00	0.38	0.22	0.91	0.79	1.00	1.00
user5	0.00	0.00	0.00	0.00	0.97	0.96	1.00	1.00
user6	0.00	0.00	0.18	0.26	1.00	1.00	1.00	1.00
Average	0.00	0.00	0.30	0.24	0.95	0.90	1.00	1.00

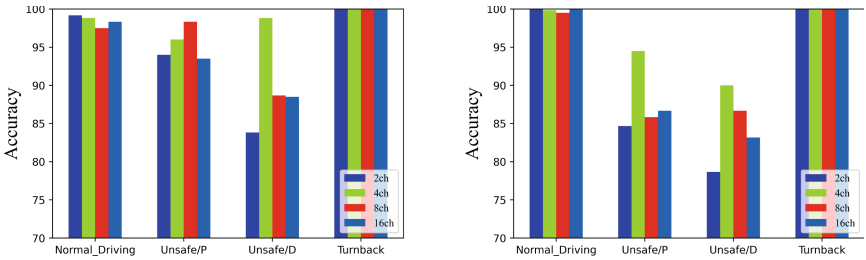


Fig. 6. Accuracy based on Mahalanobis distance and Euclidean distance

Moreover, we also evaluated the average performance of two distance metrics - Mahalanobis distance and Euclidean distance, in detecting unsafe driving activities using different channel settings with the time window length - 2 seconds. The detection accuracy comparison between the two metrics is presented Fig. 6. It can be observed from the figures that the system accuracy of using Mahalanobis distance is higher than using Euclidean distance for detecting unsafe driving activities, especially in the case of using 4-channel settings. This suggests that Mahalanobis distance is a more suitable metric for accurately detecting unsafe driving activities using EEG signals, as it takes into account the correlation among the EEG channels. It highlight the importance of choosing an appropriate distance metric when analyzing EEG signals for detecting unsafe driving activities, and we will take the Mahalanobis distance as the criterion in the subsequent experiments.

4 Related Work

In this section, we focus on reviewing mainstream schemes on smart sensing-based approaches for driving activity (distractions) monitoring. By taking the advantage of computer vision techniques, there has been active research work for detecting inattentive or distracted driving problems. Jiang et al. [13] propose a

smartphone based system to detect unsafe events at intersections such as running red lights, running stop signs, unsafe turns, etc. Moreover, a number of vision based systems leveraging the deep learning models are presented for distraction driving detection [3,23]. Meanwhile, there are several smart-sensor based systems tracking driver hands position for preventing unsafe driving activities [9]. Bi et al. [6] presented SafeWatch, a system using smartphone and smartwatches to estimate the posture of the driver's forearms and hands. Johnson et al. [15] also works on locating a smartphone in the vehicle using the acquired motion data when a user is holding it. Jiang et al. [14] focuses on driver's right hand motion to recognize secondary tasks and potential dangerous behaviors.

Many device-free systems using wireless or acoustic signals for driver behavior monitoring have been offered. Xu et al. [29] leveraged existing audio devices on smartphones to realize early recognition of inattentive driving events. Jiang et al. [12] proposed DriverSonar, a system that targets driver head motions such as nodding, yawning and adjust abnormal steering of commercial vehicles. What's more, using Wi-Fi channel state information (CSI) for driver motion sensing are gaining popularity. WiCAR [24] employed a multi-adversarial domain adaptation network model to recognize various driver inattentive activities, while Bai et al. [4] proposed a feature combination solution named CARIN to detect a series of common driver gestures and perform well in real-car setting scenarios. Although above methods achieved promising results, they are susceptible to the change of testing environments, the extra infrastructure cost should be also considered.

Early eeg based approaches mainly focus on determining the periods of driver drowsiness (drowsy or non-drowsy) and distraction (distracted and non-distracted) by exploring the correlation between EEG data and driver eye/head states [18,26]. Towards the most recent work in detecting driver behaviors, Bajwa et al. [5] presented a study on distraction stimuli by using EEG signals to record unsafe driving events such as reading, texting, and using phone cameras. Li et al. [17] further investigate EEG patterns by employing both CNN models and gated recurrent units (GRUs) on different types of driver distractions including cellphone operation task, clock task, and 2-back task. Unlike above EEG based approaches, in this paper we aim to identify distracted activities that most commonly occurred in the vehicle. To the best of our knowledge, it is the first work that leverages EEG signals targeting certain in-vehicle driver manual distractions. More importantly, we also propose a channel reduction scheme to select most representative electrodes, which could achieve a comparable accuracy (using only 2 electrodes) with much lower computational cost compare to existing solutions (e.g., 32 or 16 channel electrodes).

5 Conclusion

In this paper, we aim to build a EEG based sensing system that detects most common occurring distracted driving events. AIMSafe can actively records a number of in-vehicle distracting activities and the overall performance is promising. Moreover, we focus on the task of active human motion detection and channel reduction solutions, which is a stepping stone to general multi-channel time

series data analysis. In the future, we tend to solve the above-mentioned limitations and enhance system performance with several possible solutions.

Acknowledgement. This work has been supported by JCYJ20220531091407016, NSFC (No. 61872247), NSFC (No.62272320), Department of Science and Technology of Guangdong Province (No. KTP20210179), Department of Education of Guangdong Province (No. 2022ZDZX4102), Shenzhen Science and Technology Innovation Commission (No. 20220812222043002).

References

1. Al-Sultan, S., Al-Bayatti, A.H., Zedan, H.: Context-aware driver behavior detection system in intelligent transportation systems. IEEE Trans. Veh. Technol. **62**(9), 4264–4275 (2013)
2. Alomari, M.H., Baniyounes, A.M., Awada, E.A.: EEG-based classification of imagined fists movements using machine learning and wavelet transform analysis. Int. J. Adv. Electron. Electr. Eng. **3**(3), 83–87 (2014)
3. Assefa, A.A., Wenhong, T.: Bio-mechanical distracted driver recognition based on stacked autoencoder and convolutional neural network. In: 2019 IEEE 2nd International Conference on Information Communication and Signal Processing, pp. 449–453. IEEE (2019)
4. Bai, Y., Wang, X.: CARIN: wireless CSI-based driver activity recognition under the interference of passengers. Proc. ACM Interact. Mob. Wearable Ubiquit. Technol. **4**(1), 1–28 (2020)
5. Bajwa, G., Fazeen, M., Dantu, R.: Detecting driver distraction using stimuli-response EEG analysis. arXiv preprint arXiv:1904.09100 (2019)
6. Bi, C., Huang, J., Xing, G., Jiang, L., Liu, X., Chen, M.: SafeWatch: a wearable hand motion tracking system for improving driving safety. ACM Trans. Cyber-Phys. Syst. **4**(1), 1–21 (2019)
7. Chen, D., Cho, K.T., Han, S., Jin, Z., Shin, K.G.: Invisible sensing of vehicle steering with smartphones. In: Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services, pp. 1–13 (2015)
8. Chuang, C.H., Huang, C.S., Ko, L.W., Lin, C.T.: An EEG-based perceptual function integration network for application to drowsy driving. Knowl.-Based Syst. **80**, 143–152 (2015)
9. Goel, B., Dey, A.K., Bharti, P., Ahmed, K.B., Chellappan, S.: Detecting distracted driving using a wrist-worn wearable. In: 2018 IEEE International Conference on Pervasive Computing and Communications Workshops, pp. 233–238. IEEE (2018)
10. Gurudath, N., Riley, H.B.: Drowsy driving detection by EEG analysis using wavelet transform and k-means clustering. Procedia Comput. Sci. **34**, 400–409 (2014)
11. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7132–7141 (2018)
12. Jiang, H., Hu, J., Liu, D., Xiong, J., Cai, M.: DriverSonar: fine-grained dangerous driving detection using active sonar. Proc. ACM Interact. Mob. Wearable Ubiquit. Technol. **5**(3), 1–22 (2021)
13. Jiang, L., Chen, X., He, W.: SafeCam: analyzing intersection-related driver behaviors using multi-sensor smartphones. In: 2016 IEEE International Conference on Pervasive Computing and Communications (PerCom), pp. 1–9. IEEE (2016)

14. Jiang, L., Lin, X., Liu, X., Bi, C., Xing, G.: SafeDrive: detecting distracted driving behaviors using wrist-worn devices. Proc. ACM Interact. Mob. Wearable Ubiquit. Technol. **1**(4), 1–22 (2018)
15. Johnson, G., Rajamani, R.: Smartphone localization inside a moving car for prevention of distracted driving. Veh. Syst. Dyn. **58**(2), 290–306 (2020)
16. Lee, K., Lee, K., Lee, H., Shin, J.: A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In: Advances in Neural Information Processing Systems, vol. 31 (2018)
17. Li, G., Yan, W., Li, S., Qu, X., Chu, W., Cao, D.: A temporal-spatial deep learning approach for driver distraction detection based on EEG signals. IEEE Trans. Autom. Sci. Eng. **19**(4), 2665–2677 (2021)
18. Narejo, S., Pasero, E., Kulsoom, F.: EEG based eye state classification using deep belief network and stacked autoencoder. Int. J. Electr. Comput. Eng. (IJECE) **6**(6), 3131–3141 (2016)
19. Palipana, S., Rojas, D., Agrawal, P., Pesch, D.: FallDeFi: ubiquitous fall detection using commodity Wi-Fi devices. Proc. ACM Interact. Mob. Wearable Ubiquit. Technol. **1**(4), 155 (2018)
20. Raja, M., Ghaderi, V., Sigg, S.: WiBot! In-vehicle behaviour and gesture recognition using wireless network edge. In: 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS), pp. 376–387. IEEE (2018)
21. Suwannarat, A., Pan-ngum, S., Israsena, P.: Comparison of EEG measurement of upper limb movement in motor imagery training system. Biomed. Eng. Online **17**(1), 103 (2018)
22. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)
23. Torres, R., Ohashi, O., Carvalho, E., Pessin, G.: A deep learning approach to detect distracted drivers using a mobile phone. In: Lintas, A., Rovetta, S., Verschure, P.F.M.J., Villa, A.E.P. (eds.) ICANN 2017. LNCS, vol. 10614, pp. 72–79. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68612-7_9
24. Wang, F., Liu, J., Gong, W.: WiCAR: WiFi-based in-car activity recognition with multi-adversarial domain adaptation. In: 2019 IEEE/ACM 27th International Symposium on Quality of Service, pp. 1–10 (2019)
25. Wang, L., Gu, T., Chen, H., Tao, X., Lu, J.: Real-time activity recognition in wireless body sensor networks: from simple gestures to complex activities. In: 2010 IEEE 16th International Conference on Embedded and Real-Time Computing Systems and Applications, pp. 43–52. IEEE (2010)
26. Wang, S., Zhang, Y., Wu, C., Darvas, F., Chaovalitwongse, W.A.: Online prediction of driver distraction based on brain activity patterns. IEEE Trans. Intell. Transp. Syst. **16**(1), 136–150 (2014)
27. Wang, Y.K., Jung, T.P., Lin, C.T.: EEG-based attention tracking during distracted driving. IEEE Trans. Neural Syst. Rehabil. Eng. **23**(6), 1085–1094 (2015)
28. Wang, Y., Cao, J., Li, W., Gu, T., Shi, W.: Exploring traffic congestion correlation from multiple data sources. Pervasive Mob. Comput. **41**, 470–483 (2017)
29. Xu, X., et al.: ER: early recognition of inattentive driving leveraging audio devices on smartphones. In: IEEE Conference on Computer Communications, pp. 1–9 (2017)
30. Yang, Y., Du, Z., Jiao, F., Pan, F.: Analysis of EEG characteristics of drivers and driving safety in undersea tunnel. Int. J. Environ. Res. Public Health **18**(18), 9810 (2021)

A Multi-task Learning Framework with Features Based on Behavioral Pattern Conversation

Bo Tang[1,2], Nan Wang[1,2(✉)], Jinbao Li[3,4(✉)], and Zhonghui Shen[1]

[1] College of Computer Science and Technology, Heilongjiang University, Harbin, China
2211889@s.hlju.edu.cn, wangnan@hlju.edu.cn
[2] Key Laboratory of Database and Parallel Computing, Heilongjiang University, Harbin, China
[3] Shandong Artificial Intelligence Institute, Qilu University of Technology, Jinan, China
lijinb@sdas.org
[4] School of Mathematics and Statistics, Qilu University of Technology, Jinan, China

Abstract. Deep neural network based multi-task learning has been widely successful in many real-world large scale applications, such as recommendation systems. A large number of commodity transaction data from e-commerce platforms show that users often go through a series of behavioral transitions such as impressed → click → add shopping cart before finally forming a purchase behavior, and only very few users click and then make a purchase directly. This phenomenon precisely follows the objective reality of power-law distribution. Based on this, this paper proposes a Multi-task learning framework with features based on Behavioral Pattern Conversation (BPCM). A feature tower model based on attribute information is constructed in the framework, which is able to control the fusion and screening process of features adaptively through the designed novel gate complementary mechanism. In addition, we designed several submodules with behavioral pattern conversation (BPC) applied to multi-task learning. The class of modules is not only able to adaptively model the sequentiality and dependencies between behavioral task transitions through information transfer, but also to effectively control the amount of information transferred between different tasks. Adequate experiments show that our BPCM obtains higher performance compared to more current advanced multi-task learning frameworks.

Keywords: Multi-task Learning · Behavioral Pattern Conversation · Recommendation System

Supported by National Natural Science Foundation of China (No. 62172243), Heilongjiang Provincial Natural Science Foundation of China (No. LH2021F047).

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 256–267, 2024.
https://doi.org/10.1007/978-3-031-63992-0_17

1 Introduction

In recent years deep neural network models have been successfully applied to many real-world, large-scale applications, such as recommendation systems [5]. In the real world, there is a rich variety of forms of user feedback on things. Take e-commerce as an example, users' behaviors such as clicking, collecting, adding shopping cart and buying can reflect different interests of users in items from different perspectives. As a result, much of the work in recommendation systems has focused on dealing with the preference of different behaviors of users for items [4]. However, different behavioral patterns of users are closely related to each other [6], and their behaviors are often ordered and interdependent, most likely affecting the strength of the user's acceptance of items at different steps. For example, users tend to browse items and then make a purchase decisions after adding shopping cart. The sequentiality and dependability between these actions is often overlooked.

As shown in the example in Fig. 1, a user has recently browsed apparel items (shoes, T-shirts, pants, etc.) and electronic items (cell phones, game consoles, etc.), and then the user further added apparel (shoes, T-shirts) and electronic items (cell phones) to the shopping cart, and finally the user purchased only two apparel items (shoes, T-shirts). By further obtaining information on the features of these two categories, we found that the user's current interests and purchase intentions are in fact governed by factors such as a certain "price" range and some "brands" of clothing, in addition to individual preferences. Therefore, fully considering the feature information of interactive goods can also effectively capture the potential interest of users. In addition, utilizing feature information between user behavior conversions can also rationalize the sequences and dependencies between user behavior patterns.

It is worth noting that recommender systems do not want users to display only a single goal, such as CTR (Click-Through-Rate); they often want users to display more goals to fulfill multiple user needs [2].

In addition, in the academic field, multi-task learning is a typical approach to solve the end-to-end conversion problem [17]. Since all tasks in the learning process use the same model and need to share the underlying data, it is equally important to handle the underlying data flow in a rational way. In recent years, modeling of inter-task relationships in multi-task learning has been partially achieved, and the most common approach is to use expert models [9,13].

The main contributions of this paper are as follows:

- 1. A multi-task learning framework with features based on behavioral pattern conversation (BPCM) is proposed. The framework effectively exploits the sequential relationships existing between user behavioral patterns and the interdependencies among the tasks processed to solve the multi-task learning problem under multiple behavioral patterns.
- 2. We proposed a feature tower model (Feature Tower) based on behavioral interaction information. In addition to making full use of the interaction sequence information of different behaviors, additional analysis takes into account the feature information of users and interacted items.

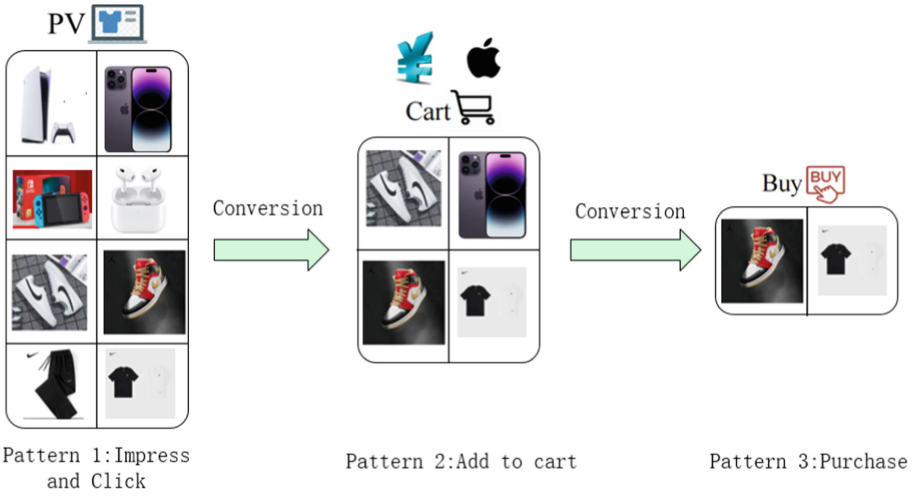


Fig. 1. Illustration of behaviors with sequential and dependent instructions through the user's history.

- 3. A Behavior Pattern Conversion (BPC) module is intelligently designed to adaptively learn the information to be passed between different behavioral task phases and the magnitude of the information. This module not only effectively guarantees the interdependence between different behavioral patterns, but also more strictly constrains the sequential nature in the behavioral pattern conversion task, and obtains more accurate prediction results.
- 4. Comprehensive experiments have verified that the model in this paper has better performance compared to recent more advanced models with the support of real large e-commerce platform datasets.

2 Related Works

In this section we introduce the related thesis of multi-task learning and prior behavior transformation, briefly describing the technical background and the current state of research.

2.1 Recommendation Model Based on Behavior Conversion

Several previous works have proposed some solutions to the end-to-end conversion rate problem and the sequential dependence problem among multiple tasks. For example, Ma et al. proposed the full-space task model (ESMM) [6]. Interlevel click-through rate (CTR) and conversion rate prediction (CVR) use dot product operations to pass the probabilities in the output layer. This method does not effectively utilize deep networks, and thus has average performance in more complex and task-unspecific application scenarios. The mixed sequence

expert model (MoSE) has also been proposed to model sequential user behavior in multi-task learning, but the lack of a tower model to guide the learning process between tasks leads to no information exchange in tasks and thus no improvement in the relationship of multiple behaviors [7]. Another method applied to advertisement analysis Adaptive Information Transfer Multitasking Framework (AITM) [17]. The framework is a more advanced and complete model for solving some of the behavior conversion problems, however, it does not filter and control the effective information and the amount of information between behaviors.

2.2 Multi-task Learning

Multi-task learning has been successful in many fields of machine learning, such as natural language processing, speech recognition, and computer vision. Multi-task models can learn commonalities and differences between different tasks in such a way that similar or identical parts between tasks can be handled equally, but at the same time there is differentiation to learn parts that are more different between tasks. This allows to improve the learning efficiency and model quality of each task [11]. Cauruana proposed a widely used multi-task learning model [3]. The model has a shared underlying model structure that can greatly reduce the risk of model overfitting. However, the variability between tasks may bring about conflicts in the parameter optimization section, and different tasks will inevitably interact with each other during the optimization process. Therefore, a more advanced way of underlying data sharing and task processing allocation is eagerly sought to handle the multi-task learning problem. Mixed expert models are often identified as an effective way to handle multi-task learning, with the basic idea that the underlying data is shared but each task is processed separately when multitasking.

3 Method

In this paper, a BPCM framework is constructed to solve the multi-task learning problem with multiple behavior patterns. The study is mainly based on an e-commerce platform and uses the generally accepted sequential relationship in this model to define a sequence of behavioral task conversions steps [17]. It is worth stating that the behaviors in this sequence of relations are sequential and monotonically ordered, except that all other task recurrence relations are not legal. Since the datasets of different platforms are very different, it is not possible to define a perfect sequentiality for all datasets, so it is sufficient to define behavioral inter-task recurrence relationships that fit the dataset and are reasonable [15]. The feature tower models consists of three components: 1) feature fusion self-attention module; 2) feature-aware aggregation module; and 3) feature tower complementary gating unit. The overall structure of BPCM is shown in Fig. 2.

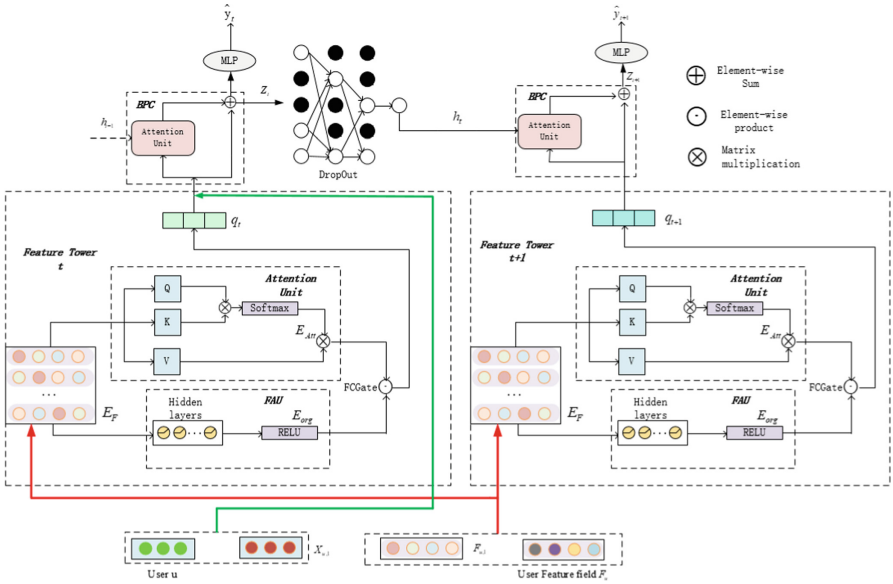


Fig. 2. Illustration of the proposed BPCM model. The middle figure shows the feature tower module corresponding to the current behavior task; the upper figure shows the Behavior Pattern Conversion (BPC) module and the information transfer process between behavior tasks; the lower figure illustrates the input and embedding module.

3.1 Problem Formulation

Given a set of users $U = \{u_1, u_2, \ldots, u_H\}$, where H is the number of users, and X_u is a set of the sequence of historical interactions of the user u, $X_u = \{X_{u,1}, X_{u,2}, ..., X_{u,I}\}$, where $X_{u,I}$ is the I-th item that the user interacted with and I is the number of items. $F_{u,i}$ denotes the set of features of item i that is interacted with user u, $F_{u,i} = \{f_{i,1}, f_{i,2}, ..., f_{i,N}\}$, where $f_{i,N}$ represents the N-th features of item i (e.g., the price of the item, the store it belongs to, etc.), N is the number of item feature information. F_u is the information about the user's features (e.g. user's age, spending power, etc.) denoted as $F_u = \{f_{u,1}, f_{u,1}, ..., f_{u,M}\}$, where M is the number of user feature information. y_t represents the label of the current task (e.g., 1 if clicked, 0 otherwise), and y_{t+1} represents the label of the next task.

3.2 Input Module

In order to avoid too long and redundant input data, we adopt the user-item pair as the model input, and append the feature information of users and items.

$$x_i = \{u_i, X_{u,i}, F_{u,i}, F_u\} \tag{1}$$

We generate the embedding matrix from the user id, item id and the respective set of corresponding features, which can transform the input of the model into a low-dimensional vector [16].

3.3 Feature Tower Module

Feature Fusion Self-attention Unit. The e-commerce data set is rich in Item features, such as price, brand, store name, category, region, etc. Empirically, different features have different effects on model performance [8]. In the BPCM framework, we introduce the self-attentive mechanism [14] to learn the relevance scores between the respective item features in the historical interaction sequence. The final output of the self-attentive unit is the integral feature matrix with adaptive relevance weights. The formula is as follows:

$$\begin{bmatrix} Q \\ K \\ V \end{bmatrix} = E_F \begin{bmatrix} W_Q \\ W_X \\ W_V \end{bmatrix} \tag{2}$$

$E_F \in \mathbb{R}^{I \times d}$ is the initial Embedding matrix of feature information in the input sequence x_i. The weight matrix $W_Q, W_K, W_V \in \mathbb{R}^{d \times d}$ represents the query vector, the key vector and the value vector, respectively. Then we use the softmax function to dot product the query vector (Q) with the key vector (K) to obtain the following formula for the attention matrix on the value vector (V).

$$E_{Att} = Attention(Q, K, V) = \text{SoftMax}\left(\frac{QK^T}{\sqrt{d}}\right)V \tag{3}$$

$E_{Att} \in \mathbb{R}^{I \times d}$ represents the output of the self-attentive mechanism. $\sqrt{d}$ plays a regulating role as a scaling factor so that the value of the inner product is not too large to affect subsequent calculations with smaller inner product values.

Feature-Aware Aggregation Unit (FAU). The self-attentive mechanism introduced in the previous section only uses pairwise feature interactions to represent the relative importance between features, but fails to further refine the learning of features using more comprehensive feature information. Therefore, we designed a feature-aware aggregation module component to aggregate the feature encoding relationships between users and items.

$$E_{\text{org}} = RELU\left(W_2\left(RELU\left(W_1 E_F + b_1\right)\right) + b_2\right) \tag{4}$$

where W_1 and W_2, b_1 and b_2 represent the trainable parameters of the first layer of the deep network and the trainable parameters of the second layer of the deep network, respectively. E_F stands for embedding the initial feature information of the input sequence as the input to the MLP. $E_{org} \in \mathbb{R}^{I \times d}$ then represents the native feature aggregation information of the input instance.

Feature Complementary Gate Unit (FCGate). Based on the GRU gate mechanism [3], we designed a novel complementary gating mechanism to control

the fusion and filtering process of all features. FCGate consists of three components: 1) feature fusion self-attention matrix E_{Att}; 2) native feature aggregation matrix E_{org}; 3) weight matrix W_b. $W_b \in \mathbb{R}^{I \times d}$ represents the learned gate signal.

$$W_b = \delta\left(W_C E_{org} + W_D E_{Att}\right) \tag{5}$$

δ represents the function Sigmoid, $W_C \in \mathbb{R}^{d \times d}$ and $W_D \in \mathbb{R}^{d \times d}$ represent trainable parameters in the gate control unit. The final output of the complementary gating unit is as follows:

$$q_t = E_{Att} \odot W_b + E_{\text{org}} \odot (1 - W_b) \tag{6}$$

where $q_t \in \mathbb{R}^{I \times d}$ represents the integral output of the feature tower module.

3.4 Behavior Pattern Conversion Module (BPC)

The purpose of the BPC module is to adaptive learn the transfer of information between behavioral tasks. Given T behavioral tasks, the output of the feature tower module corresponding to each task $t(1 \leq t \leq T)$ is q_t. For two adjacent tasks t and task $t+1$, since not all the information in the previous task t is worth passing to the next step task $t+1$. If all of them are imported, it will cause unavoidable information redundancy and overfitting, so the amount of information to be passed down needs to be traded off to solve this problem. Combining the previous methods of Layer Normalization [1] and Dropout [12] in Transformer [14]. We propose the following approach as a mode of passing between behavioral tasks.

$$Z_{t+1} = BPC\left(q_{t+1}, h_t\right) \tag{7}$$

$Z_t \in \mathbb{R}^I$ is the output corresponding to the task t in the BPC module.

$$h_t = msg\left(Z_t\right) \tag{8}$$

$msg\left(\right)$ is a function of how much information should be learned between task t and task $t+1$. The method is not unique, and in the paper the Dropout data augmentation method is used.

The BPC is designed as a module that adaptively assigns weights h_{t-1} to the passed information and the original information q_t of the current task. Previous work has also demonstrated that attention mechanisms are more effective in multi-tasking [18], and thus this effect can be achieved simply and effectively by reusing the self-attention mechanism in the feature tower model in this paper.

$$BPC() = \text{ Attention } \left(q_{t+1}, h_t\right) + q_{t+1} \tag{9}$$

In addition, for the first task, the output of the feature tower model is taken directly as the output of the first task in the initialization of the framework, that is $Z_1 = q_1$ when $t = 1$.

3.5 Prediction Module

In the prediction module we take a simple MLP approach with a final sigmoid function activation. It is additionally worth stating that the module output Z_t is used for both the current behavioral task prediction and the transmission to the next phase of the behavioral task. The predicted probability for the t-th task is

$$\hat{y}_t = \text{Sigmoid}\left(MLP\left(Z_t\right)\right) \tag{10}$$

3.6 Model Training and an Additional Compensation Loss

In the classification task, we choose the cross-entropy loss function to measure the model performance and training effect.

$$L_{Cl}(\theta) = -\frac{1}{N}\sum_{t=1}^{T}\sum_{(x_t,y_t\varepsilon D}^{N}\left(\left(y_t \log \hat{y}_t + (1-y_t)\log\left(1-\hat{y}_t\right)\right)\right) \tag{11}$$

N is the number of all samples in the sample space D that contains the inputs and labels. y_t is the true label under the t-th behavioral task and $\hat{y}_t$ is the predicted label under the t-th behavioral task.

θ is the setting hyper parameter in the BPCM framework. In addition, we introduce an additional calibrator $\mathcal{L}_{cr}(\theta)$ to minimize the task objective.

$$\mathcal{L}_{cr}(\theta) = -\frac{1}{N}\sum_{t=2}^{T}\sum_{(x_1\in D)}^{N}\max\left(\hat{y}_{t+1}-\hat{y}_t, 0\right) \tag{12}$$

Finally, we get an additional compensation Loss $\mathcal{L}(\theta)$, which combines the two components into a loss function.

$$\mathcal{L}(\theta) = \mathcal{L}_{cl}(\theta) + \alpha\mathcal{L}_{cr}(\theta) \tag{13}$$

α is controllable calibrator component weights.

4 Experiment

4.1 Datasets

– Ali-CCP Dataset is a public dataset called Ali-CCP (Alibaba Click and Conversion Prediction). This dataset is collected from the logs of mobile Taobao. Users can click on the products they are interested in from the impression results, or make further purchases. Thus the user's behavior can be summarized in a sequential pattern 'impression to click to purchase'. The dataset contains three main parts, the sample id part, the label part, and the feature part. In addition, we used all single-valued features in the feature domain and did not do any preprocessing on the features.

– TaoBao Dataset is a dataset of Taobao display ad click rate prediction. The dataset contains ad display/click log information of 1.14 million users randomly selected from the Taobao website over an 8-day period. It contains user id, ad group id, timestamp and other ad domain features. We use the data of the first seven days as training samples and the data of the eighth day as test samples.

The specific statistical information of the two datasets is shown in Table 1.

4.2 Evaluation Metrics

Following previous work, we used the widely adopted area under receiver operating characteristic curve (AUC) for evaluation in the classification task. Based on the work [14], we additionally introduced a metric called RelaImpr to more intuitively measure the degree of improvement of our model based on the baseline model. Since the AUC value of the stochastic strategy is about 0.5, the RelaImpr in this task is defined as

$$\text{RelaImpr} = \frac{AUC(\text{measured}) - 0.5}{AUC(\text{base}) - 0.5} - 1 \tag{14}$$

Table 1. Dataset statistics. "Positive" represents the percentage of positive samples in the train set.

DataSet	#Train	#Validation	#Test	Feature Num	%Positive
Ali-CCP	38M	4.2M	43M	18	3.89/0.02
TaoBao	16M	1.34M	13.4M	6	1.13/0.05

4.3 Baselines

We compare the proposed model with the following advanced and mainstream models. To demonstrate the effectiveness of our proposed BPCM, we compare the proposed model with the following advanced and mainstream models.

– MLP [10]: as the base baseline model, we use the basic structure of the AITM framework as a single task model. It is a multi-layer perceptron.
– OMoE [9]: the OMoE with an expert model bottom data sharing layer, which integrates the expert model by sharing the same gating across all tasks.
– MMoE [9]: the MMoE is an improvement of the OMoE and integrates the expert model through multiple gating on the basis of the OMoE.
– PLE [13]: the Progressive Layered Extraction (PLE) explicitly separates the task from the underlying data sharing layer using a progressive layered extraction approach based on an expert model.

– AITM [17]: the Adaptive Information Transfer Multi-task (AITM) has an advanced explicit expert model as the underlying data sharing, and also separates the tower model to handle the corresponding tasks separately.

We used MLP as the base comparison baseline model, and the results of the comparison experiment are shown in Table 2.

4.4 Ablation Study

In this section we conducted experiments on the Ali-CCP and TaoBao datasets, respectively. In this way, we analyze the significant impact of each component or design pattern in the BPCM model proposed in this paper on the multi-task prediction performance. As shown in Table 3.

– *w/o* Att denotes the removal of the feature fusion self-attentiveness mechanism, which makes the features between users and item no longer have relevance distinction.
– *w/o* Org denotes the removal of the feature-aware aggregation module. This makes the representation of the original characteristics of the multi-domain features lost in the initial feature fusion of the user with the item disappears.
– *w/o* Gate denotes the removal of the complementary gating unit part of the work. Instead, the approach is to directly sum the two parts of the representation that self-attentive fusion and initial feature fusion.
– *w/o* BPC indicates the removal of the behavior pattern conversion module. The experimental results show that the role of this module is essential. The purpose of this module is to coordinate between tasks based on different labels, while determining how much information to pass between multiple tasks.

Table 2. Performance comparisons on two real-world datasets with five baselines

Model	Ali-CCP				TaoBao			
	click AUC	purchase AUC	RelaImpr		impression AUC	click AUC	RelaImpr	
MLP	0.6035	0.5841	–	–	0.5771	0.5770	–	–
OMoE	0.6031	0.6405	−0.0386	+0.6706	0.5918	0.5887	+0.1906	+0.1519
MMoE	0.6042	0.6420	+0.0067	+0.6884	0.5922	0.5891	+0.1958	+0.1571
PLE	0.6039	0.6417	+0.0038	+0.6848	0.5852	0.5858	+0.1050	+0.1142
AITM	0.6051	0.6506	+0.0154	+0.7907	0.6191	0.6201	+0.5447	+0.5597
BPCM	**0.6192**	**0.6556**	**+0.1516**	**+0.8501**	**0.6245**	**0.6343**	**+0.6147**	**+0.7441**

4.5 Experimental Hyper-parameter Setting

The set of experiments is performed on the Ali-CCP dataset with the embedding size adjusted to $\{5, 16, 32, 64, 120\}$ and the loss function weights adjusted to $\{0.2, 0.4, 0.6, 0.8, 1.0\}$, respectively. The loss weights have a greater influence on

Table 3. Ablation study of BPCM.

Model	Ali-CCP		TaoBao	
	click AUC	purchase AUC	impression AUC	click AUC
w/o Att	0.6183	0.6240	0.6191	0.6183
w/o Org	0.6178	0.6286	0.6076	0.6096
w/o Gate	0.6185	0.6447	0.6187	0.6186
w/o BPC	0.6118	0.5324	0.5645	0.5514
BPCM	**0.6192**	**0.6556**	**0.6245**	**0.6342**

the experimental results, i.e., the influence of the loss function corrector is greater in the prediction process. It is inevitable that the prediction probability of the successor task is higher than the prediction probability of the precursor task due to the sparsity of the data during the model training. After graphical analysis, the model has the best comprehensive performance and the smoothest training process at embedding size $d = 5$ while $\alpha = 0.6$. At this point, the embedding dimension and loss function hyperparameters are optimally balanced.

5 Conclusion

In this work, we explore the problem of multi-task recommendation based on transitions between user behaviors. A new and improved end-to-end expert model framework BPCM is proposed, which realizes the goal by dealing with the underlying data sharing between different tasks and the information transfer between tasks. In addition, we introduce an additional compensation loss function during the training process to achieve an overall optimization of the model. Extensive experiments on two real e-commerce datasets show that our model significantly outperforms the other baseline models.

References

1. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization (2016)
2. Bansal, T., Belanger, D., Mccallum, A.: Ask the GRU: multi-task learning for deep text recommendations (2016)
3. Caruana, R.A.: Multitask learning: a knowledge-based source of inductive bias. Mach. Learn. Proc. **10**(1), 41–48 (1993)
4. Chen, C., Zhang, M., Liu, Y., Ma, S.: Social attentional memory network: modeling aspect- and friend-level differences in recommendation. In: The Twelfth ACM International Conference (2019)
5. Covington, P., Adams, J., Sargin, E.: Deep neural networks for YouTube recommendations. In: ACM Conference on Recommender Systems, pp. 191–198 (2016)
6. Guo, L., Hua, L., Jia, R., Zhao, B., Cui, B.: Buying or browsing?: Predicting real-time purchasing intent using attention-based deep network with multiple behavior. In: The 25th ACM SIGKDD International Conference (2019)

7. Han, X., Hu, J., Ghosh, J.: MECATS: mixture-of-experts for quantile forecasts of aggregated time series. arXiv e-prints (2021)
8. Li, M., Lu, Z., Wu, Y., Li, Y.H.: BACPI: a bi-directional attention neural network for compound-protein interaction and binding affinity prediction. Bioinformatics **38**(7), 1995–2002 (2022)
9. Ma, J., Zhe, Z., Yi, X., Chen, J., Hong, L., Chi, E.H.: Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. ACM (2018)
10. Gardner, M.W., Dorling, S.R.: Artificial neural networks (the multilayer perceptron)-a review of applications in the atmospheric sciences. Atmos. Environ. **32**(14–15), 2627–2636 (1998)
11. Ruder, S.: An overview of multi-task learning in deep neural networks (2017)
12. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
13. Tang, H., Liu, J., Zhao, M., Gong, X.: Progressive layered extraction (PLE): a novel multi-task learning (MTL) model for personalized recommendations. In: RecSys 2020: Fourteenth ACM Conference on Recommender Systems (2020)
14. Vaswani, A., et al.: Attention is all you need. arXiv (2017)
15. Wan, M., Mcauley, J.: Item recommendation on monotonic behavior chains, pp. 86–94 (2018)
16. Wang, R., Shivanna, R., Cheng, D., Jain, S., Chi, E.: DCN V2: improved deep & cross network and practical lessons for web-scale learning to rank systems. In: WWW 2021: The Web Conference 2021 (2021)
17. Xi, D., Chen, Z., Yan, P., Zhang, Y., Chen, Y.: Modeling the sequential dependence among audience multi-step conversions with multi-task learning in targeted display advertising (2021)
18. Zhu, Y., et al.: Modeling users' behavior sequences with hierarchical explainable network for cross-domain fraud detection (2022)

Efficient Parcel Damage Detection via Faster R-CNN: A Deep Learning Approach for Logistical Parcels' Automated Inspection

Zhi Chen[1,2], Cuifeng Du[5], Quanlong Guan[1,2,4], Yuyu Zhou[2,7], Vichen Hoo[1,7], Xiujie Huang[1,4], Zhefu Li[3(✉)], Shuanghuan Lv[3], Xiaofeng Wu[6(✉)], and Xiaotian Zhuang[8]

[1] College of Information Science and Technology, Jinan University, Guangzhou, China
[2] Guangdong Institute of Smart Education, Jinan University, Guangzhou, China
[3] Network and Education Technology Center, Jinan University, Guangzhou, China
[4] Guangdong Key Laboratory of Data Security and Privacy Preserving, Guangzhou, China
[5] Cetc Potevio Science&Technology Co., Ltd., Guangzhou, China
[6] Guangzhou Polytechnic of Sports, Guangzhou, China
[7] Guangdong-Macao Advanced Intelligent Computing Joint Laboratory, Guangzhou, China
[8] Beijing JD Zhenshi Information Technology Co., Ltd., Beijing, China

Abstract. Parcel damage detection poses a significant challenge in warehouse automation and conveyor belt transportation, with a particular focus on leveraging the potential of Internet of Things (IoT) technologies. With this purpose in mind, we introduce RPA R-CNN, an improved deep learning-based object detection method that seamlessly integrates IoT capabilities. Our proposed approach enhances the feature extraction network of the Faster R-CNN model and utilizes advanced post-processing techniques to further refine the extracted features. By employing a meticulously curated dataset specifically designed for parcel detection, we train our model and conduct rigorous experimental verification. Encouragingly, the outcomes convincingly demonstrate that the RPA R-CNN method surpasses the performance of the original model in various metrics in domestic or cross-border logistics parcel detection. These promising results not only offer a practical solution to the challenging problem of identifying parcel damage but also underscore the immense potential of integrating IoT principles to optimize automation and enhance operational efficiency within the domestic or cross-border logistics industry.

Keywords: Faster R-CNN · Parcel Damage Detection · IoT

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 268–279, 2024.
https://doi.org/10.1007/978-3-031-63992-0_18

1 Introduction

With the rapid growth of e-commerce [4] and the domestic or cross-border logistics industry, there are significant concerns regarding the detection and localization of package damage in domestic or cross-border logistics environments, including scenarios such as automated warehouse transport. The development of efficient methods for accurately identifying and locating damaged parcels is crucial for ensuring the safety of goods and achieving high delivery accuracy. Traditional approaches relying on manual inspection and human judgment suffer from limitations such as low efficiency, lengthy processing times, and potential errors. Meanwhile, there is a growing number of applications applying deep learning techniques to object detection [3], including video object detection [8], UAV location [5], crack detection [27], and so on. Thus, our objective is to propose an efficient object detection method for parcel damage detection through the integration of deep learning and IoT technologies.

While Faster R-CNN [19] has demonstrated remarkable performance in general object detection, it may not be the most suitable choice for parcel damage detection due to feature extraction and localization precision limitations. We propose improvements to the Faster R-CNN model to address these limitations and enhance the efficiency and accuracy of parcel damage detection in logistics scenarios. In our proposed RPA R-CNN model, we incorporate ResNet-101 [11] and PANet [13] to enhance the extraction of image features from parcel images. ResNet-101 excels at capturing detailed information within parcel images, while PANet effectively handles features at different scales. We considered alternative models, including YOLOX-Tiny [6], but ultimately selected ResNet-101 and PANet due to their superior accuracy and stability in parcel damage detection tasks. Although lightweight models like YOLOX-Tiny may offer improved speed in specific scenarios, their performance in complex scenes and with small-sized targets falls short of that of ResNet-101 and PANet. To evaluate the efficacy of our model, we conducted experiments using a parcel detection dataset provided by Roboflow [20] and our self-compiled training dataset. Subsequently, we performed verification tests. The results affirm the superiority of our proposed RPA R-CNN model over the original Faster R-CNN, demonstrating higher precision and accuracy in detecting parcel damage within logistics operations.

Our research endeavor aims to provide the e-commerce and logistics industries with an efficient and accurate method for parcel damage detection, thereby enhancing the safety and delivery precision of goods. Furthermore, our proposed methodology possesses a versatile nature that allows an extension to other detection problems, thereby offering practical value in real-world logistics applications.

2 Related Work

The R-CNN algorithm, introduced by Ross Girshick et al. [7] in 2014., was a significant milestone for object detection. However, processing individual regions separately resulted in slow performance and high computational complexity, making it less practical for real-world applications. To address this challenge, Joseph et al. [16] introduced the YOLO algorithm, which revolutionized

object detection by adopting a fully convolutional network structure. Notably, YOLOv3 [18] expanded upon YOLOv2 [17] by incorporating multi-scale prediction, residual connections, and a feature pyramid network, establishing it as a crucial object detection algorithm in the field. In 2020, Ultralytics introduced the YOLOv5-S [23] algorithm, which achieved a commendable balance between lightweight network design and object detection performance. To further enhance the accuracy of object detection, Wang et al. [24] proposed YOLOv7. This version employed trainable bag-of-freebies techniques. Although YOLOv7 attained higher accuracy compared to YOLOv5, it exhibited slower runtime performance. Compared to the traditional YOLO series, Zheng et al. [6] introduced YOLOX-Tiny in 2021, a lightweight version of the YOLOX series. This variant maintains high detection accuracy while reducing computational complexity. Similarly, with a focus on one-stage detection, Yang et al. [29] proposed PDNet, which established a more flexible detection paradigm by adopting pixel-wise prediction. Tian et al. [22] proposed a fully convolutional one-stage approach for object detection, eliminating the need for predefined anchor boxes and improving detection accuracy and efficiency. Haq et al. [9] presents a monocular 3D object detection method for autonomous driving, addressing the challenge of accurate object localization in 3D space and achieving improved detection performance. Wang et al. [25] proposed CrabNet, a fully task-specific feature learning method for one-stage object detection, resulting in improved performance and higher accuracy.

To accurately locate small-sized targets and improve detection performance, Ren et al. [19] proposed Faster R-CNN. Unlike the above-mentioned model, Faster R-CNN utilizes a two-stage detection strategy for accurate localization and detection of small objects while maintaining high speed. It also offers flexibility for future advancements in this field. Several variant algorithms derived from Faster R-CNN, such as Mask R-CNN [10], Cascade R-CNN [2], and so on [1,15,30,32], have also been developed. Tan et al. [21] proposed the EfficientDet object detector, which is built upon the EfficientNet backbone network and introduces the BiFPN and composite scaling methods. These innovations enable EfficientDet to achieve better efficiency in resource-constrained scenarios. ResNet has been widely utilized in many object detection studies with impressive results. Li et al. [12] introduced GFL, which employs ResNet-101 as the backbone network and optimizes continuous labels using the GFL, leading to efficient dense object detection. Another ResNet-based method is SOLO, introduced by Wang et al. [26]. SOLO is a position-based object segmentation algorithm that accurately segments objects by predicting their instance center points and generating dense object segmentation masks. In 2022, Zhang et al. [31] proposed ResNeSt, which utilizes the ResNet structure in the backbone network and enhances feature modeling capabilities through a segmentation attention module. Notably, the Region Proposal Network (RPN) and Region of Interest (RoI) also play critical roles in these algorithms. Xie et al. [28] developed Oriented R-CNN, which focuses on RoI and introduces a direction-sensitive RoI alignment module, resulting in improved directional object detection. Similarly, Ou et al. [14]

proposed AD-RCNN, which enhances the performance and efficiency of small object detection by incorporating a dynamic RPN, visual attention mechanism, and adaptive dynamic training module. The aforementioned research endeavors have made significant contributions to the advancement of technology in the domain of object detection. These studies furnish valuable references for future investigations concerning object detection and its integration with the IoT.

3 RPA R-CNN Framework

Within this section, we predominantly introduce our refined model for Faster R-CNN, referred to as RPA R-CNN. Our RPA R-CNN approach primarily focuses on modifying its feature extraction network while enhancing performance. This adaptation facilitates more precise detection of logistics parcels and enhances the overall detection rate. The structure of our RPA R-CNN model is illustrated in Fig. 1.

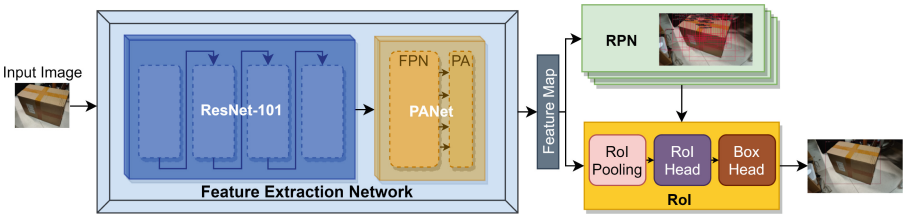


Fig. 1. The proposed RPA R-CNN framework.

3.1 Feature Extraction Network

Our feature extraction process for parcel images primarily relies on ResNet-101 as the backbone network. It consists of 4 stages, each composed of several residual blocks. These blocks extract feature vectors with different resolutions and semantics, allowing the network to capture information at various scales and enhance recognition and classification performance. However, to further optimize the feature propagation and leverage the interaction between features, we introduce the PANet module. The PANet module processes and refines the extracted features, promoting better contextual understanding and preserving relevant details.

In our model, we associate the feature maps obtained from the four stages of the ResNet-101 network with four scales of [256, 512, 1024, 2048]. These feature maps are then fed into the PANet model. This process can be represented as:

$$\begin{cases} F_1 = R_1(I) \\ F_2 = R_2(F_1) \\ F_3 = R_3(F_2) \\ F_4 = R_4(F_3) \\ F_{\text{out}} = P(F_1, F_2, F_3, F_4) \end{cases} \tag{1}$$

where I represents the input image, $R1, R2, R3, R4$ represent the upsampling modules corresponding to the four stages of ResNet-101. The features obtained from the four stages of ResNet-101, denoted as $F1, F2, F3, F4$, have scales of $S1 = 256, S2 = 512, S3 = 1024, S4 = 2048$. These features are inputted into the PANet model, resulting in a high-level abstract representation F_{out} of the input image.

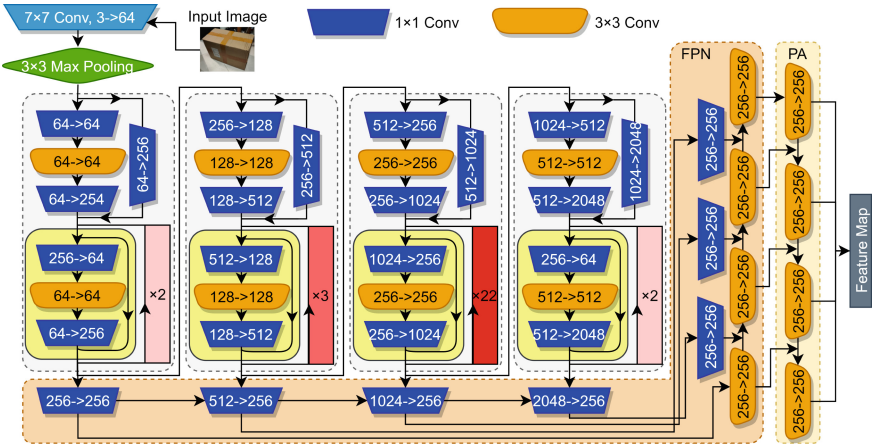


Fig. 2. Feature extraction network framework.

When features are input into PANet, we initially utilize FPN (Feature Pyramid Network) to integrate feature maps from various levels. The feature maps obtained from the four stages of the backbone network are individually passed through a top-down downsampling path, where each feature map undergoes a convolution operation and is downsampled:

$$D_i = Ds(D_{i-1}) + W_i F_i \tag{2}$$

where D_i represents the downsampled feature map at level i, $Ds(\cdot)$ denotes the downsampling operation, and W_i represents the corresponding weight matrix. Although this process may reduce some details in the image or feature maps, it still retains important structures and features. Next, the feature maps go through the bottom-up path for upsampling, and lateral connections are introduced to obtain feature maps with different resolutions and semantic levels:

$$U_i = Us(U_{i+1}) + W_i' F_i \tag{3}$$

where U_i denotes the upsampled feature map at level i, $Us(\cdot)$ represents the upsampling operation, and W_i' signifies the corresponding weight matrix. To further optimize the performance, lateral propagation is used:

$$C_i = U_i + D_i \tag{4}$$

where C_i represents the fused feature map at the i-th level, which effectively captures object information across different scales. Subsequently, the processed features are passed into PA (Path Augmentation) to enhance the feature pyramid's performance by incorporating lateral connections and bottom-up PA. This process enables the extraction of high-level abstract representations from the input image. Figure 2 illustrates our feature extraction network architecture.

3.2 RPN and RoI Network

After the feature extraction network, high-level abstract representations of the input image are obtained. The RPN is essential for generating proposals. In our approach, 1000 proposals per image are generated. These proposals are then categorized as damaged parcels, undamaged parcels, or regions excluded from training. We employ the Random Sample method to randomly select candidate boxes for damaged and undamaged parcels. These samples constitute the training set used for model training. This strategy effectively addresses the issue of imbalanced positive and negative samples during training, thereby enhancing the efficiency and detection performance of the model.

We process the proposals generated by the RPN by feeding them, along with the feature map, into the RoI layer. Here, we employ RoI pooling to pool the proposals into equally-sized feature map blocks and generate feature vectors using max pooling. Next, we feed the feature vectors, bbox (bounding box) coordinates, and labels into the ROI Head for further processing. In this layer, our primary focus is on regressing the predicted bbox coordinates using the multi-task loss function:

$$\begin{cases} L_{cls}(p_i, p_i^*) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) \\ L_{reg}(t_i, t_i^*) = \frac{1}{N_{reg}} p_i^* L_{reg}(t_i, t_i^*) \end{cases} \tag{5}$$

where L_{cls} and L_{reg} refer to the classification loss and regression loss, respectively. N_{cls} and N_{reg} ensure a balanced influence of both losses in the overall loss function. λ is a balance coefficient that adjusts the weight between the two losses. p_i and t_i are the predicted values for proposal classification and bbox regression, while p_i^* and t_i^* are the ground truth labels. Finally, the output features from the ROI Head undergo further refinement through the Box Head, enhancing the accuracy of object detection and providing us with classification results and predicted bboxes for the input image.

4 Experiment

4.1 Dataset

Our dataset comprises 2400 images, meticulously curated to encompass a wide range of packaging scenarios commonly encountered in real-world logistics operations. These images were initially collected through web scraping using a custom web scraper and were subsequently augmented by incorporating images

from Roboflow. The dataset is thoughtfully labeled with four distinct categories: Box, Box_broken, Open_package, and Package, ensuring comprehensive coverage of logistics-related packaging conditions. The training, validation, and test sets consist of 1600, 400, and 400 images, respectively.

Table 1. The parameters used during the training process.

Batch size	Epoch	Learning rate	Optimizer	Loss function
4	120	0.0003	Adam	Multi-Task Loss Function

4.2 Experimental Setup

We used MMDetection, a comprehensive framework for object detection, which supports various algorithms and components. It simplifies experimentation for researchers by providing a configuration-based approach for efficient model construction, training, and customization. The training parameters used are presented in Table 1.

Table 2. Model performance comparison of different models.

Model	mAP	mAP_50	mAP_75	Epoch
YOLOv3	0.45	0.791	0.493	120
YOLOX-Tiny	0.498	0.841	0.576	120
YOLOv5-S	0.501	0.853	0.587	120
Faster R-CNN (ResNet101)	0.632	0.867	0.758	120
Faster R-CNN (ResNeSt)	0.696	0.879	0.785	120
RPA R-CNN	**0.74**	**0.906**	**0.827**	120

4.3 Model Performance Comparison

To evaluate the effectiveness of our model, we compared the mAP values of several models, including YOLOv3, YOLOX-Tiny, YOLOv5-S, Faster-RCNN with ResNet101, Faster-RCNN with ResNeSt, and RPA R-CNN, at different confidence threshold levels. As shown in Table 2, it is evident that the RPA R-CNN model achieved the highest mean Average Precision (mAP) score of 0.74 during the same epoch. Furthermore, we examined precision values at mAP thresholds of 0.5 and 0.75. Under the more lenient matching criterion (mAP threshold of 0.5), the RPA R-CNN model reached an mAP_50 value of 0.906, confirming its high level of accuracy. Additionally, under the stricter matching criteria (mAP

threshold of 0.75), the model consistently delivered commendable performance, maintaining an mAP_75 value of 0.827. These results demonstrate the consistent superior accuracy of our RPA R-CNN model across different confidence thresholds. These results show that our RPA R-CNN model consistently achieves the highest accuracy among the six models at different confidence thresholds. It is important to note that YOLO series models, such as YOLOX-Tiny, offer faster inference speeds and can be more suitable for real-time applications, but they may compromise accuracy to some extent. In contrast, models from the RCNN series, including our proposed RPA R-CNN, exhibit higher accuracy but may require more computational resources. Our choice of RPA R-CNN is driven by the specific needs of logistics parcel inspection, where accuracy is paramount, and real-time performance can be balanced with available computing resources. Therefore, we emphasize that the efficiency of our model aligns with the precision requirements of the task. RPA R-CNN's superior accuracy, as demonstrated in the comparison results, makes it a suitable choice for damaged logistics parcel inspection tasks where high accuracy and stability are of utmost importance.

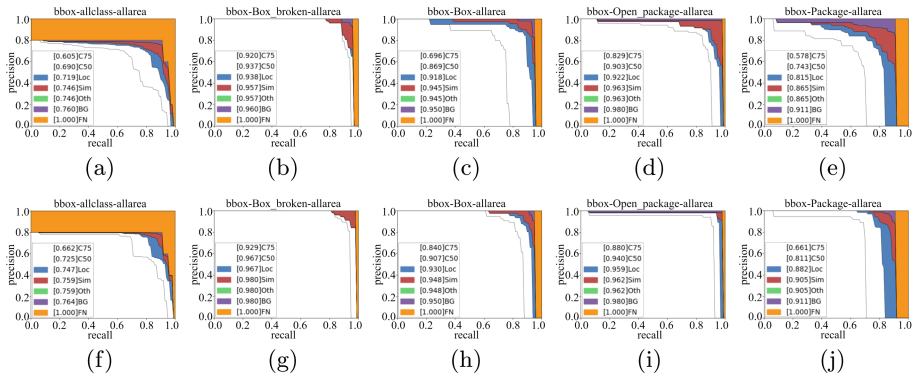


Fig. 3. PR curve of Faster R-CNN (FPN) and RPA R-CNN for parcel damage detection.

4.4 Evaluation of Model Enhancement

To validate the enhancement of our RPA R-CNN model, we compared the PR curves of the Faster R-CNN (FPN) and RPA R-CNN. As shown in Fig. 3. Subfigures (a), (b), (c), (d), and (e) depict the detection of parcels using Faster R-CNN(FPN) for various categories: allclass, box_broken, box, open_package, and package respectively. Similarly, subfigures (f), (g), (h), (i), and (j) depict the detection outcomes using RPA R-CNN for the corresponding categories.

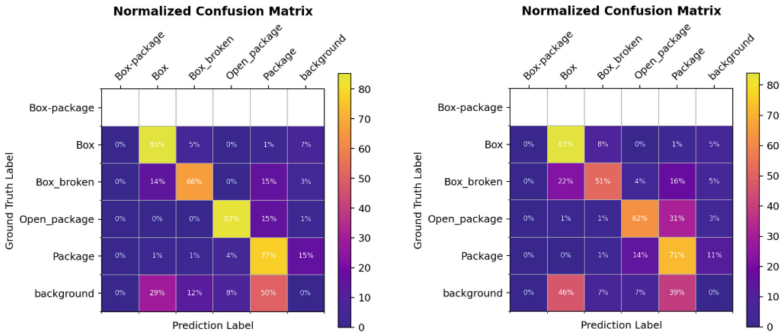
From the PR curves of the two models, RPA R-CNN demonstrates significant advancements in various performance metrics across different parcel categories. Comparing Fig. 3(a)–(e) with (f)–(i) concurrently, we can observe the

following: the "allclass" category exhibits a notable improvement with a 5.7% increase in the C75 metric. Similarly, for the "box_broken" category, RPA R-CNN achieves the maximum enhancement with a 3% boost in the C50 metric. Remarkably, the "box" category showcases the highest improvement among all categories, displaying a remarkable 14.4% increase in the C75 metric. Evaluating the "open_package" category, both the C75 and C50 metrics display relatively similar percentage improvements of 5.1% and 4% respectively. On the other hand, in the "package" category, the C75 metric demonstrates the most substantial improvement, with an impressive 8.3% increase. Additionally, despite being relatively modest, the improvements in the Loc, Oth, BG, and FN metrics, which take into account various confounding factors, indicate that these metrics have either remained stable or exhibited slight enhancements. To summarize, our findings indicate that RPA R-CNN outperforms the original model across all evaluated metrics. Furthermore, the enhanced model exhibits a PR curve that closely approximates the standard curve, indicating excellent training performance. In addition, the improved model outperforms the original model in terms of PR curve performance.

Table 3. Parcel damage detection precision comparison.

Ground Truth Label→Prediction Label	Precision	
	RPA R-CNN	Faster R-CNN(FPN)
Box→Box	**85%**	83%
Box Broken→Box Broken	**66%**	51%
Open Package→Open Package	**83%**	62%
Package→Package	**77%**	71%
Background→Box	**29%**	46%

To intuitively observe the improvement of our RPA R-CNN model, we plotted the key data as a confusion matrix, as shown in Fig. 4. By analyzing the key data presented in the confusion matrix (Table 3), we observed that our RPA-RCNN surpasses Faster-RCNN (FPN) in detecting Box, Box Broken, Open Package, and Package tasks. Despite both models showing a tendency to misclassify background labels, our RPA-RCNN exhibits significant improvements, especially in Box Broken and Open Package tasks, with a boost of 15% and 21% respectively. These results show that RPA R-CNN achieves higher precision across various parcel inspection tasks. Notably, the significant enhancement in detecting Box Broken and Open Package tasks specifically suggests that our model is particularly suitable for inspecting damaged packages. However, it's worth noting that the RPA R-CNN model's size increased to approximately 747 MB compared to around 470 MB for the Faster R-CNN (FPN) model. This increase in model size was a trade-off for improved detection accuracy.



(a) RPA R-CNN (b) Faster R-CNN(FPN)

Fig. 4. Confusion matrix of parcel damage detection on different models.

5 Conclusion

In this study, we propose RPA R-CNN, a deep learning-based object detection method for detecting parcel damage in IoT environments such as warehouse automation and conveyor belt transportation. Our approach improves upon Faster R-CNN by modifying the feature extraction network to ResNet-101 with PANet for accurate parcel image feature extraction, then leveraging RPN and ROI techniques for processing the extracted features. To evaluate our model's effectiveness, we employ parcel detection datasets from Roboflow and internet-crawled images for training. We conduct comparative experiments with commonly used object detection models, and our results indicate that RPA R-CNN excels in mean average precision. Through model enhancement evaluation, we demonstrate that RPA R-CNN achieves superior detection rates for domestic or cross-border logistics parcels compared to the original model.

We observe the relevance of parcel damage localization detection in IoT scenarios like warehouse automation and conveyor belt systems. In practical applications, our model can provide real-time monitoring, data transmission, and analysis for detecting parcel damage. This can significantly improve efficiency and service quality in the logistics industry. Future research directions include expanding our findings to other detection problems and developing practical methods and techniques for the IoT field. By leveraging the advantages of IoT, we can further enhance real-time monitoring and analysis capabilities, contributing to the improvement of logistics operations.

Acknowledgements. This work was supported by the Science and Technology Planning Project of Guangdong (2021B0101420003, 2020ZDZX3013, 2023ZZ03), NSFC (62077028, 62377028), Guangdong Key Laboratory of Data Security and Privacy Preserving (2023B1212060036), the Science and Technology Planning Project of Guangzhou (202206030007), the Opening Project of Key Laboratory of Safety of Intelligent Robots for State Market Regulation (GQI-KFKT202205), Guangdong-Macao Advanced Intelligent Computing Joint Laboratory (2020B1212030003) and the High

Performance Public Computing Service Platform of Jinan University. Zhefu Li and Xiaofeng Wu are the corresponding authors.

References

1. Ali, R., Kang, D., Suh, G., Cha, Y.J.: Real-time multiple damage mapping using autonomous UAV and deep faster region-based neural networks for GPS-denied structures. Autom. Constr. **130**, 103831 (2021)
2. Cai, Z., Vasconcelos, N.: Cascade R-CNN: delving into high quality object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6154–6162 (2018)
3. Chen, Z., et al.: Few-shot learning for quality detection of logistical parcels. In: 2023 11th International Conference on Information Systems and Computing Technology (ISCTech). IEEE (2023)
4. Cheng, X., Li, N., Rysbayeva, G., Yang, Q., Zhang, J.: Influence-aware successive point-of-interest recommendation. World Wide Web **26**(2), 615–629 (2023)
5. Dilshad, N., Ullah, A., Kim, J., Seo, J.: LocateUAV: unmanned aerial vehicle location estimation via contextual analysis in an IoT environment. IEEE Internet Things J. **10**(5), 4021–4033 (2022)
6. Ge, Z., Liu, S., Wang, F., Li, Z., Sun, J.: YOLOX: exceeding yolo series in 2021. arXiv preprint arXiv:2107.08430 (2021)
7. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 580–587 (2014)
8. Guo, S., Zhao, C., Wang, G., Yang, J., Yang, S.: EC^{2}Detect: real-time online video object detection in edge-cloud collaborative IoT. IEEE Internet Things J. **9**(20), 20382–20392 (2022)
9. Haq, M.A., Ruan, S.J., Shao, M.E., Haq, Q.M.U., Liang, P.J., Gao, D.Q.: One stage monocular 3D object detection utilizing discrete depth and orientation representation. IEEE Trans. Intell. Transp. Syst. **23**(11), 21630–21640 (2022)
10. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2961–2969 (2017)
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
12. Li, X., et al.: Generalized focal loss: learning qualified and distributed bounding boxes for dense object detection. In: Advances in Neural Information Processing Systems, vol. 33, pp. 21002–21012 (2020)
13. Liu, S., Qi, L., Qin, H., Shi, J., Jia, J.: Path aggregation network for instance segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 8759–8768 (2018)
14. Ou, Z., et al.: AD-RCNN: adaptive dynamic neural network for small object detection. IEEE Internet Things J. **10**(5), 4226–4238 (2023). https://doi.org/10.1109/JIOT.2022.3215469
15. Pang, J., Chen, K., Shi, J., Feng, H., Ouyang, W., Lin, D.: Libra R-CNN: towards balanced learning for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 821–830 (2019)
16. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016)

17. Redmon, J., Farhadi, A.: YOLO9000: better, faster, stronger. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7263–7271 (2017)
18. Redmon, J., Farhadi, A.: YOLOv3: an incremental improvement. arXiv preprint arXiv:1804.02767 (2018)
19. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, vol. 28 (2015)
20. Roboflow: Parcel detection dataset (2023). https://universe.roboflow.com/detectionbox
21. Tan, M., Pang, R., Le, Q.V.: EfficientDet: scalable and efficient object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10781–10790 (2020)
22. Tian, Z., Shen, C., Chen, H., He, T.: FCOS: fully convolutional one-stage object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9627–9636 (2019)
23. Ultralytics: YOLOv5: An open source real-time object detection system (2020). https://github.com/ultralytics/yolov5
24. Wang, C.Y., Bochkovskiy, A., Liao, H.Y.M.: YOLOv7: trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7464–7475 (2023)
25. Wang, H., Wang, Q., Zhang, H., Hu, Q., Zuo, W.: CrabNet: fully task-specific feature learning for one-stage object detection. IEEE Trans. Image Process. **31**, 2962–2974 (2022)
26. Wang, X., Kong, T., Shen, C., Jiang, Y., Li, L.: SOLO: segmenting objects by locations. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12363, pp. 649–665. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58523-5_38
27. Weng, X., Huang, Y., Li, Y., Yang, H., Yu, S.: Unsupervised domain adaptation for crack detection. Autom. Constr. **153**, 104939 (2023)
28. Xie, X., Cheng, G., Wang, J., Yao, X., Han, J.: Oriented R-CNN for object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 3520–3529 (2021)
29. Yang, L., et al.: PDNet: toward better one-stage object detection with prediction decoupling. IEEE Trans. Image Process. **31**, 5121–5133 (2022)
30. Zhai, Y., Yang, K., Zhao, Z., Wang, Q., Bai, K.: Geometric characteristic learning R-CNN for shockproof hammer defect detection. Eng. Appl. Artif. Intell. **116**, 105429 (2022)
31. Zhang, H., et al.: ResNeSt: split-attention networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2736–2746 (2022)
32. Zou, W., Zhang, Z., Peng, Y., Xiang, C., Tian, S., Zhang, L.: SC-RPN: a strong correlation learning framework for region proposal. IEEE Trans. Image Process. **30**, 4084–4098 (2021)

Fall Detection and Assessment Using Multitask Learning and Micro-sized LiDAR in Elderly Care

Shota Yamada[1(✉)], Hamada Rizk[1,2], Tatsuya Amano[1], and Hirozumi Yamaguchi[1]

[1] Osaka University, Osaka, Japan
{sho-yamada,hamada_rizk,t-amano,h-yamagu}@ist.osaka-u.ac.jp
[2] Tanta University, Tanta, Egypt
hamada_rizk@f-eng.tanta.edu.eg

Abstract. The increasing concern over the rapid aging population has brought to light a significant issue: the escalating number of falls among the elderly. As seniors grow older, they become more susceptible to physical ailments, leading to a higher frequency of falls. The lack of prompt assistance after a fall further compounds the problem, putting them at risk of severe consequences, including mortality. To address this pressing matter, the demand for fall detection systems in nursing homes and similar care facilities is on the rise. In response, our research proposes developing a point cloud-based fall detection system, complete with risk assessment capabilities, catering to the specific needs of the elderly. By employing advanced 3D LiDAR technology, we can scan the environment while preserving privacy. The algorithm employed then carefully analyzes this representation, extracting spatio-temporal discriminative features, thus enabling accurate fall detection. We have thoroughly evaluated the proposed system using collected data in our lab, and the results are promising, demonstrating its ability to detect fall events effectively. The successful implementation of this system could significantly enhance safety and improve the overall quality of life for the elderly population.

Keywords: Point cloud · LiDAR · Fall detection · Risk assessment · Healthcare

1 Introduction

Falls present a significant public health concern, particularly among the elderly. The World Health Organization (WHO) reports falls as the second leading cause of accidental or unintentional injury deaths globally, with adults aged 60 and above being at higher risk. In the United States, falls stand as the primary cause of injury-related deaths among those aged 65 and older, signifying an urgent issue that requires immediate attention. In Japan, a country recognized for its rapidly aging population, falls represent the most frequent cause of fatal accidents that occur within homes.

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 280–293, 2024.
https://doi.org/10.1007/978-3-031-63992-0_19

Falls have far-reaching consequences, often leading to severe injuries like fractures, head injuries, and traumatic brain injuries, which, in turn, can result in reduced independence, disabilities, and a decline in quality of life. For older adults, falls can even prove fatal due to their underlying health conditions, rendering them more susceptible to complications arising from such injuries. The increasing number of elderly individuals living alone adds further complexity to ensuring their safety and well-being, highlighting the urgent need to address fall-related risks. To protect the health and dignity of our aging population, it is crucial to develop robust fall detection systems. These systems should be capable of swiftly detecting falls and providing timely assistance, ensuring the well-being of seniors even when they are living independently. By addressing this pressing issue, we can significantly enhance the safety and quality of life for our elderly citizens.

In light of this critical challenge, researchers have been actively working on the development of diverse fall detection systems, utilizing a wide range of sensors and technologies. These include accelerometers [1–8], Wi-Fi [9], and video cameras [10–12]. By harnessing the potential of these tools, the systems can effectively detect falls by sensing sudden changes in acceleration or orientation or by analyzing images and video footage to recognize patterns commonly associated with falls.

Many of these systems utilize wearable sensors, such as smartwatches [3] or smartphones [7]. However, for older adults, the use of these devices can be inconvenient and uncomfortable, resulting in inconsistent or incorrect usage. Furthermore, fall detection systems that rely on visible cameras to capture videos [11,12] raise privacy concerns due to the sensitivity of the data, including images and location information. This concern holds particular importance for older adults, as they place a high priority on their privacy and are sensitive to issues of security and surveillance. Therefore, the installation of cameras for fall detection in their homes may encounter resistance, as it can be interpreted as an invasion of their privacy and a form of surveillance.

As a result, there exists a reluctance among certain older adults to fully embrace these systems, even if it means forgoing access to crucial fall detection services. However, there is a promising alternative in the form of radio-based fall detection, which ingeniously harnesses signals like WiFi RSSI or CSI [13], offering a privacy-conscious solution. This cutting-edge approach effectively mitigates the visual privacy concerns often associated with camera-based systems, presenting a more appealing option for those who prioritize their privacy. Nonetheless, radio-based fall detection is not without its challenges. It faces hurdles in terms of accuracy and the need for further refinement in technology for practical and widespread adoption.

Additionally, current fall detection techniques often lack the ability to assess the severity of falls, leading to inappropriate system responses. This limitation must be addressed urgently to guarantee precise and targeted assistance for the safety and well-being of the elderly population. Sustained research efforts are essential to develop a privacy-preserving, non-intrusive approach that can accurately detect falls and assess their risk level.

In recent times, remarkable progress has been achieved in the field of 3D range scanning, particularly with the emergence of LiDAR technology. LiDAR finds diverse applications across a broad spectrum, including human identification [14], tracking in public spaces [15–18], and activity recognition [19], among many others. One of the inherent advantages of LiDAR is its ability to prioritize privacy. By utilizing a 3D point cloud data capture approach, LiDAR effectively solves common privacy concerns that are often associated with camera-based systems. The point cloud data comprises purely non-personal information, represented by three-dimensional coordinates, making LiDAR a privacy-conscious and secure alternative.

In this paper, we present a system for fall detection utilizing low-cost, non-invasive LiDAR tailored for healthcare applications. Specifically, our system is run on Hitonavi-μ which is composed of a compact-size LiDAR device, a processing unit, and a battery [14]. By scanning and representing the environment in a point cloud format, our system achieves real-time fall detection on Hitonavi-μ across various scenarios while prioritizing individual privacy. This characteristic makes it well-suited for deployment in privacy-sensitive settings, including restrooms, where ensuring privacy is of utmost importance. Furthermore, the system demonstrates exceptional proficiency in evaluating the corresponding risk level of detected falls from *non-fall*, *low*, *moderate*, and *high* levels, depending on the activity during the fall, as it impacts the speed of the fall. This distinctive capability empowers the system to initiate prompt and appropriate actions, such as reaching out to emergency response teams or notifying family members, depending on the severity of the fall event.

Nevertheless, the system proposed faces two challenges. Firstly, extracting significant context from variable-sized, unordered, and unstructured point cloud representations proves to be a complex task in comparison to 2D images. Secondly, considering the constrained computational power available at the edge device, additional considerations require careful attention.

To address these challenges, we have devised a series of modules, introducing resolution enhancement, and downsampling techniques. Additionally, we have incorporated a computationally efficient feature extraction method based on the Fisher vector, which generates fixed-size distinctive signatures of falls by leveraging its symmetric functions. These extracted features play a vital role in our multitask learning approach, training a model that can simultaneously recognize fall events and assess their severity levels without compromising efficiency.

The proposed system underwent a comprehensive evaluation, employing data gathered from a diverse group of participants over several days. We also addressed the data imbalance by upsampling minority data. Impressively, the system achieved remarkable accuracy rates, detecting falls with 95.2% and accurately assessing fall risk levels at 94.6%.

An essential strength of the system is its unwavering dedication to safeguarding the privacy of the elderly while ensuring their safety through prompt and suitable actions determined by the recognized risk level. This aspect emphasizes the system's ability to provide effective fall detection services without compro-

mising the dignity and privacy of older adults, making it an ideal solution for maintaining their well-being in all settings. The impressive outcomes clearly highlight the revolutionary possibilities of Hitonavi-μ as the cutting-edge fall detection technology of the future.

The subsequent sections of this paper are organized as follows. In Sect. 2, we conduct a comprehensive literature review. Section 3 offers an overview of the proposed system, while Sect. 4 delves into its details. Moving on to Sect. 5, we explain the data collection process and the system's testing methodology. Lastly, we draw our conclusions in Sect. 6.

2 Related Work

Fall detection systems can be classified into three main types, depending on the sensing technology used: wearable-based, vision-based, and ambient-based systems.

Wearable sensors include accelerometers, gyroscopes, magnetometers, and electromyography. They analyze changes in body movements related to falls. Among these wearable sensors, accelerometers have received considerable attention in various studies (Wang et al., 2020). Accelerometers present numerous advantages, including their compact size, portability, cost-effectiveness, and non-intrusive nature, which help avoid privacy concerns. Moreover, the prevalence of smartphones and smartwatches equipped with built-in sensors makes them promising resources for real-world fall detection applications. Accelerometers offer flexibility in their placement on various body parts, including the waist (i.e., waist [1], thigh [2], wrist [3], sole [4], torso [5], chest [6], ankle [8], etc.) For instance, [2] proposed a patient-specific fall detection system with a single sensor. In contrast to many existing research studies that solely focus on detection, the proposed approach goes beyond detection by also addressing the aspect of prediction. The system consists of two branches for fall prediction and detection individually. They adopted a threshold approach for prediction and a machine-learning approach for detection. This is because the fall prediction requires a low computational cost algorithm to alarm the patients before falls occur. They successfully reduced the false alarm rate by taking the fall prediction result into the detection process.

– *In contrast to the wearable-based approach, point cloud-based fall detection systems do not require individuals to wear sensors or devices, making them more user-friendly and suitable for scenarios where individuals may not be willing or able to wear sensors.*

Vision-based fall detection has been extensively explored and demonstrated promising performance. Common visual sensors used in this context include infrared cameras [10], RGB cameras [11,12], and RGB-D cameras [20]. [11] proposed a vision-based method using a monocular camera for monitoring elderly people. They first extract foreground objects and conduct segmentation. Then, they apply ellipse fitting for the approximation of human-body regions. The

feature is extracted from those regions and fed into a multi-class support vector machine followed by a context-free grammar-based method that incorporates longer-range temporal constraints to validate the detected falls. [12] proposed an algorithm that determines regions of the human silhouette based on the centroids of different regions of the human body part instead of ellipses. This reduced the false alarm rate of fall-like movements, such as squatting and sitting, which had been a problem with ellipse-based detection. Images obtained from these types of sensors offer detailed scene information; however, they also raise concerns regarding privacy violations, as they may capture personal and environmental information in living spaces. RGB cameras are widely used due to advancements in computer vision technology. However, they face challenges related to their sensitivity to environmental conditions. In particular, their performance is significantly affected in low-light settings, making them less effective during nighttime. Additionally, distinguishing between real falls and certain behaviors like lying down can be difficult, leading to a higher rate of false detections.

– *Compared with visual sensors, point clouds are less intrusive in terms of privacy since they don't capture detailed color or texture information about individuals. Point clouds are also less affected by changes in lighting conditions, making them more reliable in scenarios with poor lighting.*

Ambient sensors offer a privacy-preserving approach for fall detection. Common examples of these sensors are pressure tiles, magnetic switches, seismic sensors, and Wi-Fi-based sensors [9], etc. The approach presented in the paper [9] utilized Wi-Fi signals and gained popularity due to its non-intrusive and widespread availability. Their proposal involved real-time, cost-effective indoor fall detection using CSI phase differences as the primary signal. However, the system's effectiveness is limited in multi-person scenarios, as it works effectively only when a single individual is in motion while others remain stationary. The benefit of using ambient sensors lies in their non-intrusive nature. These non-contact sensors do not require direct attachment to the body, allowing users to have a comfortable and unrestricted experience in their daily activities. In addition to being non-intrusive, ambient sensors can detect changes in the surrounding environment, providing a comprehensive monitoring approach. However, they are susceptible to interference from external factors such as walls, furniture, or other electronic devices, leading to poor detection. e.g., the pressure tile has the disadvantage of sensing all objects in its vicinity, resulting in an increased false alarm rate.

– *On the contrary, point clouds are less affected by changes in lighting, temperature, or other environmental conditions compared to ambient sensors, which can be sensitive to variations in their surroundings.*

3 System Overview

The proposed system consists of two main stages: **the sensing stage** and **the intelligence stage**. In the sensing stage, our innovative proprietary device,

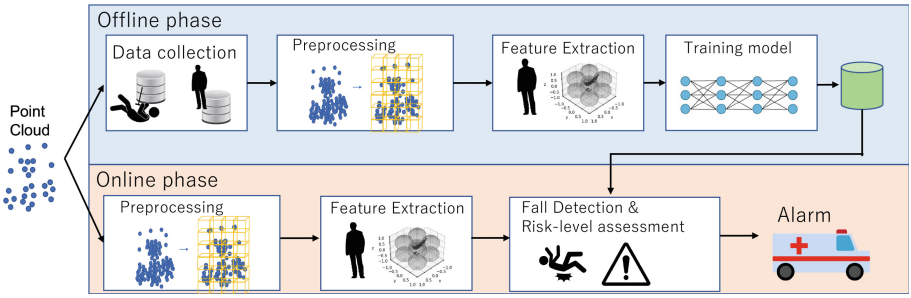


Fig. 1. System overview

referred to as Hitonavi-μ introduced in our previous work [14], plays a central role. This device has been ingeniously designed to support various applications in smart environments. By simply plugging in Hitonavi-μ, the system gains the impressive capability to scan its surroundings and create a detailed 3D point cloud representation of the scene. The main goal of this sensing functionality is to create a signature database for detecting falls. This is done by gathering point cloud scans of individuals in simulated falling scenarios. These captured point clouds are then efficiently processed within the Intelligence stage. The intelligence has two different phases (i.e., offline phase and online phase) as shown in Fig. 1. The offline phase involves model training, which includes gathering data, preprocessing, and extracting features. On the other hand, during the online phase, the system detects falls and assesses the risk level when it's operating in real-world settings. If a fall happens, the system alerts healthcare services, ensuring prompt treatment and preventing severe damage.

4 The System Details

In this section, we offer a detailed explanation of the different modules in the proposed system. These modules cover the stages of sensing and intelligence including multiple modules.

4.1 The Sensing Stage

This section provides an overview of the key features of the proposed sensing device, referred to as Hitonavi-μ. Hitonavi-μ is a compact, battery-equipped version of Light Detection and Ranging (LiDAR), accompanied by its computing unit. LiDAR, in general, is a remote sensing technology used to measure the distance to the surface of surrounding objects. It achieves this by calculating the time taken for a laser pulse to travel from the sensor to the objects/surfaces and back, thus estimating the distance. LiDAR technology has found extensive applications, including obstacle detection in autonomous cars [21], as well as human and robot detection, tracking, and navigation in public indoor environments [18]. However, the existing LiDAR systems used in these applications are

typically large and expensive, making them more suitable for commercial environments like malls and museums. This restricts their widespread adoption as privacy-preserving vision technology, especially in private settings.

To address this limitation, we employ a micro-sized LiDAR sensor[1] This sensor is relatively smaller and lighter compared to consumer-grade LiDAR devices, with dimensions of 44 mm width, 24 mm depth, and 16 mm height. The design rationale behind Hitonavi-μ is to enable its deployment in any indoor environment, including homes where camera-based systems may not be suitable. This paves the way for the next generation of privacy-preserving vision technology.

The LiDAR sensor is linked to a battery-equipped computing unit, specifically the Raspberry Pi 4 Model B [22], which serves as the processing platform for our algorithms. The Raspberry Pi 4 Model B is a compact single-board computer suitable for mobile devices. It features an ARM Cortex-A72 quad-core CPU running at 1.5 GHz and is equipped with 4 GB of RAM. Figure 2 illustrates the composition of Hitonavi-μ, consisting of the small-sized sensor and the Raspberry Pi compute module. The sensor operates at an average frame rate of approximately 30 frames per second.

4.2 The Intelligence Stage

In this section, we describe the process shown in Fig. 1, including data collection, preprocessing, feature extraction, and classifier.

The Data Collection. During this phase, we collect 3D point cloud data from individuals to create a unique database. This is done by using the sensor to record the scene and then sending the collected point cloud to our server using the built-in WiFi on Hitonavi-μ. Figure 3 and Fig. 4 show the scene of data collection and corresponding point cloud captured by the sensor. We position the sensor on a tripod, which allows it to capture objects and people around it within a distance of up to 3 m. As can be seen in visualization of point cloud, it is challenging to know the attribute (e.g., gender, age) of people by direct observation, making it suitable in privacy-related situations.

Preprocessing. This section describes modules to handle input point clouds for system stability and improvement. There are two modules for frame integration, downsampling.

Frame Integration. In this module, frame integration serves two primary purposes. Firstly, it combines multiple successive frames to enhance the sensor's resolution, resulting in a more intricate representation of the captured scene. While commercial LiDAR systems typically gather a substantial number of points (around $\sim 100k$ points) within a single point cloud, our Hitonavi-μ sensor averages just $0.8k$ points. This lower point density preserves privacy but poses

[1] The sensor used in this study was manufactured by MagikEye company.

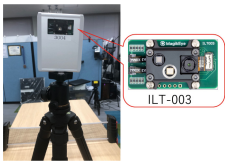


Fig. 2. Hitonavi-μ

Fig. 3. Scene of Data Collection

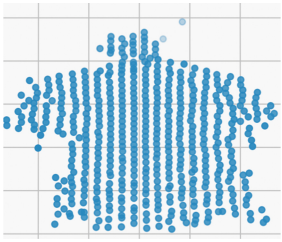


Fig. 4. Point cloud as captured by the sensor

recognition challenges. To address this, we utilize the sensor's high sampling rate to boost its resolution by merging m consecutive frames into a single frame with higher point density. This strategy enhances visual clarity and data accuracy.

Secondly, frame integration allows us to capture the temporal changes within the time sequences. This capability is crucial for fall detection, which involves capturing sequences of events that happen over time. Falls often unfold gradually, involving a series of movements, shifts, and changes in position. By observing these changes across a sequence of frames, we can more accurately identify and differentiate a fall from other activities or situations.

Downsampling. This module serves two primary objectives. Firstly, we aim to reduce the computational cost by removing some points from the point clouds. Processing point clouds using machine learning requires substantial computational resources, which can be challenging, especially when deploying the system on edge devices with limited processing power. Downsampling points is one approach to address this challenge. Secondly, we aim to mitigate sampling biases. Point cloud representations often exhibit non-uniform distribution, resulting in varying densities across the scene. Certain areas may have denser point clouds due to factors like sensor-object distances. This non-uniform density can impact the performance of our system, as areas with more points contain more information compared to areas with lower density. Our second goal is to address this issue.

To accomplish these goals, we utilize Voxel downsampling. This approach involves dividing the 3D space into grids, creating individual boxes known as 'Voxels.' Each voxel then generates a single point by averaging all the points within it. These processes can be easily performed using the distributed open3D library.

Feature Extraction. This section introduces our proposed feature extraction technique based on Fisher Vector (FV) representations. Before we delve into the feature extraction technique, we discuss the challenges associated with representing a 3D point cloud. The primary challenge stems from the fact that a 3D point cloud lacks a structured grid, unlike images that have a fixed grid

arrangement. This variance in the number of points within a point cloud makes it challenging to apply traditional computer vision techniques. To address this issue, voxelization is commonly employed, which involves discretizing the 3D space into cells to facilitate the application of Convolutional Neural Networks (CNNs). However, voxelization introduces quantization loss and imposes high computational resources. Additionally, there is a challenge related to the ordering of points within the point cloud. It is crucial for models to produce consistent results regardless of the order in which the data points are presented.

To address these challenges, we utilized the FV representation. The FV representation is employed to create distinctive signatures for the data of different activities including fall, which can vary in size, such as 3D point cloud frames. This representation inherently captures the spatial locality of the points. It defines the signature as the deviation of the 3D points from a generative model, such as a Gaussian Mixture Model (GMM). This is achieved by calculating the gradients of the sample's log-likelihood with respect to the model parameters, including weight, mean, and covariance. Furthermore, the FV maintains a fixed-size grid structure, making it a convenient input for any classifier.

The key advantage of the Fisher vector representation is its independence from the sample size, making it a suitable choice for representing point cloud data.

Classifier Training. In this section, we explain our approach to constructing a classification model capable of accurately identifying falls and assessing associated risks using the extracted Fisher representations.

We employed multi-task learning for our classifier, which involved fall detection and risk assessment tasks. Multi-task learning is an approach in machine learning where a single model is trained to perform multiple related tasks simultaneously. Instead of training separate models for each task, the multi-task learning model shares information and parameters across tasks to exploit their inherent relationships.

The motivation for utilizing multi-task learning is to have a simplified model that can efficiently run on edge devices. By training multiple tasks with a single model, we eliminate the need for separate models for each task. This reduces the complexity of the model and allows for efficient utilization of memory and computational resources. Moreover, multi-task learning enables us to train multiple related tasks simultaneously, even with less data compared to training separate models for each task. The sharing of features and parameters promotes information exchange between tasks, leading to improved data efficiency. This advantage is crucial due to the difficulty of collecting fall data.

The architecture of the network is shown in Fig. 5. We used three layers of fully connected neural layers for the shared components with 512, 256, and 128 units, respectively. As for the subsequent branches, both fall detection and risk-level detection consist of two fully connected layers with 64 and 32 units. The activation function used throughout the model is Rectified Linear Unit (ReLU). Finally, the output layers of each branch are designed with a number of neurons

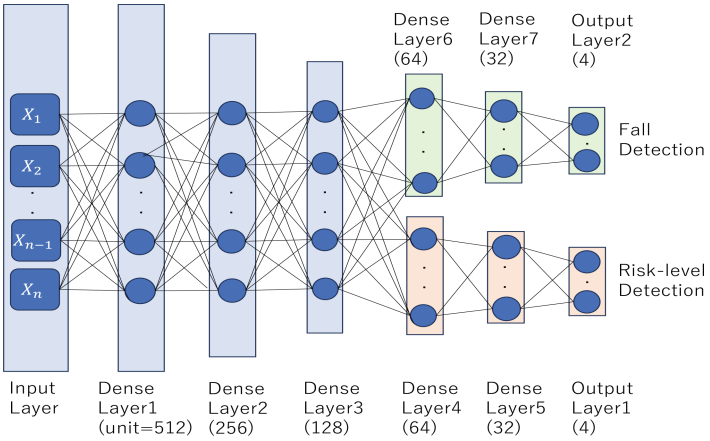


Fig. 5. Model architecture

corresponding to the number of fall classes and risk levels under consideration, with a Softmax activation function.

The term "risk level" specifically refers to the severity of falls that have occurred while engaging in various activities. Our categorization comprises four classes: non-fall, low, moderate, and high, which are labeled by experts in our lab who were familiar with fall events in their research [23]. The "non-fall" classification pertains to activities of daily living (ADLs) where the risk of falling is minimal. However, the other categories relate to instances of falls, indicating different levels of risk associated with each. Our classification of fall-related events is primarily determined by the particular activities being performed during the fall. Specifically, falls that occur during standing activities or transitioning from standing to sitting are labeled as "low risk." Falls that happen during activities of picking up objects or sitting down are categorized as "moderate risk." Conversely, falls that occur while walking or transitioning from sitting to standing are considered "high risk." These are basically associated with the poses, particularly the positions of heads. Generally, higher head positions lead to higher risks when falls happen.

5 Evaluation

In this section, we conduct an evaluation of the proposed system's end-to-end performance. To ensure a thorough assessment, we considered various activities of daily life (ADL), encompassing standing, walking, sitting, sit-to-stand, stand-to-sit, and picking up objects. The dataset utilized for these experiments was collected from seven subjects, each exhibiting different attributes such as height and body shape, among others. To gather fall events data, we recorded falls occurring in different directions – front, right, left, and back – a total of ten times for each direction. Note that we consider only front falls for the walking activity.

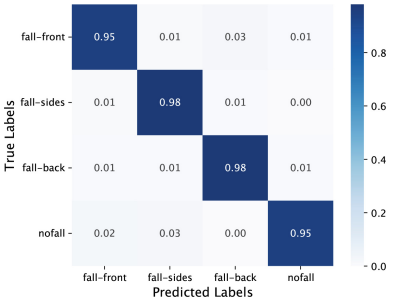


Fig. 6. Fall detection

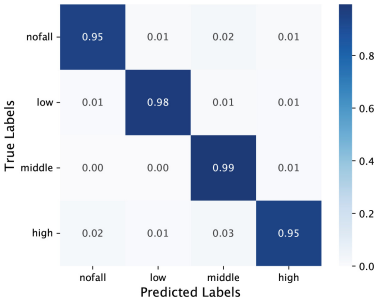


Fig. 7. Risk-level detection

Additionally, for the collection of ADL data, we recorded stationary activities (i.e., standing and sitting) for a duration of 30 s, while mobility activities (i.e., walking, sit-to-stand, and stand-to-sit) were recorded ten times each.

For the purpose of evaluation, we split the data into an 80% training set and a 20% testing set.

5.1 System Parameters

The default parameters of each module are selected (based on the performance obtained by grid search) as follows: Number of frames integration = 9 and Downsampling voxel grid size = 5. The details of the grid search are summarized in Table 1.

Table 1. Details of the grid search

Target module	Parameter	Value
Frame integration	Number of integrated frames	1, 3, 5, 7, 9, 11
Voxel Downsampling	Grid length(r)	5, 15, 30, 50

5.2 Results

In this section, we present the outcomes of the system's performance evaluation. The confusion matrices for fall and risk-level detection are illustrated in Fig. 6 and Fig. 7, respectively, showing accuracies of 95.2% and 94.6%.

These promising outcomes highlight the effectiveness of our point cloud-based fall detection system. Its ability to achieve such high accuracies in detecting falls and assessing risk levels reaffirms its credibility and potential for real-world healthcare applications.

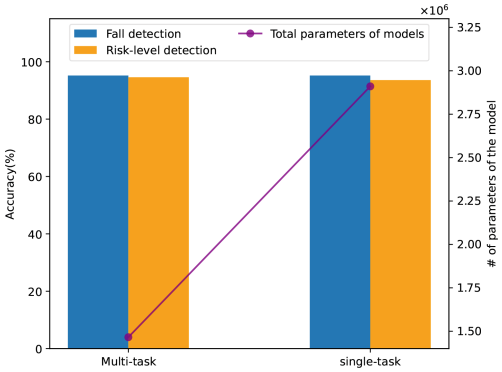


Fig. 8. Comparison between single-task and multi-task learning

Comparison with Single Task Model. This section investigates the performance comparison with the single-task model to show the efficacy of multi-task learning. Figure 8 shows the results, including the accuracy and memory consumption. The system built with a single-task-based model incurs twice the burden in memory consumption compared to the multi-task-based one. Implementing this on edge devices with limited computational power and memory capacity is not suitable. Additionally, it is notable that the model with multi-task learning achieves slightly better accuracy compared with that with single-task learning. i.e., both 95.2% for fall detection, while 94.6% and 93.6% for risk detection. As such, the multitask learning model is a better choice than the single-task model as described in Sect. 4.2.

6 Conclusion

In this paper, we introduced a novel fall detection system with risk-level assessment, utilizing privacy-preserving point clouds. Our system includes modules to achieve accurate recognition while minimizing computational overhead. These modules enhance point cloud resolution, downsample point clouds, and learn salient representations through the Fisher vector. To optimize the recognition of falls and simultaneously assess the associated risk level, we employed a multi-task learning approach. Our evaluation showed remarkable results: 95.2% accuracy in fall detection and an outstanding 94.6% accuracy in risk-level assessment, with strong generalization across diverse scenarios. These findings highlight Hitonavi-μ's potential as a transformative plug-and-play healthcare support device with far-reaching implications for assistive technology. Future plans involve large-scale deployment in residential and hospital settings, aiming to positively impact healthcare outcomes.

Acknowledgment. This work was supported by JST, A-STEP Grant Number JPMJTR20RV, Japan.

References

1. Saleh, M., Le Bouquin Jeannès, R.: Elderly fall detection using wearable sensors: a low cost highly accurate algorithm. IEEE Sens. J. **19**(8), 3156–3164 (2019)
2. Saadeh, W., Butt, S.A., Altaf, M.A.B.: A patient-specific single sensor IoT-based wearable fall prediction and detection system. IEEE Trans. Neural Syst. Rehabil. Eng. **27**(5), 995–1003 (2019)
3. De Araujo, I.L., Dourado, L., Fernandes, L., Andrade, R.M.C., Aguilar, P.A.C.: An algorithm for fall detection using data from smartwatch. In: Proceedings of the 13th Annual Conference on System of Systems Engineering (SoSE 2018), pp. 124–131 (2018)
4. Zitouni, M., Pan, Q., Brulin, D., Campo, E.: Design of a smart sole with advanced fall detection algorithm. J. Sens. Technol. **9**(04), 71 (2019)
5. Thilo, F.J.S., Hahn, S., Halfens, R.J.G., Schols, J.M.G.A.: Usability of a wearable fall detection prototype from the perspective of older people-a real field testing approach. J. Clin. Nurs. **28**(1–2), 310–320 (2019)
6. Wu, T., Gu, Y., Chen, Y., Wang, J., Zhang, S.: A mobile cloud collaboration fall detection system based on ensemble learning. In: Proceedings of the 22nd International ACM SIGACCESS Conference on Computers and Accessibility, pp. 1–7 (2020)
7. Islam, Z.Z., Tazwar, S.M., Islam, M.Z., Serikawa, S., Ahad, M.A.R.: Automatic fall detection system of unsupervised elderly people using smartphone. In: 5th IIAE International Conference on Intelligent Systems and Image Processing, vol. 5 (2017)
8. Xi, X., Tang, M., Miran, S.M., Luo, Z.: Evaluation of feature extraction and recognition for activity monitoring and fall detection based on wearable sEMG sensors. Sensors **17**(6), 1229 (2017)
9. Wang, H., Zhang, D., Wang, Y., Ma, J., Wang, Y., Li, S.: RT-Fall: a real-time and contactless fall detection system with commodity WiFi devices. IEEE Trans. Mob. Comput. **16**(2), 511–526 (2016)
10. Mastorakis, G., Makris, D.: Fall detection system using Kinect's infrared sensor. J. Real-Time Image Process. **9**, 635–646 (2014)
11. Feng, W., Liu, R., Zhu, M.: Fall detection for elderly person care in a vision-based home surveillance environment using a monocular camera. Sig. Image Video Process. **8**, 1129–1138 (2014)
12. Chua, J.-L., Chang, Y.C., Lim, W.K.: A simple vision-based fall detection technique for indoor video surveillance. Sig. Image Video Process. **9**, 623–633 (2015)
13. Sugimoto, Y., Rizk, H., Uchiyama, A., Yamaguchi, H.: Towards environment-independent activity recognition using Wi-Fi CSI with an encoder-decoder network. In: Proceedings of the 8th Workshop on Body-Centric Computing Systems, BodySys 2023, pp. 13–18. ACM, New York (2023)
14. Yamada, S., Rizk, H., Yamaguchi, H.: An accurate point cloud-based human identification using micro-size LiDAR. In: Proceedings of the 2022 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops), pp. 569–574. IEEE (2022)
15. Ukyo, R., Amano, T., Rizk, H., Yamaguchi, H.: Pedestrian tracking using 3D LiDARs - case for proximity scenario. In: Proceedings of the IEEE Intelligent Transportation Systems Society Conference Management System (ITSC 2023) (2023)

16. Ohno, M., Ukyo, R., Amano, T., Rizk, H., Yamaguchi, H.: Privacy-preserving pedestrian tracking using distributed 3D LiDARs. In: Proceedings of the 2023 IEEE International Conference on Pervasive Computing and Communications (PerCom), pp. 43–52 (2023)
17. Rizk, H., Yamaguchi, H., Youssef, M., Higashino, T.: Laser range scanners for enabling zero-overhead WiFi-based indoor localization system. ACM Trans. Spatial Algorithms Syst. **9**(1), 1–25 (2023)
18. Rizk, H., Yamaguchi, H., Youssef, M., Higashino, T.: Gain without pain: enabling fingerprinting-based indoor localization using tracking scanners. In: Proceedings of the 28th International Conference on Advances in Geographic Information Systems, pp. 550–559 (2020)
19. Rizk, H., Okochi, Y., Yamaguchi, H.: Demonstrating hitonavi-μ: a novel wearable LiDAR for human activity recognition. In: Proceedings of the 28th Annual International Conference on Mobile Computing And Networking, MobiCom 2022, pp. 756–757 (2022)
20. Min, W., Yao, L., Lin, Z., Liu, L.: Support vector machine approach to fall recognition based on simplified expression of human skeleton action and fast detection of start key frame using torso angle. IET Comput. Vis. **12**(8), 1133–1140 (2018)
21. Zhang, J., Zhao, X., Chen, Z., Zhejun, L.: A review of deep learning-based semantic segmentation for point cloud. IEEE Access **7**, 179118–179133 (2019)
22. Raspberry pi foundation. https://www.raspberrypi.com/products/raspberry-pi-4-model-b/. Accessed 30 Nov 2021
23. Fukushima, Y., Miura, D., Hamatani, T., Yamaguchi, H., Higashino, T.: MicroDeep: in-network deep learning by micro-sensor coordination for pervasive computing. In: 2018 IEEE International Conference on Smart Computing (SMARTCOMP), pp. 163–170 (2018)

Online Dynamic Path Planner for UAVs

Manisha Wadhwa[1], Neelima Gupta[2], and Sanjay Madria[3(✉)]

[1] Department of Computer Science, Ram Lal Anand College, University of Delhi, New Delhi, India
mwadhwa@cs.du.ac.in
[2] Department of Computer Science, University of Delhi, New Delhi, India
ngupta@cs.du.ac.in
[3] Department of Computer Science, Missouri University of Science and Technology, Rolla, USA
madrias@mst.edu

Abstract. Unmanned Aerial Vehicles (UAVs) flight path generation that passes through specified waypoints while satisfying spatio-temporal task points (delivering goods or taking aerial pictures) is an important application in battlefield and disaster management among others. Most of the previous work on UAV path planning has been on finding a shortest distance path from source to destination. However, the shortest distance does not always mean fastest route due to congestion or other environmental conditions such as rain, storm, wind etc, passing by aerial objects and no fly zone that may affect the speed/direction of the UAV. Moreover most of these approaches are offline (using A* type algorithms) and hence cannot respond to the dynamically changing environmental conditions when the number of dynamic task points increases. In this paper, we present a greedy Online Dynamic Path Planning algorithm (ODP) considering the dynamically changing requests of tasks, and the environmental conditions. We model the problem as an online multiple choice knapsack problem where-the task points have rewards (earned on completing tasks on the path) and the aim is to maximise the sum of reward points earned on completing the tasks on the path while satisfying the way points with a limit on the total traverse time. To analyze the efficiency of ODP (an online approach), we compare its performance empirically with that of the optimal offline (brute force) algorithm. It was found that ODP provides a competitive ratio (competitive ratio is defined as the ratio of performance of an online algorithm to that of the optimal offline algorithm in the context of algorithm analysis) of 0.8 indicating a near-optimal performance. That is, ODP returns a path that obtains about 80% of the optimal rewards.

Keywords: Path Planning · Knapsack · UAV · Online

1 Introduction

Various industries have emerged to fulfill the demand for services provided by Unmanned Aerial Vehicles (UAVs) such as delivering goods and collecting

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 294–315, 2024.
https://doi.org/10.1007/978-3-031-63992-0_20

imagery data at various locations (e.g., for battlefield and disaster applications). Thus, UAV path planning is an important problem where time-constrained waypoints (in flight path) have been defined and UAVs have to efficiently satisfy the tasks at different locations requested by the ground station while traversing the waypoints. Since the tasks may be spread over a wider space, one needs to plan the path of the UAV in such a way to finish as many tasks as possible satisfying the waypoints constrained with location and time.

UAV path planning is one of the most fundamental problems addressed in the literature [6,10,17,20,21]. Most of the previous work on UAV path planning has been on finding a shortest distance path from source to destination using A* algorithm [8]. However, the shortest distance does not always mean fastest route due to congestion or other environmental conditions such as rain, storm, wind etc, passing by aerial objects and no fly zone that may affect the speed/direction of the UAV from time to time. Improved A^* [21] handles bad weather conditions predicted in advance using climate models and consider them as no-fly zones and thus do not perform well in real-time (still an off-line scheme). In this paper, we present an online algorithm considering the dynamically changing requests of tasks, and the environmental conditions where the bad environmental conditions (which can affect the speed of the UAV) are encountered on-the-go. Hence, an off-line algorithm that provides a distance-efficient path cannot be modified to obtain a time-efficient path.

[12] maximizes the number of tasks performed, given a trajectory defined by a sequence of waypoints without deviating too much from the trajectory. They do capture task priorities but they do not consider dynamically changing environmental conditions as their approach is offline in the sense that they solve an optimization problem with the pre-defined waypoints and the newly generated task points as inputs. They have considered their problem in open space meaning UAVs can reach to any point.

In this work, each task has some reward value that one gets on completing the task. Thus, we consider the problem of finding a path, following a trajectory defined by the waypoints, that maximizes the rewards associated with tasks. We would like to consider such task points so long as they can be covered within the time constraint. A weighted directed graph is used to establish an initial structured representation of the environment, with vertices representing the three dimensional waypoints and task points and, edges representing the connectivity between them and the weights on the edges represent the distance between the two end points; the UAV can travel only along the edges of the graph.

Sometimes a task point is difficult to reach as it may be located in a congested area or in a difficult terrain. Thus, we also associate a quality-of-task (QoT) with each task, $0 \leq QoT \leq 1$; a smaller value indicating difficult to reach task point. A point located in such an area may cost us more in terms of fuel consumption or take more time and we may want to avoid it even if it has high rewards.

Formally, we model the problem as a multiple-choice Online Knapsack Problem [15] where we have a set of task points with associated reward points r_i and the time to complete the task tc_i with a knapsack capacity of kc on the total time taken by the UAV on completing all the tasks. In addition to the

time constraint, other constraints such as turning angle and no-fly zone have also been considered in this work. At every node v, for each outgoing edge, we check the feasibility in terms of turning angle and no-fly zone. For each feasible edge (v, w), we compute the speed with which the UAV can fly depending upon several environmental factors such as weather conditions, other aerial objects crossing the path to avoid collision and, difficult terrain. Thus the time to reach does not only depend upon the distance of w from v (given by the weight on the edge (v, w)) but also on the dynamically changing environmental conditions. We then greedily visit a feasible out-neighbour w of v that maximizes r_w/t_w, where t_w is the sum of the time to reach the task point and the time to complete the task. The algorithm works like a multiple-choice online algorithm in the sense that, at a node v, only the task points that are out-neighbours of v are added to our known set of task points and we have to select the best from them.

To the best of our knowledge, this is the first work that considers the dynamic UAV path planning by modeling it as a greedy algorithm using the online multiple-choice knapsack problem [5], referred to as ODP (Online Dynamic Path Planning). The online aspect of the problem comes from the sequential presentation of tasks and the algorithm has to choose at most one task point from the set, where the decision is based upon the current state of the knapsack and the available information at that moment. We do not have the prior knowledge of future tasks when making decisions about the current task. As a result, the algorithm makes irrevocable choices based on incomplete information and the decision to include a task in the knapsack must be made immediately.

Based on the experimental results, the ODP algorithm is shown to provide a competitive ratio of about 0.8 when compared to the brute force approach which suggests that the online algorithm (ODP) is able to achieve a solution that is very close to the offline algorithm (brute force) for a wide range of input scenarios. This implies that the online algorithm is effective in completing high reward tasks while operating in real-time or in online settings where complete information is not available in advance.

2 Related Work

[8] introduced the A* search algorithm for graph traversal by combining elements of Dijkstra's algorithm and heuristic search to find optimal path while considering both the cost to reach a node and an estimate of the remaining cost to the goal but the trade-off between the optimality and computational efficiency is not explicitly explored. [3] proposed a two step path-planning algorithm for UAVs using Voronoi polygons [7] by assigning length cost to edges. This work does not address the practical constraints relevant to UAV path planning, such as energy consumption, uncertainties or mission-specific requirements. [16] presented two methods of path planning of UAV using A* algorithm, and genetic algorithm where the path is composed of the lines on the Voronoi diagram with the edge cost. The UAV's turning angle constraint was considered, path was smoothed by a geometry method. This work lacks the comparison with existing approaches

to determine the effectiveness of proposed approach. [22] used Ant Colony Optimization (ACO) algorithm for UAV path planning by dividing flying area into grids and optimizing path between grid point and destination point. Their simulation in two dimension proved that ACO can effectively and quickly accomplish UAV path planning but since it includes a series of connected points, it still need smoothness processing in order to be applied for real UAV flying. [17] relates to A* type algorithm in finding a path from source to destination keeping no fly/exclusion zones and turning angle in consideration while addressing the issue of (deconfliction) avoidance of collisions or conflicts with other UAVs, obstacles, or airspace regulations. They formulate the problem as a multi-objective optimization task, where the objectives include finding the shortest path while ensuring deconfliction with other UAVs and obstacles in three dimension. They have not discussed about the performance of the approach developed by them moreover it can not be used in real world UAV path planning as it does not take environmental conditions into account.

[1,10,11] and [13] provide a comprehensive review of path planning techniques for unmanned aerial vehicles (UAVs) and discuss their strengths, limitations, and challenges. They also discuss the impact of various factors such as environment modeling, obstacle avoidance, optimization criteria, and real-time considerations on path planning algorithms. They also provide insights into different path planning algorithms commonly used in UAV applications.

[6] proposed A* type algorithm by adding direction parameter considering the barriers between source and destination and focused on finding an optimal flight path in two dimension. This work does not consider the flight parameters and environmental constraints such as limited or no flying range into account.

Some researchers [6,17,21] have attempted to find a paradigm using A* algorithm to find cost optimum path from source to destination but it finds the result in exponential time; not feasible for real time scenarios. Also, the constraints - QoS, trajectory, uncertainties like thunder, rain, lightning etc., variable speed, limited driving range are not taken into account. Some researchers [12] have not considered these constraints but finds a path by going forward and choosing the node that seems best at that particular point of time. It would give a solution in a feasible time adhering to real time constraints on total time due to limited driving range, QoS, angle component and timestamps.

3 System Model and Nomenclature

Here we discuss the system model, basic terminology and notations used.

Task Points: Each task point is a 3D point where UAV performs some task (such as delivering some goods or clicking aerial pictures). Thus, a task point is specified by the tuple $<id, x, y, z, r, tc, QoT>$ where id = task id, x = latitude, y = longitude, z = altitude, r = reward associated, tc = Time to complete (in seconds) and QoT = Quality of Task point as explained below.

Reward: Each task point tp has a reward r associated with it ($tp.r$). For example, in case of delivering food packets, it could mean a bigger order; for a TV

channel collecting pictures from a prime event may fetch them more viewership leading to higher Television Rating Point (TRP).

Quality of Task Point: Each task point tp also has quality of accessibility QoT associated with it ($tp.QoT$). It refers to the accessibility and reachability of the task points. An easy-to-reach task point has higher value of QoT as compared to another task point that is difficult to reach. It takes values in the range from 0 to 1. [19] discusses that the locations lying in the open space can be easily covered by UAV in comparison to congested areas. We have assigned QoT values with different numerical values. Interpretation is as follows:

0.05: Difficult Terrain - task point in an area with challenging terrain.

0.10: Congested Areas - task point in a heavily congested area, which may pose challenges for navigation and maneuvering.

0.15: High-rise areas - task point in an area with tall buildings or structures, potentially affecting line-of-sight communication or requiring special consideration for aerial navigation.

0.20: Delivery point on the main road - task point corresponds to a delivery location on a primary road or a major intersection.

0.40: Delivery point in streets - task point represents a delivery location situated within smaller streets or narrower pathways.

0.50: Metropolitan Areas - task point in a densely populated urban area, typically associated with cities or metropolitan regions.

0.60: Delivery point on a terrace - task point corresponds to a delivery location situated on the rooftop or terrace of a building.

1.00: Task point in Open Area - task point located in a region of high quality of service, typically indicating an open area without significant obstacles.

Waypoints: 3D points which define the flight path trajectory. Each waypoint is defined by the tuple $<id, x, y, z, ts>$ where id = waypoint id, x = latitude, y = longitude, z = altitude, ts = timestamp. Time stamps are defined as $[yyyy, mm, dd, hh, mn, ss, ms]$ where $yyyy$ = year, mm = month, dd = day, hh = hour, mn = minutes, ss = seconds, ms = milliseconds. Each waypoint has time stamp associated with it that need to be covered on or before by the UAV.

Angle Component (θ): Let θ be the angle by which UAV needs to turn to reach a task. The angle between vector **ab** and **bc** is described as:

$$\theta = \arccos\left(\frac{\mathbf{ab} \cdot \mathbf{bc}}{|\mathbf{ab}||\mathbf{bc}|}\right)$$

where $\cdot$ denotes the dot product between vectors **ab** and **bc**. arccos is the inverse cosine function. $|\mathbf{ab}|$ and $|\mathbf{bc}|$ represent the magnitudes (lengths) of vectors **ab** and **bc** respectively. A UAV has a limit on the angle by which it can rotate. Thus, a task point is feasible if $\theta \leq \tau$. The value of θ must be lesser in order to provide a smooth path and at the same time it should allow the UAV to explore a wide variety of task points. Hence an optimum value of τ is chosen to be 60° in order to balance the trade-off as discussed in [4] and [14].

Uncertainty Parameter (μ_e): At any point of time, μ_e denotes if one or more bad weather condition is present along an edge e or if the weather is clear along e.

[18] lists down various weather conditions affecting the speed of UAV according to their severity. According to the paper, thunderstorm affects the UAV most severely whereas extreme temperatures may not pose immediate danger to the UAV. A high value of μ_e thus may denote the presence of thunderstorm on e whereas a low value may denote extreme temperatures and a value of 0 denotes clear weather. We assign following values to various weather conditions according to their severity as discussed in [18]:

0.95: Thunderstorm - It can impact visibility and may pose risks to the operation of UAV.

0.90: Fog/Clouds - presence of haze or fog in the area can reduce visibility and impact the operation of UAV.

0.85: Snow - Icing can affect the UAVs aerodynamics and performance, potentially leading to control problems and instability.

0.80: Rain and Hail - hail and rain in the area can cause damage to UAV and pose risks to safety.

0.70: Windy - Strong winds can affect a UAV's stability and control.

0.60: Passing by aerial vehicle - presence of another aerial vehicle in the vicinity may introduce additional uncertainity or safety concerns.

0.50: Moisture and Humidity - Exposure to moisture and water can cause the drone's flight path to be disrupted due to potential electronics malfunctions or loss of control.

0.40: Temperature Extremes - Extreme cold or heat can impact the UAVs battery performance and overall functionality, but it might not immediately pose as much danger as other conditions.

0.00: No significant uncertainties - no adverse conditions present in the area.

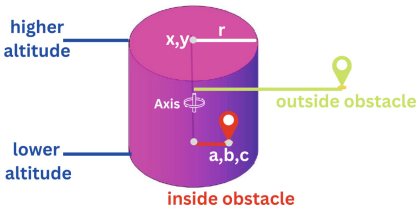


Fig. 1. Obstacle/No-fly zone

Speed (u_x): A UAV flies at the maximum attainable speed (mentioned by [2] and [9], and similar predetermined value used in our simulations) and it's speed is affected by some environmental condition or passing by vehicle on the flight path. We assume that the UAV changes its speed only at the task points and waypoints and not along the edges. Let u_x denote the speed of the UAV at location x i.e., while leaving the location x. Note that the effective speed of the UAV along an edge may be lesser in presence of bad environmental conditions. We denote the effective speed along an edge e by $effu_e$. Then $effu_{(x,j)} = u_x * (1 - \mu_{(x,j)})$.

Obstacles/No-Fly Zones (NFZ): Exclusion area where the UAVs can not fly. For example the airspace around airports is NFZ for UAVs to prevent any interference with manned aircrafts take off and landing. We have simulated the obstacles with the help of 3D cylindrical shapes. A UAV is not allowed to choose a node which is present inside the 3D shaped cylindrical obstacle (Fig. 1). Let $NFZ = \{NFZ_1, NFZ_2, NFZ_3....NFZ_m\}$ be the set of m no-fly zones known before the mission where NFZ_i is a vertical cylinder specified by its higher altitude, lower altitude, axis and radius. Axis is specified by its x and y coordinates. Thus $NFZ_i = <xa, ya, radius, la, ha>$ for all $i = 1, 2, \ldots, m$ with $xa = x$ coordinate of the axis, $ya = y$ coordinate of the axis, $radius =$ radius of cylinder, $la =$ lower altitude of cylinder and $ha =$ higher altitude of cylinder. A task point is feasible if it does not lie in the no fly zone.

4 Problem Statement

We can formally define our problem via tuple $<g,\ s,\ d,\ r,\ tc,\ ts,\ kc,\ \mu_e,\ QoT,\ NFZ>$ where $g = (v, e)$ is a graph with v as the set of task points and waypoints in $3D$ space. e is the set of edges. s is the source, d is the destination and time constraint kc to reach the destination is the knapsack capacity. NFZ is the set of no-fly zones known before-hand. Each task point is associated with reward r, Quality of task point QoT and time to complete tc. Each way point is associated with the timestamp ts. Each edge has environmental uncertainty value μ_e associated with it, which in turn affects the speed of the UAV. Aim is to find a path from s to d such that it maximizes the reward (by finishing the tasks on its route) and takes no more than kc units of time from s to d. Waypoints define the trajectory of the path and a UAV has to cover the waypoints on the path on or before their respective timestamps to reach. The source and the destination are also considered as waypoints with timestamps 0 and kc respectively. We assume that there is a path from each task point to the destination following the given trajectory. In this paper, we have considered the above problem in an online setting. i.e. the task points, the way points and the edges arrive online. At any point of time, we are exposed to the next waypoint along with its time stamp, a set of task points along with their attributes and a set of edges along with the environmental conditions on its route. The next waypoint is exposed only when we have reached the previous waypoint.

4.1 Proposed Online Dynamic Path Planning Approach

We propose Online Dynamic Path Planning (ODP) approach modelled as an optimization problem using knapsack paradigm given that:

$$\textbf{Maximize: } \sum_{i=1}^{n} r_i \cdot s_i$$

$$\textbf{Subject to: } \sum_{i=1}^{n} t_i \cdot s_i \leq kc$$

$$s_i \in \{0, 1\} \quad \forall i = 1, 2, \ldots, n$$

each task from a set of n task points is associated with time t_i, the sum of time to reach the task point from the current location, and the time to complete the task, and reward r_i. Let s_i be a binary decision variable that indicates whether task i is selected ($s_i = 1$) or not selected ($s_i = 0$). We need to choose a subset of task points that maximize the reward such that knapsack capacity is satisfied. Its a classical knapsack problem where each task is selected at most once.

4.2 ODP Approach

An online greedy multiple-choice approach is used to solve the knapsack problem with the knapsack capacity kc. Suppose we are at location x called the current location at any point of time. The current location x can be represented by the tuple $<T_x, w_x, \mu_e>$ where T_x is the set of task points that are out-neighbours of x; some or all of them may become available online; edges to the vertices in T_x are available online, w_x is the next waypoint to reach from x; this may be same as that at the previous location if x is a task point or may become available online if x is a waypoint. u_x denote the speed of the UAV at location x i.e., while leaving the location x. μ_e denotes one or more bad weather condition or if the weather is clear along an edge e.

We consider the task points in T_x, and extract the set of feasible points from them and decide to visit the task point tp that gives maximum efficiency. A task point is considered feasible if (i) $\theta \leq \tau$, (ii) it does not lie in a no fly zone (iii) UAV can reach the next way point within the time limit.

Efficiency of a feasible point is computed as reward per unit time. However, while computing the reward, we also consider the ease of reaching a task point. A task point that is hard to reach may consume more fuel and hence is less useful for us. Thus, we compute what we call as the effective reward denoted by $effr_{tp}$. Effective reward of a task is the product of the reward and the quality of the task. i.e. $effr_{tp} = tp.r * tp.QoT$; lower value of QoT indicates that it is more difficult to reach the task point thereby reducing the significance of the task point for us. Time is the sum of time-to-reach ($t_{(x,tp)}$) and time-to-complete ($tp.tc$) a task point. Thus, the efficiency of a task point tp is:

$$eff_{tp} = \frac{effr_{tp}}{tp.tc + t_{(x,tp)}}$$

The environmental uncertainties can affect the effective speed of the UAV and hence the time-to-reach a task point. Effective speed, denoted by $effu_{(x,tp)}$ is computed as $effu_{(x,tp)} = u_x * (1 - \mu_{(x,tp)})$ indicating the impact of higher uncertainties on slowing down the UAV. And, $t_{(x,tp)} = \frac{d_{(x,tp)}}{effu_{(x,tp)}}$ where $d_{(x,tp)}$ is the distance to reach from x to tp (calculated as the sum of horizontal and vertical distance, where the vertical distance is calculated as the absolute difference of the altitude of two $3D$ points and the horizontal distance is calculated as the Euclidean distance between them in a horizontal plane.

Whether a task point lies in a no fly zone is checked as follows: If the altitude of the task point is greater than the lower altitude and lesser than the higher

altitude of the cylinder, then we calculate the distance of the task point from the axis. If the distance calculated is less than the radius of the cylinder, then the task point lies in the no fly zone otherwise the point doesn't lies in the no fly zone. For a task point tp, we also estimate the time to reach the next way point via tp. This is the sum of time-to-reach the task tp from the current location, time-to-complete the task tp and time to reach the waypoint from tp i.e. $(t_{(tp,w_x)})$ where $t_{(tp,w_x)} = \frac{d_{(tp,w_x)}}{effu_{(tp,x_x)}}$. We say that the task point is feasible if this estimate is at most the remaining available time.

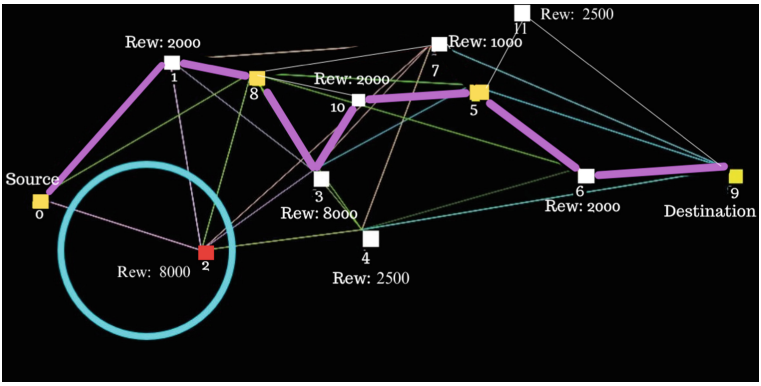


Fig. 2. A Simulated Graph (Color figure online)

Figure 2 depicts a simulation graph using some synthetic data where node numbered 0 is the source and node numbered 9 is the destination. The task points are coloured white, and waypoints are coloured yellow. The no-fly zones is represented by blue coloured circle. The purple line is the flight path taken by UAV to move from source to destination. The path taken by UAV is 0− >1− >8− >3 − >10− >5− >6− >9. We can see that the UAV started from node 0 and since node 2 was in no-fly zone, so it covered node 1 with reward 2000 and reached the waypoint (node 8). Then it has node 3, node 7 and 10 as choices. So UAV deviated in order to fetch a good reward of 8000 instead of covering node 10 (that was nearer) with reward 2000. At node 3, 4 and 10 are the available choices. 4 turns out to be infeasible as the time to reach 5, the next waypoint, via 4 is more. Since 10 turns to be a feasible task point, node 10 is covered before moving to 5. At node 5, 6 and 11 are the available choices. Since for node 11 turning angle is more than τ. So, node 6 is covered before reaching destination.

5 ODP Traversal Algorithms

In order to obtain the flight path in real-time, an online multiple-choice algorithm modeled as Knapsack is developed based on a greedy approach where the path is generated without going back to the previously visited nodes thus eliminating the overhead of repeated task of path planning from a certain set of points.

Algorithm 1 describes our greedy approach where starting from the current location x at source (line 2), it gets the next set T_x of task points (line 6) along with the uncertainties μ_e (line 7) on their way from x and uses Algorithm 2 (line 10) to obtain the most efficient task point (with maximum rewards per unit time) from them. It moves to the most efficient task point (line 12) with the remaining knapsack capacity. We remember the previous node as we move to the next node (lines 3 and 11); this is used to compute the angle of rotation required to reach next location. In case x is a waypoint, we also get the next waypoint w_x in line 9. The process is repeated (line 5) until we reach the destination.

Algorithm 2 finds the most efficient out-neighbour of the current node x considering flight parameters (turning angle at lines 6, 7), no-fly zones (line 8) and environmental constraints. At lines 9 and 10, it computes the impact of environmental uncertainties on the effective speed of the UAV. In addition to the time-to-reach and time-to-complete a task point, we also estimate the time-to-reach the next waypoint from the task point under the impact of environmental uncertainties (lines 13 and 14). If the total time is within the available time (line 15), we compute the efficiency (reward per unit time, line 16) of the task point. Lines 16–20 maintain the efficiency and time-to-reach the most efficient task point. At line 27, we compute the time that will be consumed in reaching the most efficient task point and completing it; this time is subtracted from the knapsack capacity to compute the remaining available time. If no feasible out-neighbour of x is found (line 29), then we go to the next waypoint.

Algorithm 1 : Online Dynamic Path Planner for UAVs
Input : s, d, Knapsack Capacity kc
Output : Path from s to d

1: **procedure** ODP(s, d, kc)
2: $x \leftarrow s$
3: $prev \leftarrow NULL$
4: $resKC \leftarrow kc$
5: **while** $(x! = d)$ **do**
6: getT_x
7: get μ_e for all T_x
8: getu_x
9: If $x \in W getnextw_x$
10: $(maxeffpt, resKC) \leftarrow mep(x, T_x, w_x, \mu_e, u_x, prev, resKC)$
11: $prev \leftarrow x$
12: $x \leftarrow maxeffpt$
13: **end while**
14: **end procedure**

Algorithm 3 calculates the distance between any two 3D points $p1$ and $p2$ as the sum of horizontal distance (calculated as the Euclidean distance between the two points (line 3) and vertical distance (calculated as the absolute difference of the altitude of two points (line 2).

Algorithm 2 : Most Efficient Point(MEP)
Input : current node x, T_x, w_x, μ_e, $prev$, Knapsack Capacity kc
Output : returns the next maximum efficient point

1: **procedure** MEP(x, T_x, w_x, μ_e, u_x, $prev$, kc)
2: $current \leftarrow x$
3: $maxeffpoint \leftarrow null$
4: $maxefficiency \leftarrow 0$
5: **for** tp `in` T_x **do**
6: $\theta \leftarrow angle(prev, current, tp)$
7: **if** $\theta \leq \tau$ **then**
8: **if** $!(inNFZ(tp, NFZ))$ **then**
9: $effu_{(x,tp)} \leftarrow (u_x * (1 - \mu_{(x,tp)}))$
10: $effu_{(tp,w_x)} \leftarrow (u_x * (1 - \mu_{(tp,w_x)}))$
11: $d_{(x,tp)} \leftarrow distance(current, tp)$
12: $t_{(x,tp)} \leftarrow \frac{d_{(x,tp)}}{effu_{(x,tp)}}$
13: $d_{(tp,w_x)} \leftarrow distance(tp, w_x)$
14: $t_{(tp,w_x)} \leftarrow \frac{d_{(tp,w_x)}}{effu_{(tp,w_x)}}$
15: **if** $tp.tc + t_{(x,tp)} + t_{(tp,w_x)} \leq kc$ **then**
16: $effr_{tp} \leftarrow tp.r * tp.QoT$
17: $eff_{tp} \leftarrow \frac{effr_{tp}}{tp.tc + t_{(x,tp)}}$
18: **if** $eff_{tp} > maxefficiency$ **then**
19: $maxeffpoint \leftarrow tp$
20: $maxefficiency \leftarrow eff_{tp}$
21: $t \leftarrow t_{(x,tp)}$
22: $tpc \leftarrow tp.tc$
23: **end if**
24: **end if**
25: **end if**
26: **end if**
27: **end for**
28: $usedCapacity \leftarrow t + tpc$
29: **if** $maxeffpoint == null$ **then**
30: $maxeffpoint \leftarrow w_x$
31: $d_{(x,w_x)} \leftarrow distance(x, maxeffpoint)$
32: $effu_{(x,w_x)} \leftarrow (u_x * (1 - \mu_{(x,w_x)})$
33: $t_{(x,w_x)} \leftarrow \frac{d_{(x,w_x)}}{effu_{(x,w_x)}}$
34: $usedCapacity \leftarrow t_{(x,w_x)}$
35: **end if**
36: $resCapacity \leftarrow kc - usedCapacity$
37: **return** $maxeffpoint, resCapacity$
38: **end procedure**

Algorithm 4 is used to calculate the angle between any three data-points. Vectors ab and bc are obtained by using the coordinates of a, b and c respectively (lines 2 and 3). Magnitudes (lengths) of vectors ab and bc is calculated using the Euclidean distance formula (lines 4 and 5). The inverse cosine ($acos$) function is

calculated using the dot product and magnitude of the vectors and is converted to degrees (line 6).

Algorithm 5 checks if a task point lies in the no-fly zones. If the altitude of the task point is greater than the lower altitude la and lesser than the higher altitude of the cylinder ha (line 6), then we calculate the distance of the task point from the axis (line 12). If the distance calculated is less than the radius of the cylinder (line 13), then the task point lies in the no fly zone otherwise not. The same step is repeated for each no fly zone (line 2).

6 Experimental Set up

This section explains the numerical simulations performed, experimental analysis, results obtained and the performance of the proposed algorithm.

6.1 Numerical Simulations

We have taken a canvas of 1500 * 800 for all the experiments. The coordinates of the points range from 10 to 1400 pixels for the x-axis, 10 to 700 pixels for y-axis and from 3 to 17 for the z-axis. The task points are randomly distributed over the region to simulate the task points covering various cities. We conducted our experimental studies on two types of synthetic data sets:

- 10 data sets, each consisting of 10 randomly generated $3D$ points (first type of data set - used to compare the performance of ODP with brute force).
- 10 data sets, each consisting of 400 randomly generated $3D$ points (second type of data set - to study the effect of parameters under different scenarios).

Each data set is in the form of a graph where the next waypoint and set of tasks appear in the online fashion as explained in the approach i.e. the tasks can be issued anytime. For both types of the data sets, each set of data points, edges and other input parameters are generated randomly from the values as explained next: the values of reward for the task points are in the range from 100 to 1000, for tc the values are taken between 100 and 1000 s, the QoT values, the value of τ and the values of the uncertainty parameters are chosen from the values as discussed in Sect. 3. For the scenarios considering variable speed, we have taken the value of speed (u_x) in the range: $Smin$ is the minimum limit on the UAV speed i.e. 3.6 km/hr and $Smax$ is the maximum limit on the UAV speed. i.e. 18 km/hr. Points with the smallest and the largest x-coordinates are chosen as the source and the destination respectively.

Algorithm 3 : To calculate distance
Input : Two 3D points $p1$ and $p2$
Output : Distance between $p1$ and $p2$

1: **procedure** DISTANCE($p1.x, p1.y, p1.z, p2.x, p2.y, p2.z$)
2: $vd \leftarrow abs(p2.z - p1.z)$
3: $hd \leftarrow \sqrt{(p2.y - p1.y)^2 + (p2.x - p1.x)^2}$
4: **return** $vd + hd$
5: **end procedure**

The ids for task points and waypoints are the unique numbers from 0 to 9 for the first type of data set and from 0 to 399 for the second type of data set.

6.2 Comparison with Offline Optimal

We have used brute force algorithm to obtain a feasible path with the maximum reward points offline. Brute force algorithm finds all the paths from the source to the destination and from there, we select all the feasible paths (those paths which covers all the waypoints including the destination, at the given time and order). From all the feasible paths, we select the best path (i.e., having maximum reward). In case we have one or more such path, one path is selected arbitrary. Thus, brute force approach gives the optimal solution. The path obtained by ODP for all the sixteen scenarios on ten different data sets is compared with the brute force.

Algorithm 4 : To calculate angle
Input : Any three 3D points
Output : Angle between the three 3D points

procedure ANGLE(a, b, c)
  $ab \leftarrow [b.x - a.x, b.y - a.y, b.z - a.z]$
  $bc \leftarrow [c.x - b.x, c.y - b.y, c.z - b.z]$
  $|\mathbf{ab}| \leftarrow sqrt(ab[0] * ab[0] + ab[1] * ab[1] + ab[2] * ab[2])$
  $|\mathbf{bc}| \leftarrow sqrt(bc[0] * bc[0] + bc[1] * bc[1] + bc[2] * bc[2])$
  $\theta \leftarrow \arccos\left(\frac{\mathbf{ab} \cdot \mathbf{bc}}{|\mathbf{ab}||\mathbf{bc}|}\right)$
  $a \leftarrow \theta * 180.0/3.141592653589793$
  return a
end procedure

We have used the following metrics to compare the performance of our approach with the brute force algorithm: competitive ratio, reward per unit traverse time, reward per unit distance traveled, reward per unit computational time and theoretical time complexity. The values reported are average over the 10 data-sets for each scenario.

Competitive Ratio: Let *opt* denote the reward obtained by an optimal offline algorithm or the benchmark solution that has access to complete information about the input sequence in advance. Let *alg* denote the online algorithm that makes decisions without having complete knowledge of the future input. For any input sequence, let $alg(i)$ represent the cost or performance achieved by the online algorithm *alg* on input sequence i, and let $opt(i)$ represent the cost or performance achieved by the optimal offline algorithm *opt* on the same input sequence i. The competitive ratio, $cr(alg)$ of *alg* with respect to *opt* is given as:

$$cr(alg) = sup(\frac{alg(i)}{opt(i)})$$

for all possible input sequences i where *sup* denotes the supremum or the least upper bound. Simply, the competitive ratio measures the performance of the

Algorithm 5 : To find out if a task point lies in no-fly zones
Input : Task point tp, NFZ
Output : returns true if the task point lies in a no-fly zone

```
1: procedure inNFZ(tp, NFZ)
2:     for i in NFZ do
3:         if tp.z ≥ i.la && tp.z ≤ i.ha then
4:             distance ← sqrt((tp.x − i.x)^2 + (tp.y − i.y)^2)
5:             if distance ≤ i.radius then
6:                 return true
7:             end if
8:         end if
9:     end for
10:    return false
11: end procedure
```

online algorithm compared to the optimal offline algorithm over all possible input sequences. It quantifies the ratio of the cost or performance of the online algorithm to the cost or performance of the optimal offline algorithm.

Our Results: Table 1 shows that in more than 55% of the scenarios (i.e., 9 out of 16 scenarios), ODP gives a competitive ratio of more than 0.9 i.e. in more than 55% scenarios our algorithm is able to fetch about 90% of the maximum obtainable rewards. In all the sixteen scenarios, ODP gives a competitive ratio of more than 0.8. i.e. in all the scenarios our algorithm is able to fetch more than 80% of the maximum obtainable rewards. It can be observed that in presence of the restriction of turning angle, the competitive ratio achieved is less than 0.9 whereas in all other cases it is more than 0.9. Thus, our approach handles the environmental uncertainties with a competitive ratio of more than 0.9 in the absence of restriction imposed on the turning angle and with a competitive ratio of more than 0.8 in the presence of restriction imposed on the turning angle.

Reward/Traverse Time: Though brute force gives more reward points than our algorithm, our approach provides faster routes in more than 65% of the cases. Since time is important to save fuel consumption, reward obtained per unit of fuel spent (in terms of per unit of time) is an important parameter to compare any two approaches. In more than 60% of the scenarios reward per unit traverse time is more in our case as compared to that of the brute force as displayed in Fig. 4.

Reward/Traverse Distance: In 50% of the scenarios, the traverse distance of the ODP approach is lesser than that of brute force and the reward per unit traverse distance is comparable in all the scenarios as displayed in Fig. 5 (of 10 data-sets).

Reward/Runtime: ODP always have hundred times less run time than brute force and always have ten times higher reward per unit run time as displayed in Fig. 6 (of 10 data-sets).

Table 1. Comparison of reward - ODP vs. Brute Force

SNo.	Scenario Type	Reward_ODP	Reward_BF	Competitive Ratio
1	No parameters	53855.2	53855.2	1
2	Turning angle (TA)	40673.9	46293.6	0.8786
3	Environmental uncertainties (EU)	53706.8	53855.2	0.9972
4	QoT	53706.8	53855.2	0.9972
5	Variable speed (VS)	53706.8	53855.2	0.9972
6	TA and EU	39412.9	45723.4	0.8619
7	TA and QoT	38714.7	46293.6	0.8362
8	TA and VS	40426.2	46293.6	0.8732
9	EU and QoT	53706.8	53855.2	0.9972
10	EU and VS	50069.0	53855.2	0.9296
11	QoT and VS	53706.8	53855.2	0.9972
12	TA, QoT and VS	39143.9	46293.6	0.8455
13	TA, QoT and EU	40086.6	45723.4	0.8767
14	EU, VS and TA	37938.1	42269.0	0.8975
15	QoT, VS and EU	48869.8	53855.2	0.9074
16	All parameters	38222.7	42269.0	0.9042

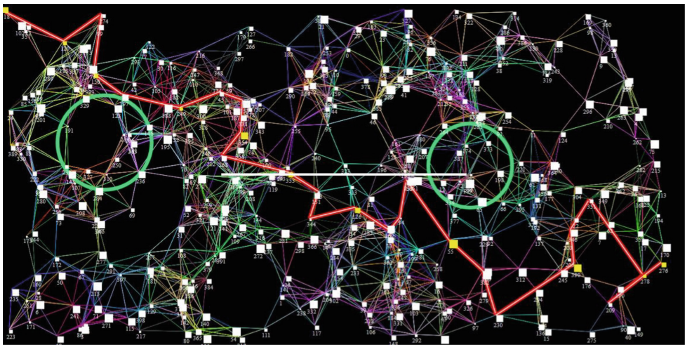


Fig. 3. Flight Path of UAV for 400 Data-points (Red line is flight path), White nodes: task points, yellow nodes: waypoints, Green circles: No Fly Zones (Color figure online)

Theoretical Time Complexity: ODP does constant amount of work along each edge of the graph leading to a complexity of $O(n^2)$ even if the graph is complete whereas as the name suggest, brute force tries out all the paths and selects the one with the maximum rewards leading to exponential time complexity.

Thus, our algorithm is not only dynamic in the sense that it responds to the dynamically changing weather conditions but is also time-efficient obtaining about 80% of the rewards most of the time.

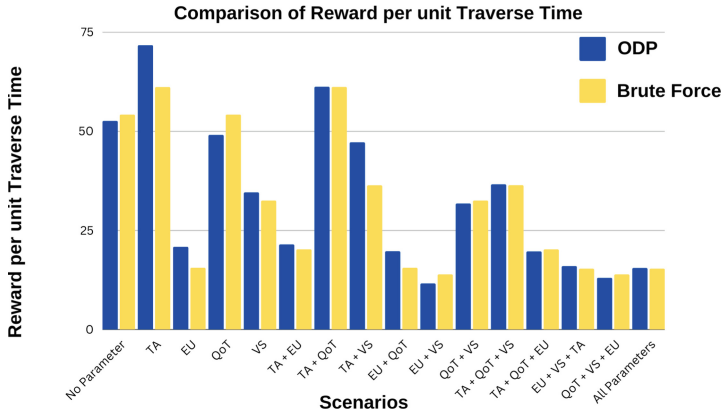


Fig. 4. Reward per unit Traverse Time - Brute Force vs. ODP

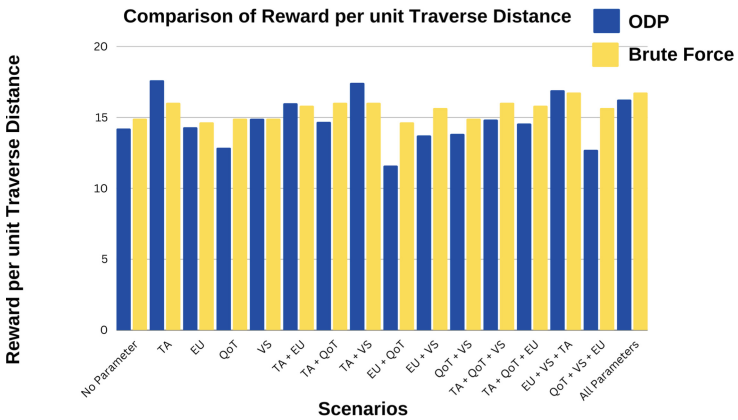


Fig. 5. Reward per unit Traverse Distance - Brute Force vs. ODP

6.3 Experimental Analysis

To study the effect of parameters under different scenarios, we performed simulations on 10 randomly generated data sets each of 400 points (the second type of data-set). Figure 3 shows the simulation for the same. We have used the following metrics for the doing the detailed analysis: reward, reward per unit traverse time, reward per unit distance traveled and reward per unit computational time. The values reported are average over the 10 data-sets for each scenario.

Reward: Figure 7 shows that the total reward earned is getting decreased with the increase in parameters in different scenarios. It can be observed that ODP handles environmental uncertainties with 70.74% of reward as compared to scenario 1 in the absence of restriction imposed on the turning angle and with decrease in the reward by 57.67% in the presence of restriction imposed on the

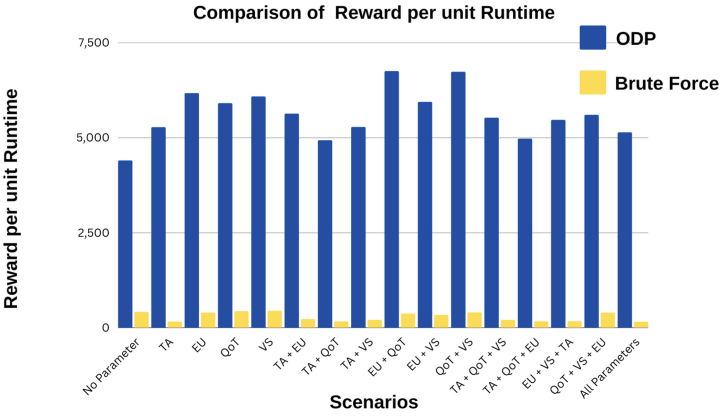


Fig. 6. Reward per unit Runtime - Brute Force vs. ODP

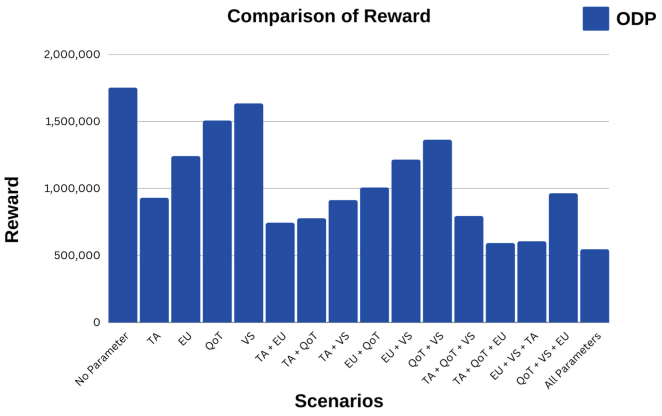


Fig. 7. Reward earned by ODP in different scenarios

turning angle. It can also be observed that when no parameter is considered we got a distance value of 29173.96, a time covered value of 17625.59, and a reward value of 1750181.4, but when all the parameters are considered then the distance covered decreases to 15414.00, time taken decreases to 16917.77 and reward decreases to 543325.4. Without considering the angle, the UAV can choose any point from the space which generates a zig-zag path with sharp turns that are not desired in a real scenario whereas when the angle component is considered, then UAV follows a smooth path directed towards the goal without considering nodes out of coverage area thus decreasing the total distance covered which decreases the total time taken.

Reward/Traverse Time: Figure 8 shows that the scenarios considering environmental uncertainties which decreases the reward per unit traverse time from

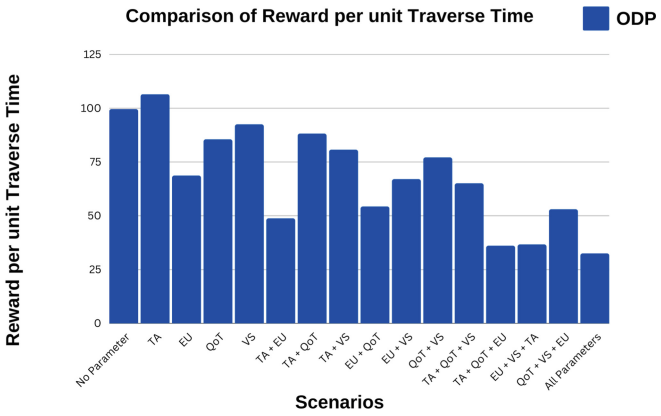


Fig. 8. Reward per unit Traverse Time: ODP

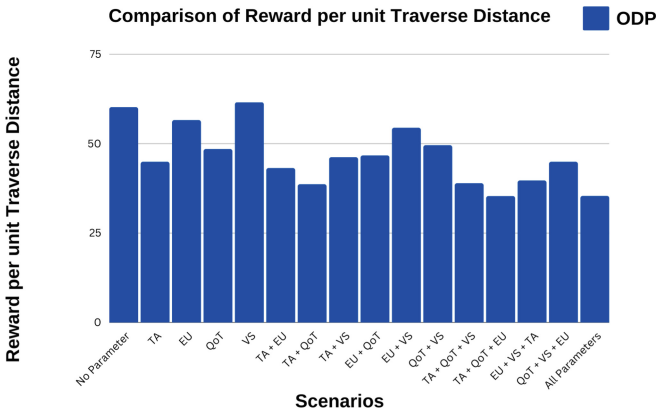


Fig. 9. Reward per unit Traverse Distance: ODP

a range of 32% to 52% as compared to scenario 1. Since environmental uncertainties slows down the speed, traverse time increases.

Reward/Traverse Distance: Figure 9 shows that the scenarios considering turning angle restriction decreases the reward per unit traverse distance from a range of 27% to 37% as compared to scenario 1.

Reward/Runtime: Figure 10 shows that the reward per unit runtime for the scenario considering all the parameters decreases by 45% as compared to scenario 1. Since runtime is higher for the case not considering any parameter in comparison to the case where all the parameters are considered but the reward generated in scenario 1 is higher which in turn results in a greater overall ratio of reward/runtime.

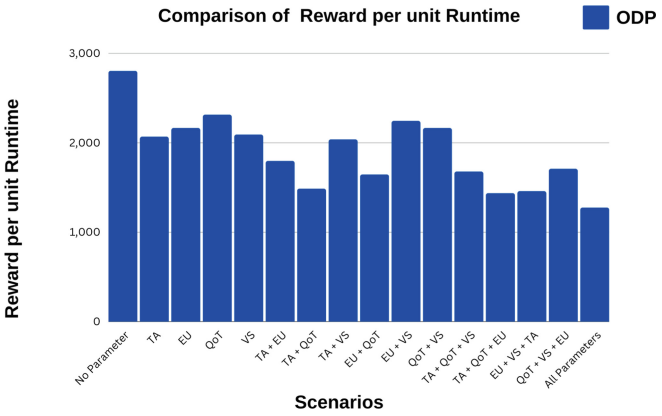


Fig. 10. Reward per unit Runtime: ODP

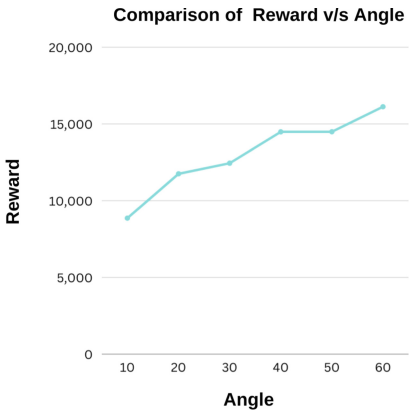


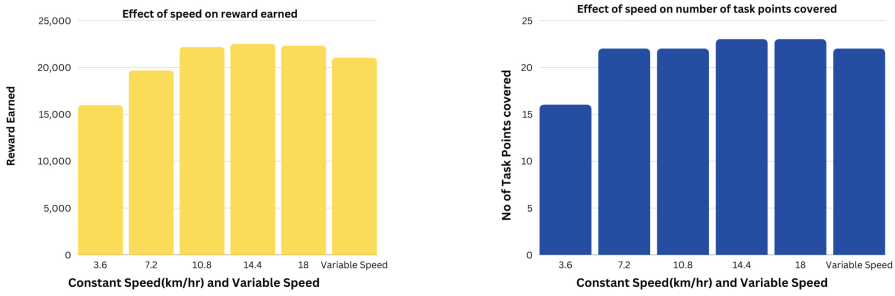
Fig. 11. Line graph for Reward v/s Angle

Turning Angle v/s Reward: In Fig. 11, we evaluate the importance of turning angle value in the process of trajectory generation by analyzing the flight path changes with respect to reward value for the six different angles. As the angle increases, reward earned also increases. This inference can be based on the fact that when a wider angle is provided to the UAV then it is allowed to choose its path from a larger set of task points lead to higher probability of generating rewards.

Speed v/s Reward Earned and Number of Task Points Covered: Figure 12(a and b) shows the implications of constant and variable speed. If we fly the UAV with constant speed throughout the mission (where u_x is not affected by the environmental uncertainties), then the following trend is observed: As the speed increases, there is a relative increase in the total rewards earned and number of task points covered. This can be based on the fact that a UAV with higher

speed is able to cover larger number of points in less time satisfying timestamp constraint. We have also performed ten simulations and recorded the results with variable speed (Scenario 5: where u_x is randomly chosen between S_{min} and S_{max} for every location x and it is not affected by the environmental uncertainties). The algorithm is able to generate the optimum value of reward and cover a good number of task points.

Comparison with A* Algorithm: Since A* algorithm aims for finding the shortest path we cannot provide all the constraints and parameters to A* algorithm as ODP. A* algorithm is not time bound as ODP and thus, it does not work with a prior given trajectory with timestamped task and waypoints.



(a) Speed v/s Reward earned

(b) Speed v/s Task points covered

Fig. 12. Effect of speed on reward earned and number of task point covered

On comparing A* algorithm with the ODP approach like we compared brute force with ODP, the path found by A* (offline algorithm) will not always be feasible (not satisfying the condition of covering waypoints at exact timestamp). Moreover the path returned by A* algorithm would already have been given as one of the path by brute force approach and is already compared with, if that is feasible. If we try to apply A* algorithm between every x and w_x, then the time complexity would be very high, which cannot be used in real world applications. Also, if we consider the scenario where we have 3D task points which are associated with the reward points and we have to find a path from source to destination. The cost function of A* will be $f(n) = (g(n) + h(n))/reward$, where $g(n)$ is the distance from the source to the current node and $h(n)$ is the distance from the current node to the destination. The efficiency measure of ODP works as $efficiency = reward/DistanceToReach$. ODP focuses on generating a flight path in real time without going back to the points previously visited thus eliminating the overhead of repeated task of path planning from a certain set of points whereas A* algorithm re-evaluates the path from the points saved in an open set leading to higher run time and brute force like time complexity in the worst case. Moreover since A* algorithm works with the distance in it's cost function and uses $h(n)$ as the euclidean distance from the current node to

the destination, to compare it with ODP, we need to modify the cost function in terms of time (with varying speed on facing environmental uncertainties) and then make it time-bound, which in turn will modify the algorithm itself.

7 Conclusion and Future Work

This work addresses the problem of generating UAV flight paths that pass through specified waypoints and satisfy spatio-temporal dynamic set of task points with applications in battlefield management and disaster response requiring adaptive flight paths. One of the major differences between our approach and others is that majority of the algorithms are offline (A* type algorithms), and thus do not perform well in real-time as task points are timestamped and are not-predefined. Unlike existing offline algorithms, which are not suitable for real-time applications, ODP incorporates flight parameters and environmental constraints into a single framework, allowing for real-time path planning.

To assess the effectiveness of ODP, it is compared to a brute force offline approach to demonstrate that the online algorithm maximizes the reward within the given knapsack capacity indicating a near-optimal performance close to the optimal offline solution. The online algorithm follows forward exploration and works well in dynamic environment and even in the absence of any specified trajectory where some task need to be performed at waypoint, if necessary. To address the limited travel range of UAVs, instead of minimizing deviation, we set an upper bound on the maximum time for the UAV to travel from the source to the destination to focus on time constraints rather than distances.

For future, we will work on multi-dimensional UAV path planning involving multiple UAVs with applications in a broader range of real-world scenarios.

References

1. Aggarwal, S., Kumar, N.: Path planning techniques for unmanned aerial vehicles: a review, solutions, and challenges. Comput. Commun. **149**, 270–299 (2020). https://doi.org/10.1016/j.comcom.2019.10.014
2. Ažaltovič, V., Škvareková, I., Pecho, P., Kandera, B.: The correctness and reaction time of piloting the unmanned aerial vehicle. Transp. Res. Procedia **51**, 342–348 (2020)
3. Bortoff, S.A.: Path planning for UAVs. In: Proceedings of the 2000 American Control Conference, ACC (IEEE Cat. No. 00CH36334), vol. 1, no. 6, pp. 364–368. IEEE (2000). https://doi.org/10.1109/ACC.2000.878915
4. Cabreira, T.M., Brisolara, L.B., Ferreira Jr, P.R.: Survey on coverage path planning with unmanned aerial vehicles. Drones **3**(1), 4 (2019)
5. Chakrabarty, D., Zhou, Y., Lukose, R.: Online knapsack problems. In: Workshop on Internet and Network Economics (WINE) (2008)
6. Chen, J., Li, M., Yuan, Z., Gu, Q.: An improved A* algorithm for UAV path planning problems. In: 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chongqing, China, pp. 958–962. IEEE (2020)

7. Edelsbrunner, H., Seidel, R.: Voronoi diagrams and arrangements. In: Proceedings of the First Annual Symposium on Computational Geometry – SCG 1985. ACM Press, Baltimore (1985)
8. Hart, P.E., Nilsson, N.J., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE Trans. Syst. Sci. Cybern. **4**(2), 100–107 (1968)
9. Herath MPC Jayaweera and Samer Hanoun: Path planning of unmanned aerial vehicles (UAVs) in windy environments. Drones **6**(5), 101 (2022)
10. Khan, M.T.R., Muhammad Saad, M., Ru, Y., Seo, J., Kim, D.: Aspects of unmanned aerial vehicles path planning: overview and applications. Int. J. Commun Syst **34**(10), e4827 (2021)
11. Luo, J., Wang, Z., Xia, M., Wu, L., Tian, Y., Chen, Y.: Path planning for UAV communication networks: related technologies, solutions, and opportunities. ACM Comput. Surv. **55**(9), 1–37 (2023)
12. Mekala, A.R., Madria, S., Linderman, M.: Aerial vehicle trajectory design for spatio-temporal task aggregation. In 2016 International Conference on Unmanned Aircraft Systems (ICUAS), USA. IEEE (2016)
13. Mohsan, S.A.H., Khan, M.A., Noor, F., Ullah, I., Alsharif, M.H.: Towards the unmanned aerial vehicles (UAVs): a comprehensive review. Drones **6**(6), 147 (2022)
14. Nam, L.H., Huang, L., Li, X.J., Xu, J.F.: An approach for coverage path planning for UAVs. In: 2016 IEEE 14th International Workshop on Advanced Motion Control (AMC), pp. 411–416. IEEE (2016)
15. Padmanabhan, J., Swagath, S.: A study report on solving 0–1 knapsack problem with imprecise data. In: International Conference on Computer Communication and Informatics (ICCCI), pp. 1–5. IEEE (2017)
16. Qu, Y., Pan, Q., Yan, J.: Flight path planning of UAV based on heuristically search and genetic algorithms. In: 31st Annual Conference of IEEE Industrial Electronics Society, IECON 2005, USA, pp. 5–pp. IEEE (2005)
17. Razzaq, S., Xydeas, C., Everett, M.E., Mahmood, A., Alquthami, T.: Three-dimensional UAV routing with deconfliction. IEEE Access **6**, 21536–21551 (2018)
18. Roseman, C.A., Argrow, B.M.: Weather hazard risk quantification for sUAS safety risk management. J. Atmos. Ocean. Technol. **37**(7), 1251–1268 (2020)
19. She, R., Ouyang, Y.: Efficiency of UAV-based last-mile delivery under congestion in low-altitude air. Transp. Res. Part C Emerg. Technol. **122**, 102878 (2021)
20. Tseng, F.H., Liang, T.T., Lee, C.H., Der Chou, L., Chao, H.C.: A star search algorithm for civil UAV path planning with 3G communication. In: 2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, pp. 942–945 (2014). https://doi.org/10.1109/IIH-MSP.2014.236
21. Zhan, W., Wang, W., Chen, N., Wang, C.: Efficient UAV path planning with multiconstraints in a 3D large battlefield environment. Math. Probl. Eng. **2014**, 1–12 (2014)
22. Zhang, C., Zhen, Z., Wang, D., Li, M.: UAV path planning method based on ant colony optimization. In: 2010 Chinese Control and Decision Conference, Xuzhou, China, pp. 3790–3792. IEEE (2010)

Mobile Edge and Fog Computing

Achieving Observability on Fog Computing with the Use of Open-Source Tools

Breno Costa[1,2(✉)], Abhik Banerjee[2], Prem Prakash Jayaraman[2], Leonardo R. Carvalho[1], João Bachiega Jr.[1], and Aleteia Araujo[1]

[1] Department of Computer Science, University of Brasília (UnB), Brasília, DF, Brazil
brenogscosta@gmail.com, aleteia@unb.br
[2] School of Science, Computing and Engineering Technologies, Swinburne University of Technology, Melbourne, VIC, Australia
{abanerjee,pjayaraman}@swin.edu.au

Abstract. Fog computing can provide computational resources and low-latency communication at the network edge. But with it comes uncertainties that must be managed in order to guarantee Service Level Agreements. Service observability can help the environment better deal with uncertainties, delivering relevant and up-to-date information in a timely manner to support decision making. Observability is considered a superset of monitoring since it uses not only performance metrics, but also other instrumentation domains such as logs and traces. However, as Fog Computing is typically characterised by resource-constrained nodes and network uncertainties, increasing observability in fog can be risky due to the additional load injected into a restricted environment. There is no work in the literature that evaluated fog observability. In this paper, we first outline the challenges of achieving observability in a Fog environment, based on which we present a formal definition of fog observability. Subsequently, a real-world Fog Computing testbed running a smart city use case is deployed, and an empirical evaluation of fog observability using open-source tools is presented. The results show that under certain conditions, it is viable to provide observability in a Fog Computing environment using open-source tools, although it is necessary to control the overhead modifying their default configuration according to the application characteristics.

Keywords: Observability · Fog Computing · Edge Computing · Metrics · Logs · Traces

1 Introduction

Fog computing is a computing model that brings computation and data storage closer to where it is needed, typically at the edge of the network. The main characteristics of Fog Computing are its distributed and decentralised nature,

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 319–340, 2024.
https://doi.org/10.1007/978-3-031-63992-0_21

its focus on low latency and high bandwidth communication, and its ability to support a wide range of devices and applications [7]. Fog computing can also leverage existing network infrastructure and resources, such as routers, gateways, and edge devices, to provide a more efficient and cost-effective computing environment [26].

In the context of IoT applications, Fog Computing can play a crucial role in enabling real-time data processing, analytics, and decision making at the edge of the network. This is particularly important for applications that require low-latency and high-bandwidth communication, such as industrial automation, autonomous vehicles, and smart cities [38]. In this way, Fog Computing can enable more intelligent and autonomous IoT systems, reduce network congestion and latency, and improve the overall performance and reliability of IoT applications [38].

IoT applications can have a more fragmented code base compared to traditional applications. This is because they often involve multiple devices and systems that need to work together seamlessly, which can require different programming languages, frameworks, and protocols. In this scenario, there is increased complexity in providing system maintenance and a higher risk of not meeting service level agreements (SLAs) [51]. A possible solution to deal with this scenario is to increase the observability of IoT applications to achieve timely decision-making and fast application troubleshooting [41]. Observability is a concept borrowed from control theory and recently applied to the monitoring and debugging of distributed systems [32]. It refers to the ability to understand the internal state of a system based on its external outputs [30].

Monitoring is based on the collection and analysis of performance metrics from applications and helps to determine whether SLAs are guaranteed [45]. Observability is referenced as a superset of monitoring that uses data analytics techniques on the collected monitoring data with the objective of reducing the time it takes to know why something is not working as it should [36]. In addition to metrics, observability also uses application logs and service call traces, allowing a cross-analysis of these data to gain more insight into system behaviour [32], providing better opportunities for actuation, and shortening the time required to return applications to a healthy state. However, an observability platform must be able to manage and process large amounts of data in a timely manner, and Fog Computing is composed of resource-restricted nodes and interconnected by potentially uncertain networks. This is a critical issue that requires additional research [49]. Because of this there is a need for defining observability from a Fog Computing perspective.

1.1 Motivating Scenario: Smart City

A smart city application is software developed to help manage and improve the quality of life in a city. These applications take advantage of advanced technologies such as sensors, IoT devices, 5G networks, and artificial intelligence to collect and analyse real-time data on different aspects of the city, including public safety, traffic, energy consumption, air quality, and more [21]. Mobile

IoT-RoadBot [18] is a Smart City application that monitors and detects roadside asset maintenance issues. Furthermore, it is used as a real-world testbed to assess 5G performance. It has been deployed on 11 waste collection service trucks (WCST) in Brimbank City Council, located within the metropolitan area of Melbourne, Australia. This testbed provided an ideal real-world deployment scenario to assess the performance of the 5G network due to the natural mobility of the trucks. It allowed the performance of a 5G network to be tested using an application that streams a great amount of real-time video data across a large geographical area.

The 11 WCSTs have been equipped with IoT devices, including depth-sensing stereovision cameras, 5G routers, 5G dome antennas, edge computers, and Global Navigation Satellite Systems (GNSS). Mobile IoT-RoadBot collects footage from the roadside and transmits data continuously while being in service for garbage collection around Brimbank. The application must process the video on the edge to shrink its size and aggregate context data. After this, the data are sent to the cloud to allow server-side functionalities, such as recognition of roadside assets and integration into the maintenance dashboard. WCSTs use 5G technology, but the availability of 5G networks is not ubiquitous in the region. Therefore, sometimes there will be no available connection to send data to the server, and the videos must be stored for further transmission as soon as the network becomes available. Additionally, there are other issues that can cause data flow interruption between WCSTs and servers, such as lack of space to store videos, camera malfunction, and application crash. In such cases, the delay in identifying such problems may cause data loss, money waste, and low-quality service to the citizen.

It would be beneficial for the Smart City IoT application if such issues were identified quickly and automatically. A solution could be delivered faster and the system could return to its normal state, potentially actuating autonomically [17]. For example, after identifying that the application is stuck, i.e. hardware resources and network are available and healthy, but the application is not collecting roadside footage nor transmitting it, it could send a remote command to restart the application and verify after a short period of time whether a different actuation is still necessary. Thus, to provide a comprehensive and updated view of the whole system, it is necessary to increase its observability by promptly collecting the status of each component (sensors, hardware, network, application modules). These data can be processed to provide triggers for actuation, according to pre-defined thresholds and rules. Additionally, the data will be available for historical analysis.

In our previous work [12], we conducted a systematic review of the literature on fog monitoring solutions. We have found that no available proposal was capable of properly managing the collection of metrics, logs, and traces simultaneously in a Fog Computing environment. This work provides a clarification of the challenges of increasing the observability of the application in this scenario. The contributions of this work are: 1. The definition of observability in a Fog Computing environment; 2. Definition of a Fog Observability Data Life Cycle

and 3. An empirical evaluation of delivering observability in a real world Fog Computing environment using a smart city use case.

This work is organised as follows. Section 2 summarises the characteristics of Fog Computing and presents a definition of observability for distributed systems. The most relevant related work is provided in Sect. 3. Section 4 presents a definition of Fog Observability based on a specific data life cycle, and modelling its overhead. A real-world Fog Computing testbed is detailed in Sect. 5. Section 6 describes the experiments and offers a discussion of the evaluation results. Finally, Sect. 7 concludes this work.

2 Background

This section presents relevant information on Fog Computing and Observability.

Fog Computing - Fog computing was presented in 2012 [7] with the objective of providing computing, storage and network services between end devices and cloud providers, complementing resources when it is not possible to meet the requirements with traditional cloud services. In recent years, the concept of Fog Computing has been improved both by academia [13,38,52,53] and industry [25,39]. However, due to the lack of consensus on its definition in terms of scope, composition devices, architecture, service models, etc., there are some other similar paradigms, such as Edge Computing [13], Mobile Edge Computing (MEC) [14], and Mist Computing [43] that are frequently confused with fog. In this work we consider Fog Computing as a broader and more complete concept that can be considered as an umbrella that encompasses all other similar paradigms [10].

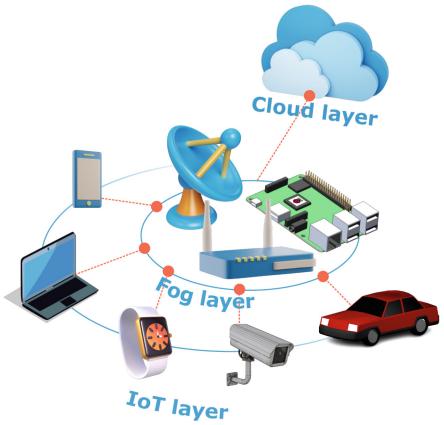


Fig. 1. Overview of the Fog Computing Architecture.

The architecture most used to represent a Fog Computing environment is composed of three layers: IoT layer, Fog Layer, and Cloud Layer, as presented in Fig. 1. The IoT layer represents the IoT devices connected at the edge of

the network by which the end users can request the services to be processed in the above layers. The Fog Layer is placed between the IoT and Cloud Layers and provides shared resources that IoT applications can use as needed, such as processing and data storage resources, before data are transferred to the Cloud [3]. This layer is made up of nodes, commonly called *fog nodes*, i.e. any hardware device that has software and hardware resources with high communication capability [4]. Finally, the Cloud Layer is composed of cloud providers' services, with more robust computational resources to deliver high-order processing and long-term storage.

A Fog Computing environment is characterised by having a more distributed organisation, heterogeneity of physical devices and networks, and connectivity uncertainty, caused by device mobility, network instabilities, and battery exhaustion [26]. This scenario is different from a cloud computing environment, supported by homogeneous resource-rich servers, continuous power supply, and stable redundant network connections.

Observability - Observability is a characteristic of systems that provide information about their internal states by means of external output [30]. The higher the observability, the easier it will be to understand the current and past behaviours of the system and actuate over it when needed. textbfObservability Instrumentation Domains - Observability can be instrumented in a system by generating outputs that inform the internal state of the system at specific points in time. The different data types that compose the output are named Instrumentation Domains. Each instrumentation domain contributes to the observability of a system, offering a different perspective on the system. There is a consensus in the literature that the most important instrumentation domains of observability are metrics, logs, and traces [11,24,32]. Some authors also consider other instrumentation domains such as events [32], profiles [11,24] and crash dumps [11], although they are not recognised as such nowadays by most researchers. As time passes, some of those new domains could be standardised and incorporated into the observability's context. So we can define $ID = \{ID_1, ID_2, ID_3 ... ID_n\}$ as the set of instrumentation domains that are outputed by a system. Having more instrumentation domains available means a higher level of observability. Thus, observability is directly related to the cardinality of ID ($|ID|$). For instance, a Fog monitoring solution that manages only metrics has lower observability than one that manages metrics, logs, and traces simultaneously.

It is feasible to connect the instrumentation domains by the time at which each piece of information was generated. When it is viable to relate two or more of them in the same analysis, more opportunities for actuation arise. In addition to the independent value of each domain, there is an additional value in the cross-analysis between domains, due to their synergistic interactions [37], i.e., when two or more factors act as causes of a particular outcome. This effect is popularly known as "The whole is more than the sum of its parts". So in addition to determining the observability level of a system by the number of its instrumentation domains, we need to also consider the synergistic interactions between them as well. Synergistic interactions could be modelled

as $X(ID) = ID_1\ X\ ID_2 ...\ X\ ID_n$, where X is an operator that filters the data from all available instrumentation domains (ID_i) and returns the subset of each ID_i that matches a specific period of time. Whenever more than one ID_i returns a non-empty subset after applying X, the system has a potentially higher observability level for that period of time. This definition shows that to increase the observability of a system it is important not only to collect information from the instrumentation domains and analyse each data set isolatedly. It is also relevant to be prepared to learn from their interactions and correlate them.

Metrics, Logs, and Traces - There is a consensus in the literature that Metrics, Logs, and Traces are the most important instrumentation domains. This study will focus on them from now on. **Metrics** are more related to the performance of a system. They are numerical values collected at a point in time and their collection can be characterised as a time series. In the motivating scenario, the following metrics are available: percentage of CPU usage, speed of a truck in km/h, throughput of the 5G network in Mbps, amount of video data sent to the Cloud in MB, etc. **Logs** are unstructured or semi-structured text files that report relevant events and contextual information, and the instrumentation is usually done at the development time. Using the motivating scenario, there is information in the logs related to the quality of service of the network connection, the geographic coordinates of the truck, etc. **Traces** are records of service calls made by the system. They allow observations of the call sequence delays, from the beginning to the end of a request. Trace analysis can show which service calls are taking longer in the response time composition of an application. They can also show requests that do not finish correctly. In the motivating scenario, the application performance information (upload throughput) is reported aggregated by suburb to the City Council. This aggregation took a long time to process due to the high volume (2.5 million measurements per week). After optimising the code using the point-in-polygon approach [28] instead of brute force, the time spent on this operation was reduced to 1% of the original time.

Metrics, logs, and traces carry different types of information, as can be seen in Table 1. Each of them contributes to increasing the observability of a system, allowing for complimentary actuation. Metrics deliver objective information about the external interface of a system, e.g., video upload throughput. Logs usually provide internal information about failure events, such as specific error messages, exception handling messages, and runtime errors. This information is

Table 1. The three most important domains of observability differ in their data characteristics.

Domain	Type	Query	Storage
Metric	Numeric	Aggregations	Time Series Database
Log	Semi/not structured strings	Approximate string search	Inverted Index
Trace	DAGs of duration of execution	Disassociated graph search	Inverted Index

necessary to speed up root cause analysis, help the maintenance team improve error treatment, and return the system to a healthy state. Traces provide details about the internal flow of information. These data can be visualised as a graph and a critical path can be generated from it, allowing scrutiny of the dependency among the components of a distributed system [44]. The volume of data depends on the amount of requests and can be bursty.

3 Related Work

This section will present relevant work on metrics, logs, and traces as isolated information domains. After that, some works concerning their integrated management in the context of observability are provided. Several works have already evaluated monitoring solutions made for cloud and prior distributed paradigms. Their results show that none of them was suitable for running in Fog environments [1,2,6,47]. Therefore, the focus of this section will be on proposals that can be applied to Fog environments.

Metrics, Logs, Traces - Metrics, logs, and traces have been used in distributed systems for decades [29] and there are several works that have studied them in isolation. The works [8] and [19] proposed FogMon, a tool based on an agent running on the fog node, collecting metrics about the use of hardware and network resources. It relies on differential updates to reduce overall network overhead. FogMon has the purpose of monitoring performance and collects metrics at a periodic rate. The solution was evaluated in a real testbed for the additional overhead it adds [20]. Although developed for Fog computing, the proposal does not collect logs and traces, providing a low level of observability for the systems it monitors. In [9], the authors suggest a method to connect data from multiple traces by treating all traces as a single graph and breaking down tracing information into multiple vertices and edges. The work uses container as an execution environment and has chosen the same open-source solutions for traces management, although it was evaluated in a cloud native testbed. There is no use of a data life cycle or a definition of observability.

Observability - The work [32] presented an industry experience of a cloud-based team collaboration service with observability data and its production use cases. The author proposed a data management architecture that decouples the real-time and historical data access tiers and the querying tier, with the aim of scaling them independently. OpenTelemetry [40] is a project that seeks to standardise the collection of observability data from cloud-native applications and their infrastructure. The goal of this project is to enable monitoring of the overall health and performance of distributed systems by gathering telemetry data. In [48], techniques to enhance the observability of distributed applications were discussed. This was done by combining multiple data sources to create a comprehensive view of the application's functioning, with the goal of improving decision-making in the orchestration process. In [34], an extended Berkley Packet Filters (eBPF)-based microservices collection system was proposed to provide a

range of metrics for single-tenant environments. The paper [50] introduces an open-source system to control observability data of IoT applications that run on a Kubernetes edge cluster.

Unlike the described proposals, this work is focused on the challenge of achieving a higher level of observability on Fog Computing. To do so, a more formal definition of observability is proposed, as well as a Fog Observability Data Life Cycle (ODLC). Using open source tools, deployed on a fog testbed running a real-world IoT smart city application, experiments were carried out to assess how to implement an ODLC in such an environment with low overhead.

4 Fog Observability

Due to the Fog characteristics described in Sect. 2, the collection, storage, and analysis of a large volume of data in Fog Computing remains an open challenge.

4.1 Fog Observability Data Life Cycle

To obtain valuable information from each instrumentation domain and to increase the observability of an application running in a Fog environment, it is necessary to be aware of the following six-step Observability Data Life Cycle, depicted in Fig. 2: 1. Collection; 2. IoT storage; 3. Transmission of data to the Fog; 4. Fog storage; 5. Data analysis and visualisation; 6. Cloud storage and analysis. The first three Steps make up the Data Collection phase of the life cycle. The last three Steps form the Data Analysis phase.

1. Collection - In the initial Step of the fog observability data life cycle, the data are collected. This can happen in a multitude of ways depending on the instrumentation domain in place. Metrics can be acquired from the operating system by means of system calls. Logs are written according to the specific event flow that was instrumented to be recorded in text. When previously instrumented, traces can be created by specific API calls that record the sequence and delay of each service call. **2. IoT Storage - data staging in the device awaiting transmission** - Observability data are usually immutable and append-heavy [31]. In order to avoid running out of storage resources, a data removal policy should be in place. The period of time that a device can handle stored observability data will depend on several factors, such as data footprint by period of time, the frequency of generation, and the available storage space reserved for the system. Although metrics can be stable in terms of data volume, logs and traces have greater variability [32]. **3. Data transmission to Fog** - Observability may allow timely and proper decision making. Although it is possible to make some minor decisions locally using a single device, critical decisions are expected to be made using a process that can assess a higher volume of data that came from different subcomponents of the system, granting a more comprehensive view of the system. Therefore, the data collected from the IoT layer should be transmitted to the Fog Layer, where a resource-richer node will store them and allow for a

Fig. 2. Fog observability's data life cycle.

more comprehensive data analysis. The network connections used by the application to receive and respond to user requests may be the same as those used by the observability data flow. An adaptive process may be in place to define the amount of data that can be transferred from the devices, selecting which instrumentation domains will be included in each transmission, and the period of time to which the collected data will refer. **4. Fog Storage - Specialised pre-processing and storage according to the type of data and usage** - Fog nodes are expected to be resource richer compared to IoT devices [4]. Due to this, it is on the Fog Layer where observability data from several IoT devices are stored with the aim of rapid actuation and decision-making. The metrics should be stored in a time series database (TSDB). However, logs and traces are structured differently and will benefit from other storage solutions, such as inverted index-based storage, due to the type of queries that are usually made to retrieve meaningful information from them [32]. Therefore, an observability data ingestion service on the fog should consider the data requirements that each instrumentation domain needs (see Table 1), while allowing cross-analysis to be performed. **5. Data analysis and visualisation for decision making** - Once the observability data are available on the Fog, it is possible to query them and make decisions and actuations accordingly. Observability data tend to give more relevant answers when they are queried as soon as they arrive, which means that most queries and analysis use more recent data (less than 24 h) [31]. Thus, it is important to guarantee fast access to this time window data. In addition to that, to save resources to continue receiving IoT data, it is important to provide automated mechanisms to send the data out of this range to long-term storage in the Cloud. **6. Cloud Storage - Long-term storage and historical analysis** - Cloud is the appropriate environment to store large data volumes and run heavy data processing models, such as historical analysis of observability data [23].

4.2 Fog Observability Overhead

Observability is not an end in itself. The positive outcome that an increased observability level can deliver must be balanced with the amount of computational resource needed (the overhead) to reach that level. Although obvious in a generic sense, this statement gains more relevance in a Fog Computing environment due to the characteristics of resource restriction and network uncertainty. This outcome can be modelled as in Eq. 1:

$$Outcome = \frac{Observability}{Overhead} \tag{1}$$

where *Overhead* is a non-zero number between 0 and 100, representing the percentage of resources (i.e. CPU, memory, network bandwidth) consumed when running the life cycle described in Sect. 4.1. The higher the *overhead*, the lower could be the *Outcome*, considering that resource restriction is a fog characteristic.

The observability of a system is directly related to the cardinality of ID ($|ID|$) in addition to the synergistic interactions $X(ID)$, as seen in Sect. 2. Having $ID = \{ID_{Met}, ID_{Log}, ID_{Tra}\}$ means that the ODLC is capable of managing metrics, logs, and traces throughout the system. Due to connectivity uncertainty and resource restriction of Fog environments, part of the observability data set might be lost when there are no available resources such as storage, network bandwidth, device battery, etc. To cope with this, it is useful to define weights (W_i) for each instrumentation domain, representing the importance of that domain compared to the others in a given moment. As a data property, it is important to ensure that the overall sum of required weights is equal to 100%, so $\sum W_i = 1$. These weights could be used to make adaptive decisions on which domain data will be collected or transmitted in the event that resources run out or in other specific situations. For example, a weight of 0% can mean that data from that domain should not be managed. Values above 0% will determine the rank of priorities if necessary. The weights may vary according to the needs, such as being in a regular system operation or after a massive error has occurred.

Using the observability definition of Sect. 2 in Eq. 1, but replacing ID by the weight set $W = \{W_{Met}, W_{Log}, W_{Tra}\}$. Splitting the resultant equation to isolate each instrumentation domain with its intrinsic overhead, and also isolating cross-analysis among the domains, the result is as follows:

$$Outcome = \frac{W_{Met}}{Over_{Met}} + \frac{W_{Log}}{Over_{Log}} + \frac{W_{Tra}}{Over_{Tra}} + \frac{X(ID)}{Over_{X(ID)}} \tag{2}$$

When running the observability data life cycle, the overhead of each instrumentation domain, namely, $Over_{Met}$, $Over_{Log}$ and $Over_{Tra}$ for metrics, logs, and traces, respectively, can be modelled as a function of the amount of resources consumed to generate, store, and transmit the information for that domain. A suggested model is to assign the maximum percentage of resource consumption among key hardware resources, such as CPU, memory, and network bandwidth, as in $Over_{Met} = \max\{\%CPU_{Met}, \%Mem_{Met}, \%Net_{Met}\}$ * 100. For simplicity, we consider the first 3 factors of Eq. 2 together as representing the data collection phase. The overhead of the synergistic interactions $Over_{X(ID)}$ can be modelled in the same way, but with respect to resources consumed to store and analyse the data in the fog, and transmit them to the Cloud (Steps 4 to 6 of ODLC).

$$Outcome = \overbrace{\frac{W_{Met}}{Over_{Met}} + \frac{W_{Log}}{Over_{Log}} + \frac{W_{Tra}}{Over_{Tra}}}^{\text{DATA COLLECTION PHASE}} + \overbrace{\frac{X(ID)}{Over_{X(ID)}}}^{\text{DATA ANALYSIS PHASE}} \tag{3}$$

In a dynamic Fog Computing environment, with resource restriction and network uncertainty, decision making can be more effective using a formula such as Eq. 2 that balances the relative importance of each instrumentation domain with the effort required to collect and analyse it.

5 Experimental Testbed

This section presents the infrastructure used to build the testbed and carry out the experiments.

Hardware - The testbed consists of four Raspberry Pi 4 units (IoT devices), two more resourceful devices (fog nodes) and one virtual machine running in the Cloud, as described in Table 2. Raspberry Pi 4 has a configuration of 4 GB of RAM and runs Ubuntu 22.04 LTS. Each Raspberry Pi 4 unit replays the data generated by a real waste collection service truck when it was in service in 2022 on the streets of Brimbank, Australia, running the Mobile-IoT-RoadBot [18] application. These devices and the connections that link them to the Fog Layer will run the data collection phase of the ODLC described in Sect. 4.1.

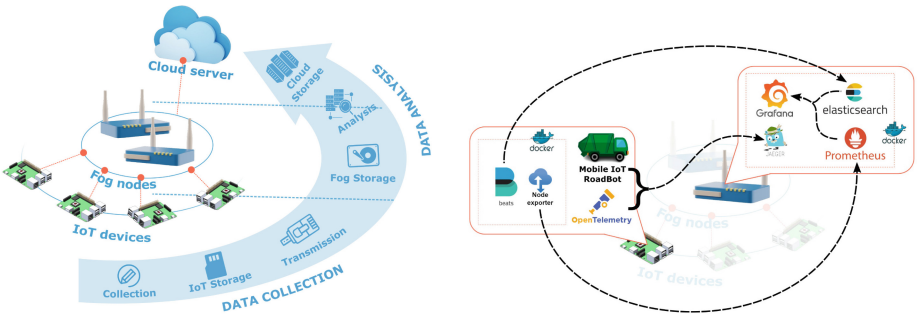
Table 2. Hardware configuration for experimental testbed.

Device name	Configuration	Quantity	Node type
Raspberry PI 4B model	Quad-core Cortex A72, 4 GB RAM	4	IoT device
Fog server	Quad-core 16 GB RAM	2	Fog node
Virtual machine	8-core 32 GB RAM	1	Cloud server

The next components of the testbed are the fog nodes, devices with 4 CPU and 16 GB RAM, which hosts the open source tools used to store and analyse observability data. Metrics, logs, and traces come from IoT devices and are stored in the fog node. This data aggregation allows for a comprehensive view of the status of each component of this distributed smart city application. Decision making can be quick after applying simple rules on updated data, creating alerts, updating dashboards, and performing cross-analysis among domains. This is where most of the Data Analysis phase takes place.

The google cloud e2-standard-2 instance comprises 8 VCPUs, 32 GB RAM, and 200 GB hard disk. The virtual machine on the Google Cloud hosts the long-term storage of observability data. Observability data are append-heavy and tend to grow indefinitely. To control the resources needed to provide fast decision-making in the Fog Layer, the volume of data should be limited. In this experiment, a week was defined as the age limit; however, this can easily be

adapted to each application. Thus, every day, the data that exceed this limit are automatically moved to the Cloud. In the Cloud, a more complex and heavy analysis can be performed on historical data. Figure 3a) shows the testbed as a 3-layer Fog Computing architecture, relating each ODLC Step to the devices where it usually occurs.



(a) Testbed architecture.

(b) Observability open source tools data flow.

Fig. 3. Fog Experimental Testbed.

Software - To our knowledge, there is no available solution that allows the collection of metrics, logs, and traces simultaneously in a Fog Computing environment [12]. Therefore, a solution must consist of several different tools, increasing the complexity of management. The most referenced open-source tools have been selected to characterise the challenges of increasing the observability level in a Fog Computing environment. They are described in the following paragraphs, and a summary of them is described in Table 3. Every selected tool was deployed on the devices using Docker containers. The device that hosts each piece of software and its data flow can be seen in Fig. 3b. The virtual machine on the Cloud mimics the environment of the fog node and is not shown in the figure for simplicity. The data flow from the Fog Layer to the Cloud Layer is based on export and import operations, respectively.

Metrics Management - Prometheus [42] is the open-source standard for the collection and storage of metrics. It can be configured to pull the metrics from the IoT devices in a defined frequency (each 5 s by default), and all exposed metrics are scraped and stored in a TSDB. Prometheus was deployed in the fog node. Grafana [33] is a lightweight dashboard software that is used to visualise the metrics stored in Prometheus. In this experiment, it will also connect to ElasticSearch to show log information. Prometheus relies on external agents that run on IoT devices to generate metrics data and make them available for collection. In this work, Node Exporter is used. It is an agent that exposes hardware and network metrics such as the percentage of free CPU and the throughput of the network connection.

Table 3. Selected open source tools deployed on the testbed.

Tool name	Domain	IoT device	Fog node	Cloud server	Observation
Node Exporter	Metrics	✓			Collects metrics from the device and exposes them via an HTTP call
Filebeat	Logs	✓			Monitors specific files and transmits their content to Elastic Search
OpenTelemetry	Traces	✓			Programming language libraries to create and transmit traces to Jaeger
Prometheus	Metrics		✓	✓	Pull metrics from the Node Exporter on each IoT device
Elastic Search	Logs		✓	✓	Stores logs transmitted by Filebeat and traces data received by Jaeger
Jaeger	Traces		✓	✓	Receives trace data transmitted by Open Telemetry SDK
Grafana			✓	✓	Presents dashboards of data from Prometheus, Elastic Search and Jaeger

Logs Management - ElasticSearch [15] is an open source database of inverted index. It is used for indexing, storage, and fast retrieval of logs and traces data sent to it. Filebeat [16] is an agent that runs locally on the IoT device, sending log data to ElasticSearch.

Traces Management - Jaeger [27] was selected as an open source trace management tool. It can be configured to use ElasticSearch as storage and provides proper visualisation and fast query response time. To collect, store, and transmit traces to Jaeger, the application should be instrumented with the calls that create the traces. In this experiment, Open Telemetry SDK [40], open source libraries available in the most used programming languages, will be used to instrument a report functionality of Mobile IoT-RoadBot.

6 Evaluation

This section presents the experimental evaluation of open source observability tools using a real-world fog testbed running an IoT smart city application. The overhead of the software components and the data volume are measured. Additionally, findings on the benefits and challenges of deploying an observability framework in a Fog environment are presented and discussed. This evaluation demonstrates that an ODLC can be deployed in a Fog environment using open-source tools and shows the challenges and design decisions that should be in perspective.

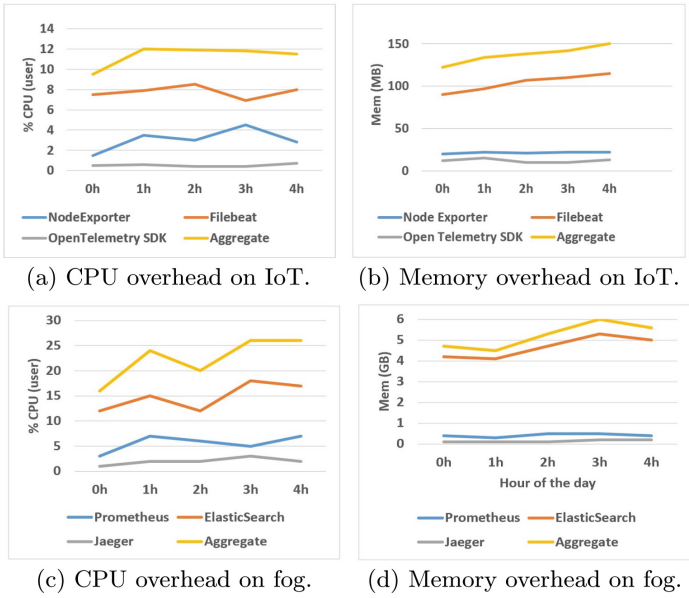
6.1 Methodology

To assess the overhead of achieving a higher level of observability in a Fog environment, the testbed was used to measure: 1. The overhead of each tool in the

IoT devices; 2. The overhead of the server component of each tool running on a fog node; 3. Volume of observability data collected, stored, and transmitted through the data life cycle. While Mobile IoT-RoadBot was running on a Raspberry Pi, we deployed each agent on the same device, and measured the overhead in terms of CPU and memory usage for 4 h. We use the System Activity Reporter (SAR) [46] to get the average CPU and memory usage each 5 min. After recording this information, we uninstalled the tool and repeated the process with the other tools (Node Exporter, Filebeat, and OpenTelemetry SDK, described in Sect. 5). We performed a similar procedure related to the server side of the tools (Prometheus, ElasticSearch, and Jaeger, described in Sect. 5). After measuring the overhead, we run Mobile IoT-RoadBot for 9 h (a regular whole-day journey for each truck) to assess the volume of data collected by each instrumentation domain and transmitted through the layers of the testbed architecture. Finally, to assess the benefits of achieving a higher level of observability on Fog environments, we replayed by 9 h on the testbed the real-world output of 4 WCSTs. Each real-world WCST output is replayed by a Raspberry Pi that also runs observability agents sending data through ODLC from the IoT Devices to the Cloud server.

6.2 Overhead

Figure 4 shows the CPU and memory usage of each component of the open source observability tool set being evaluated in this work. Figures 4a and 4b show the overhead of the IoT devices. We observe negligible CPU and memory overhead. The aggregated amount of CPU when all three components, i.e. NodeExporter, Filebeat, and OpenTelemetry SDK, run simultaneously is under 12% on average. In the case of memory usage, an aggregated footprint of less than 150MiB of RAM is required. Figures 4c and 4d show the overhead on the Fog node. Unlike the IoT devices, CPU usage is higher. This is expected since the Fog node deals with 4 times the volume of data (sent by the 4 IoT devices), with the purpose of receiving, processing and storage of data. A CPU usage average of less than 25% for this kind of processing seems to be worth it. However, it prevents the server side of the observability tool set from being on the IoT layer, where devices have less resources. To guarantee steady performance, the volume of stored data was limited to a week. In terms of memory usage on fog nodes, Prometheus allocated 400 MiB of RAM on average, while Jaeger allocated around 200MiB, a very low memory footprint for the load of Mobile IoT-RoadBot observability data. However, ElasticSearch allocated almost 4.5 GB of RAM. Mobile IoT-RoadBot has a steady data collection flow and usually does not generate peaks of transmitted data. However, when dealing with a more data-intensive application or an application that has a bursty behaviour, the server side overhead should be monitored to guarantee that it copes with the needed load.



(a) CPU overhead on IoT. (b) Memory overhead on IoT.

(c) CPU overhead on fog. (d) Memory overhead on fog.

Fig. 4. Assessing overhead of observability tools on: IoT devices (a) and (b); Fog Nodes (c) and (d).

6.3 Cross Analysis of Metrics, Logs and Traces on the Fog

In the last subsection, the overhead added to the Fog infrastructure after deploying open source observability tools that implement an ODLC (Sect. 4.1) was detailed. This subsection will show the benefits that Mobile IoT-RoadBot could have if it was using such a set of observability tools. Figure 5 shows how the observability data can be analysed on the Fog using the open source framework deployed on the testbed. Figure 5a shows the value of some metrics (CPU, memory and bandwidth usage) collected by the experiments in the last hour from IoT devices. If any metric is outside the range considered safe, an alert message could be sent to the maintenance crew, allowing quick action. Figure 5b shows the visualisation of logs collected from IoT devices each 1 s. The logs can be easily preprocessed while being ingested to identify different fields of information, providing fast queries and allowing for alert management. Figure 5b shows the result of a query made on log data, detailing the latency between the IoT device and some servers of interest on the Internet. Figure 5c shows a dependency graph, created from the collected traces, where one can see the average delay of each dependent component after hundreds of requests and identify which cause the majority of the response time. This information is relevant for planning future performance improvements. Finally, Fig. 5d shows the details of a specific trace, where it is possible to identify the components that cause the longest response time. When looking for the root cause of an identified issue, this information is very useful.

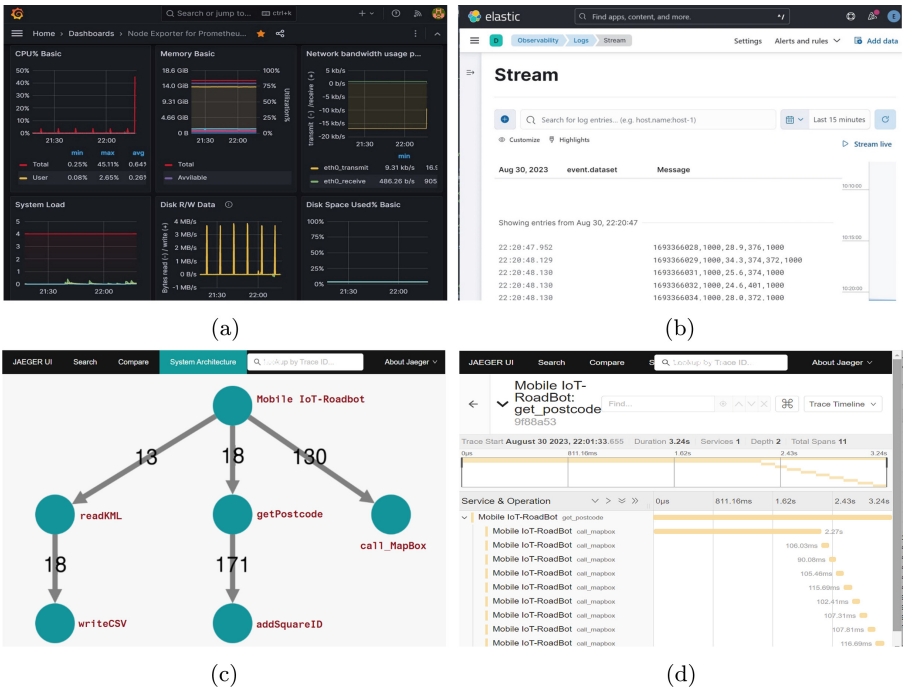


Fig. 5. Observability data visualisation on Fog Layer: Metrics (a), Logs (b), and Traces (c) and (d).

6.4 Discussion

Table 4 presents the volume of observability data that were managed by the data life cycle during the experiments. NodeExporter was deployed with the default configuration. Although it is a tool with a small footprint in terms of CPU and memory usage [22], it may have a not negligible impact in terms of the volume of data it collects. The default set of metrics that it exposes accounts for 65KB of information. These data are presented only when Prometheus pulls them using an HTTP call. This means that there is no IoT storage for these data. Prometheus is configured by default to scrape the NodeExporter page every 5 s, getting all the metrics exposed and storing them in its TSDB on the Fog node. Considering that there are four IoT devices exposing metrics, the data volume transmitted to and stored on the Fog node is about 8.75 GB in the span of a week, the period when these data will be available for decision-making and other analysis on the Fog Layer. After reaching a week of age, the information is removed from the fog node and sent to the Cloud for long-term storage and historical analysis. As a matter of estimation, the volume on the Cloud will reach 75 GB after 2 months of operation. The default output from NodeExporter provides help text for each metric, as shown in Fig. 6. This information accounts for at least 20% of the total output footprint and should be removed prior to

Table 4. Mobile IoT-Roadbot assessment of each observability domain .

Tool	Domain	Data Collection	Frequency	Volume by Hour	IoT Storage	Fog Storage	Fog Volume (1 week)	Cloud Storage	Cloud Vol. (2 months)
Node Exporter	Metrics	65KB	each 5 s	46 MB	No	Yes	8.75 GB	Yes	75 GB
Filebeat	Logs	1KB	each 1 s	3.50 MB	Yes	Yes	0.67 GB	Yes	5.77 GB
Open Telemetry	Traces	4KB	each 15 s	1 MB	No	Yes	0.2 GB	Yes	1.54 GB

exposing the metrics. The default set of metrics is very extensive and probably not all metrics are useful for every use case. For example, the node exporter exposes dozens of Go environment metrics (Fig. 6) that are not of interest for the monitoring of Mobile IoT- RoadBot and should be removed. In addition to cutting off metrics that are not of interest, machine learning over historical data can be used to figure out metric correlations and keep the target metric set at minimum [5]. Furthermore, the frequency of scraping can be decreased in the Prometheus configuration without a relevant loss of opportunities for actuation. Increasing the scrap delay to 10 s will reduce the data volume transmitted to and stored on the Fog Layer by half. Using the strategies of removing the help text and changing the configuration of Node Exporter to expose only metrics about CPU, memory, disk, network, and power supply, and increasing the scrap delay to 10 s on Prometheus, we could reduce the volume of metric data on the Fog node by 87%, which also positively affected CPU and memory usage by Prometheus. Regarding the logs generated by Mobile IoT-Roadbot while the trucks were moving around the city, they record information about 5G network analysis, such as latency and throughput, and contextual information (GNSS coordinates, truck speed, etc.). Although the application writes information in the logs every second, the volume of data written is low (0.67 GB, in the span of a week, as seen in Table 4), being smaller than the volume of data generated by Node Exporter after applying volume reduction strategies. Filebeat was configured to harvest only the logs written by Mobile IoT-RoadBot and transmit them to the Fog Layer. It was necessary to change the default configuration of Filebeat to turn off the auto-discovery feature. When this feature is active, Filebeat receives from the Docker manager every status change of any container on the device, consuming more memory than necessary. As the application was not originally instrumented to record trace calls, we instrumented it in a reporting feature, utilised to aggregate 5G data by the suburbs of Brimbank. To make this aggregation, geographic coordinates were used to find the full address of Australia using a service called MapBox [35]. Using the data footprint of these traces, we estimate the data volume to generate the traces of regular operation of the Mobile IoT-RoadBot. This use case does not have bursty behaviour in terms of request processing because it performs the same volume of operations while in service. Therefore, the volume of trace data is steady.

The aggregated data volume, collected by the four IoT devices, transmitted and stored on the fog node for a period of one week was approximately 10 GB, considering the default configuration of the open source tools used as shown in

```
# TYPE go_memstats_frees_total counter
go_memstats_frees_total 697
# HELP go_memstats_gc_sys_bytes Number of bytes used for garbage collection system metadata.
# TYPE go_memstats_gc_sys_bytes gauge
go_memstats_gc_sys_bytes 2.991584e+06
# HELP go_memstats_heap_alloc_bytes Number of heap bytes allocated and still in use.
# TYPE go_memstats_heap_alloc_bytes gauge
```

Fig. 6. Snippet of default output of NodeExporter.

Table 4. Using the strategies described above, the volume of aggregated data was reduced to 2 GB, a reduction of 80%. The four trucks whose observability data are replayed by the IoT devices in this experiment transmitted 291 GB of video data using the 5G network in a week of real world operation. Therefore, the observability data (2 GB) would represent an overhead of less than 1% in this use case. The experiments show that it is possible to collect the benefits of achieving a higher level of observability for a system in a Fog computing environment. In addition, the overhead of deploying an observability data life cycle can be low, if properly managed. The utilisation of Docker containers as the runtime environment for the observability tools help to address the Fog challenge of device heterogeneity. Due to the resource restriction of IoT devices, observability data collection should be done by lightweight agents. In addition, the data footprint should be minimised to reduce the risk of network congestion and increased overhead to collect and transmit the data to the Fog Layer. Each instrumentation domain has specific data requirements (Table 1) that must be met to optimise storage and minimise the average delay in analysing the observability data in the Fog Layer for decision-making and actuation. Leaving in the Fog Layer only a window of most recent observability data is another strategy to cope with the resource-restriction of fog nodes. Data that are outside the age range are sent to the Cloud for long-term storage and historical analysis.

The open source tools selected to make up the experimental setup are managed independently. This scenario makes more complex actuation difficult to implement. For instance, in the dynamic environment of Fog Computing, a system may present errors running on specific devices while it is functioning properly on others. In such cases, if there are not enough resources to transmit all observability data to the Fog Layer, a proper decision should be prioritising data from those specific devices and returning to regular operation when the issue is solved. To implement such adaptive and autonomous behaviour, it might be necessary to orchestrate the observability data life cycle and its agents. To our knowledge, there is no Fog solution in the literature that provides this functionality [12].

7 Conclusion and Future Work

This work defined the characteristics and challenges of increasing the observability of systems in a Fog Computing environment. The higher the observability,

the more likely it is to understand the root cause of issues in the runtime environment and solve those issues more quickly, helping to guarantee that SLAs are met. A definition of observability aligned with the literature and considering Fog challenges is proposed. Observability can be instrumented using metrics, logs, and traces. Each of these observability instrumentation domains has specific characteristics with respect to the type of data, the volume, and the frequency of collection. Specific mechanisms are needed to collect this information simultaneously and store it centrally, implementing a data life cycle that allows rapid decision-making, even autonomously, and that meets the requirements imposed by the Fog environment. An IoT Smart City use case was deployed in a real Fog Computing testbed, configured with open source observability tools, to characterise the overhead that these tools add to the infrastructure and to understand the challenges that must be faced to increase the observability of a system in this scenario. The results show that although none of the selected tools was developed specifically for Fog computing, it was possible to collect the benefits of increased observability with a minimum overhead, after using strategies to reduce data volume while maintaining opportunities for actuation. Future work may propose appropriate support to overcome the identified challenges.

Acknowledgements. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001, in the context of the Capes PrInt programme. This work is based upon work supported by the Google Cloud Research Credits programme with the award "sabreenar 216815189".

References

1. Abderrahim, M., Ouzzif, M., Guillouard, K., Francois, J., Lebre, A.: A holistic monitoring service for fog/edge infrastructures: a foresight study. In: 2017 IEEE 5th International Conference on Future Internet of Things and Cloud (FiCloud), pp. 337–344. IEEE (2017)
2. Abreha, H.G., Bernardos, C.J., Oliva, A.D.L., Cominardi, L., Azcorra, A.: Monitoring in fog computing: state-of-the-art and research challenges. Int. J. Ad Hoc Ubiquit. Comp. **36**(2), 114–130 (2021)
3. Al-Doghman, F., Chaczko, Z., Ajayan, A.R., Klempous, R.: A review on fog computing technology. In: 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 001525–001530 (2016). https://doi.org/10.1109/SMC.2016.7844455
4. Bachiega, J., da Costa, B.G.S., Araujo, A.P.F.: Computational perspective of the fog node. 22nd International Conference on Internet Computing & IoT (2021)
5. Bali, A., Gherbi, A.: Rule based lightweight approach for resources monitoring on IoT edge devices. In: Proceedings of the 5th International Workshop on Container Technologies and Container Clouds, pp. 43–48 (2019)
6. Battula, S.K., Garg, S., Montgomery, J., Kang, B.: An efficient resource monitoring service for fog computing environments. IEEE Trans. Serv. Comput. **13**(4), 709–722 (2019)
7. Bonomi, F., Milito, R., Zhu, J., Addepalli, S.: Fog computing and its role in the internet of things. In: Proceedings of the First Edition of the MCC Workshop

on Mobile Cloud Computing. MCC 2012, pp. 13–16. ACM, New York, NY, USA (2012). https://doi.org/10.1145/2342509.2342513
8. Brogi, A., Forti, S., Gaglianese, M.: Measuring the fog, gently. In: Yangui, S., Bouassida Rodriguez, I., Drira, K., Tari, Z. (eds.) ICSOC 2019. LNCS, vol. 11895, pp. 523–538. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-33702-5_40
9. Cassé, C., Berthou, P., Owezarski, P., Josset, S.: Using distributed tracing to identify inefficient resources composition in cloud applications. In: 2021 IEEE 10th International Conference on Cloud Networking (CloudNet), pp. 40–47
10. Chiang, M., Ha, S., Risso, F., Zhang, T., Chih-Lin, I.: Clarifying fog computing and networking: 10 questions and answers. IEEE Commun. Mag. **55**(4), 18–20 (2017)
11. CNCF - Cloud Native Computing Foundation: Observability whitepaper. (2023). https://github.com/cncf/tag-observability/blob/whitepaper-v1.0.0/whitepaper.md. Accessed Oct 2023
12. Costa, B., Bachiega Jr, J., Carvalho, L.R., Rosa, M., Araujo, A.: Monitoring fog computing: a review, taxonomy and open challenges. Comput. Netw. 109189 (2022)
13. Dastjerdi, A.V., Gupta, H., Calheiros, R.N., Ghosh, S.K., Buyya, R.: Fog computing: principles, architectures, and applications. Internet Things: Principles Paradigms 61–75 (2016)
14. Dolui, K., Datta, S.K.: Comparison of edge computing implementations: fog computing, cloudlet and mobile edge computing. In: GIoTS 2017 - Global Internet of Things Summit, Proceedings (2017)
15. Elastic: The ELK Stack (2023). https://grafana.com/. Accessed July 2023
16. Filebeat: Lightweight shipper for logs. (2023). https://www.elastic.co/guide/en/beats/filebeat/current/filebeat-overview.html. Accessed July 2023
17. Fizza, K., et al.: A survey on evaluating the quality of autonomic internet of things applications. IEEE Commun. Surv. Tutorials (2022)
18. Forkan, A.R.M., et al.: Mobile IoT-RoadBot: an AI-Powered mobile IoT solution for real-time roadside asset management. In: Proceedings of MobiCom 2022, pp. 883–885. MobiCom 2022, ACM (2022)
19. Forti, S., Gaglianese, M., Brogi, A.: Lightweight self-organising distributed monitoring of fog infrastructures. Futur. Gener. Comput. Syst. **114**, 605–618 (2021)
20. Gaglianese, M., Forti, S., Paganelli, F., Brogi, A.: Lightweight self-adaptive cloud-IoT monitoring across fed4fire+ testbeds. In: 2022-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 1–6
21. Gharaibeh, A., et al.: Smart cities: a survey on data management, security, and enabling technologies. IEEE Commun. Surv. Tutorials **19**(4), 2456–2501 (2017)
22. Grossmann, M., Schenk, C.: A comparison of monitoring approaches for virtualized services at the network edge. In: 2018 International Conference on Internet of Things, Embedded Systems and Communications (IINTEC), pp. 85–90
23. Hashem, I.A.T., Yaqoob, I., Anuar, N.B., Mokhtar, S., Gani, A., Khan, S.U.: The rise of "big data" on cloud computing: review and open research issues. Inf. Syst. **47**, 98–115 (2015)
24. Hausenblas, M.: Cloud Observability in Action. Manning (2023)
25. Iorga, M., Feldman, L., Barton, R., Martin, M., Goren, N., Mahmoudi, C.: The NIST definition of fog computing. Technical report National Institute of Standards and Technology (2018)
26. Iorga, M., Feldman, L., Barton, R., Martin, M.J., Goren, N.S., Mahmoudi, C.: Fog computing conceptual model (2018)
27. Jaeger: Open source, end-to-end distributed tracing. (2023). https://www.jaegertracing.io/. Accessed July 2023

28. Jordahl, K.: Geopandas documentation (2016). http://sethc23 github.io/wiki/Python/geopandas. **26**, 2022 (2016)
29. Joyce, J., Lomow, G., Slind, K., Unger, B.: Monitoring distributed systems. ACM Trans. Comput. Syst. (TOCS) **5**(2), 121–150 (1987)
30. Kalman, R.E.: On the general theory of control systems. In: Proceedings First International Conference on Automatic Control, Moscow, USSR, pp. 481–492 (1960)
31. Karumuri, S., Solleza, F., Zdonik, S., Tatbul, N.: Cloud observability: a melting pot for petabytes of heterogenous time series. In: CIDR (2021)
32. Karumuri, S., Solleza, F., Zdonik, S., Tatbul, N.: Towards observability data management at scale. ACM SIGMOD Rec. **49**(4), 18–23 (2021)
33. Labs, G.: An open source visualisation tool. (2023). https://www.elastic.co/what-is/elk-stack. Accessed July 2023
34. Levin, J., Benson, T.A.: Viperprobe: rethinking microservice observability with eBPF. In: 2020 IEEE 9th International Conference on Cloud Networking (CloudNet), pp. 1–8 (2020)
35. MapBox: (2023). https://www.mapbox.com/. Accessed Aug 2023
36. Marie-Magdelaine, N., Ahmed, T., Astruc-Amato, G.: Demonstration of an observability framework for cloud native microservices. In: 2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), pp. 722–724
37. McGrane, M., Poon, S.K.: A method to estimate high-dimensional synergistic interactions: a case study on information technology business value (2011)
38. Naha, R.K., et al.: Fog computing: survey of trends, architectures, requirements, and research directions. IEEE Access **6**, 47980–48009 (2018)
39. OpenFog: Openfog consortium architecture working group (2017)
40. OpenTelemetry: - Python Library. https://opentelemetry.io/docs/instrumentation/python/. Accessed Aug 2023
41. Pallewatta, S., Kostakos, V., Buyya, R.: Placement of microservices-based IoT applications in fog computing: a taxonomy and future directions. ACM Comput. Surv. (2023)
42. Prometheus: An open-source metrics management system (2023). https://prometheus.io/. Accessed July 2023
43. Ranaweera, P., Jurcut, A.D., Liyanage, M.: Survey on multi-access edge computing security and privacy. IEEE Commun. Surv. Tutorials **23**(2), 1078–1124 (2021)
44. Sigelman, B.H., et al.: Dapper, a large-scale distributed systems tracing infrastructure (2010)
45. Srirama, S.N.: A decade of research in fog computing: Relevance, challenges, and future directions. arXiv preprint arXiv:2305.01974 (2023)
46. SysStat: Sar - system activity report (2023). https://github.com/sysstat/sysstat. Accessed Aug 2023
47. Taherizadeh, S., Jones, A.C., Taylor, I., Zhao, Z., Stankovski, V.: Monitoring self-adaptive applications within edge computing frameworks: a state-of-the-art review. J. Syst. Softw. **136**, 19–38 (2018)
48. Tzanettis, I., Androna, C.M., Zafeiropoulos, A., Fotopoulou, E., Papavassiliou, S.: Data fusion of observability signals for assisting orchestration of distributed applications. Sensors **22**(5), 2061 (2022)
49. Usman, M., Ferlin, S., Brunstrom, A., Taheri, J.: A survey on observability of distributed edge & container-based microservices. IEEE Access (2022)
50. Usman, M., Ferlin, S., Brunstrom, A., Taheri, J.: Desk: distributed observability framework for edge-based containerized microservices. In: EuCNC/6G Summit, pp. 617–622. IEEE (2023)

51. Vaquero, L.M., Cuadrado, F., Elkhatib, Y., Bernal-Bernabe, J., Srirama, S.N., Zhani, M.F.: Research challenges in nextgen service orchestration. Futur. Gener. Comput. Syst. **90**, 20–38 (2019)
52. Vaquero, L.M., Rodero-Merino, L.: Finding your way in the fog: towards a comprehensive definition of fog computing. SIGCOMM Comput. Commun. Rev. **44**(5), 27–32 (2014)
53. Yi, S., Li, C., Li, Q.: A survey of fog computing. InL Proceedings of the 2015 Workshop on Mobile Big Data - Mobidata 2015, pp. 37–42 (2015). http://dl.acm.org/citation.cfm?doid=2757384.2757397

Adaptive QoS-Aware Task Offloading in Dynamic Mobile Edge Computing Environment

Jacob Don, Sajib Mistry, Redowan Mahmud(✉), and Aneesh Krishna

School of Electrical Engineering, Computing and Mathematical Sciences, Curtin University, Perth, Australia
mdredowan.mahmud@curtin.edu.au

Abstract. Ensuring Quality of Service (QoS) for real-time applications like Augmented Reality in a Mobile Edge Computing (MEC) setting is both vital and demanding in research. In this work, we propose a novel framework of hybrid ML approaches to enable QoS-aware offloading in dynamic MEC. First, we create a method using deep reinforcement learning to figure out the best way to offload tasks for an application in a new MEC environment, even when we don't have early information about the application's QoS. Then, we deploy a transfer learning approach in the dynamic MEC that transfers knowledge from previously trained deep reinforcement off-loading policies to new optimal policies. Experimental results show that the proposed framework adapts to the dynamic MEC environments efficiently, reducing ML training time and retaining higher accuracy and precision during the task offloading process.

Keywords: Quality of Service · Mobile Edge Computing · Task offloading · Hybrid machine learning

1 Introduction

The advent of the *Internet of Things* has seen a rise in many real-time application services where the Quality of Service (QoS) is vital; one such service is *Augmented Reality* (AR). AR is an interface between a user and the natural world that artificially augments the user's experience by adding computer-generated content [3]. The content is provided in real-time and is dependent on the context of the user's environment. Some of the most well-known uses of AR today are on modern smartphones, with the AR game Pokemon Go receiving millions of downloads and the social media app Snapchat providing face-altering AR [12].

Despite AR being a functional area of computing, there have been few real-world implementations. This is due to AR requiring devices to be small and portable but also a large amount of RAM, power and fast CPUs to run efficiently [11]. While modern-day smartphones have powerful cameras and are highly portable, the intensive requirements of AR mean that mobile implementations are slow, poor quality, and power-draining. This is just one of many applications unable to be fully functional due to the divide between their computational complexity and the capabilities of modern devices.

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 341–352, 2024.
https://doi.org/10.1007/978-3-031-63992-0_22

A potential solution to this issue is utilising *cloud computing* to solve the problem of user devices not having enough computational resources for the AR tasks [9]. The use of cloud computing caused a new bottleneck, the network connecting the cloud to the user. These networks cannot keep up with the demands of the computations resulting in significant lag [15]. This bottleneck means that real-time applications such as AR that cannot run natively on mobile devices experience substantial latency problems when offloading to the cloud [16]. Hence, the use of Mobile Edge Computing (MEC) is becoming prevalent, which focuses on moving the computation from central cloud servers to mobile edge computing nodes at the user premises [16]. Task offloading with MEC and 5G allows devices to compute tasks with improved response times and QoS.

Utilising MEC improves latency by reducing the distance the data travels, and combined with 5G technology, results in even lower latency. The use of MEC brings up the issue of how to best distribute the tasks among the Mobile Edge Servers. This problem of optimal task distribution is known to be NP-Hard [10]. The NP-Hard nature makes using heuristic algorithms to solve this problem difficult. Machine Learning based solutions are suitable candidates, e.g. Artificial Neural Networks (ANN), to address the NP-Hard problem of trask offloading [2]. Due to ANN's fixed input and output sizes, it is challenging to fit ANNs in the dynamic environment as a new model must be trained for every new environment and task offloading.

Deep Q Networks (DQNs) are a type of *reinforcement* learning that utilizes ANNs to learn optimal actions to take in a dynamic environment. DQNs and algorithms derived from them have shown strong performances in the area of dynamic task distribution [2,7,18]. *We have identified the following limitations of DQN-based solutions in the dynamic MEC task offloading problem*:

- **Fluid Adaptivity**: Solutions require retraining on every possible environment change due to the neural network used to estimate values which is not optimal in real-world applications.
- **Faster Convergence**: Training DQNs has a slow convergence, meaning it takes a considerable amount of time and computing capability to develop a DQN that provides strong placements of tasks in the runtime.

Without addressing the issues of adaptivity, the performance of a network can be greatly reduced, resulting in lag and poor QoS. *To address this, we introduce a hybrid DQN-transfer learning approach in which learned domain knowledge is kept when the environment changes.* The main **contributions** of the paper are:

- A hybrid DQN-transfer learning framework for dynamic MEC environment.
- The formulation of MEC task offloading environment as a Markov Decision Problem through which reinforcement learning algorithms can be trained.
- The performance evaluation of the proposed framework with synthetic data, comparing it to the ideal results using the baseline brute force algorithm.

The rest of the paper is organised as follows. Section 2 highlights the literature on task offloading in MEC. Section 3 discusses the proposed adaptive QoS-aware task offloading policy. Section 4 outlines the experimental setups to evaluate the performance of our proposed solution. Finally, Sect. 5 concludes the paper.

2 Related Work

Deep Q learning is a type of reinforcement learning that has seen significant research in the area of task offloading for MEC architectures due to its ability to learn to navigate a given environment to maximize an arbitrary reward function [14]. Unlike its predecessor, tabular Q learning, DQN is able to learn environments with arbitrarily large state spaces. X. Chen, et al., [2] uses deep reinforcement learning to solve energy-efficient task distribution. They use a MADDPG deep learning algorithm as the basis of their artificial intelligence. Reinforcement learning means the model can be taught without knowing the ideal outcome beforehand. Combined with deep learning means, the model can learn very complicated patterns that even a human would miss. Promising experimental results, power consumption was down 10% in the multi-MEC system, with the delay staying well below the desired threshold.

K. Zhang, et al., [18] and M. Hossain, et al., [7] use Deep Q learning to produce a model that can offload tasks in an efficient manner. The model is one of the most simple forms of deep Q learning, meaning that the implementation is easy. Deep Q learning has issues with these stochastic problems as the state space can rapidly expand to larger than the deep Q learning algorithm can handle [1,4,13] meaning that this algorithm will not work with multiple users.

Transfer learning is a method of translating knowledge that a machine learning algorithm has in one task space into another closely related space [19]. There are many methods for each machine learning algorithm. The most common use for transfer learning is reducing the required training time. This is a useful area because training machine learning algorithms can often take multiple hours to multiple days. Therefore, any reduction in that time is ideal, especially when fine-tuning models, which can require them to be trained multiple times. Transfer learning methods are effective in deep reinforcement learning [19] and more generally in deep learning [17].

Z. Zhu et al., [19], provided a survey of the current research into transfer learning in the area of deep reinforcement learning. Providing reviews and descriptions of current methods such as Reward Shaping, Learning from Demonstrations and Representation Transfer. Our method focuses on the use of the Representation Transfer method. Representation transfer is an area of methods that employs the ability of DNNs to abstract information into representations and then transfer those learned representations.

JT. Huang et al., [8] was analyzed in C. Tan et al., [17] and shows how a DNN can be split into two parts, the first a reusable set of hidden layers that extracts features from a language, and the second portion that is retrained for each possible language and acts as a language classifier. The transferable first layer is a strong example of transfer learning.

One of the main issues in the existing research is that they have limited focus on fluid adaptability to a frequently changing MEC environment. To address this issue we propose a hybrid DQN-transfer learning approach for the task offloading incorporating fluild adaptivity and faster convergence in the runtime.

3 Adaptive QoS-Aware Task Offloading Framework

3.1 Motivation Scenario

Let's assume, George is a user of an augmented reality application running on his smartphone that provides information on local flora. However, while looking up information on plants once they are known is an easy task that any smartphone can complete, localising and identifying the plant in real-time is more complicated. The smartphone, however, has access to nearby servers with far more processing power through a 5G radio connection, which provides low latency. Utilising these servers to execute computationally intensive tasks results in a faster application. An example of an off-loadable task is finding the screen locations of any flowers. Figure 1 shows a simple situation of MEC task offloading. However, the number of tasks required for computation is far more dynamic in many situations, as is the number of compute servers that the user has available. Therefore, a method that can both provide intelligent task offloading and the ability to handle a dynamic changing environment must be selected.

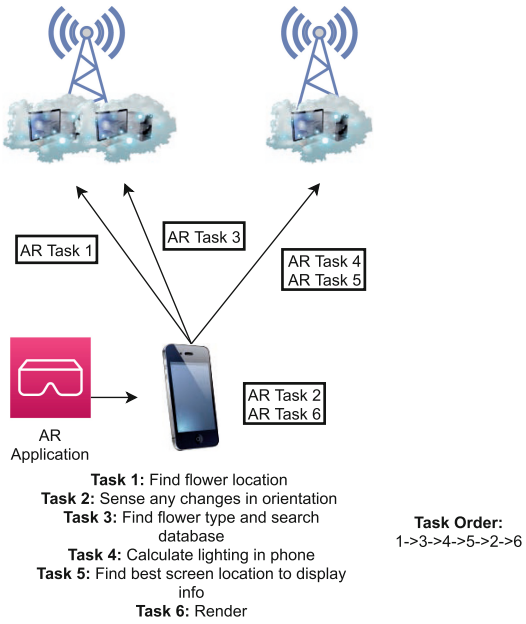


Fig. 1. AR Task Offloading

3.2 Problem Definition

Intelligent task offloading is a problematic issue that can be solved with reinforcement learning [2]. However, it is not feasible to train individual deep rein-

forcement models for each possible environment in a dynamically changing environment with current methods. The environment notations are given below:

- Task (T): A Task consists of an input size (T_i) in Mb, a processing requirement (T_p) in seconds based on a 1Ghz computer and an execution location (T_l) which is always initially set to C
- Edge Servers (S): An Edge Server is a computing location that can process tasks, it has a network connectivity speed (S_c) to the client in MBps, a processing speed (S_p) in Ghz and a count of tasks on the server (S_n)
- Clients (C): A client is the start location for all the tasks and another potential processing location with no network connection and therefore is not limited by input size and has a processing speed (C_p) in Ghz. The client also keeps track of the count of tasks running on the client (C_n).
- QoS: Quality of service is a set of factors that determines how well a distribution performs.

Assuming there is a set of tasks: $\{T_1, T_2, ...T_N\}$ and $\{S_1, S_2, ...S_M\}$ edge servers. The goal is to select a distribution of tasks that increases QoS to a maximum in the small time frame required by augmented reality applications.

3.3 Hybrid Framework for Task Offloading

In Fig. 2, the framework begins with the client device determining available edge servers and the task count. It checks pretrained models to find a suitable fit for the current environment. If a model fits, it's chosen; if not, a new model is trained. The client then utilizes the decision model for task offloading and location selection. The process repeats as the environment remains stable with infrequent changes in edge servers and task count.

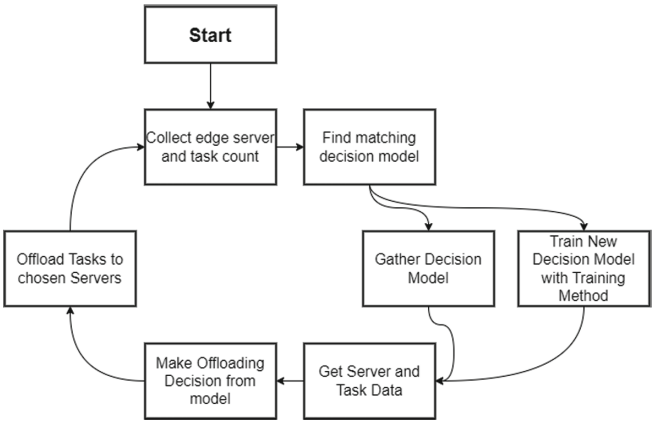


Fig. 2. Proposed Framework

We selected a DQN for the decision model despite its limitation with multiple users due to its large state space [1,4,13]. Its simplicity and proven effectiveness in research, along with its role in building other deep reinforcement

algorithms like MADDPG, led to the choice. We opted for the network-based transfer learning method of hidden weight copying due to its adaptability to deep neural network models, a modern AI paradigm.

3.4 DQN-Based Task-Offloading Decision Model

A DQN is a type of off-policy reinforcement learning that is derived from Tabular Q-Learning [14], wherein the Q-Table that is normally present is replaced with a Neural Network that learns to predict the Q Values of the current state and associated actions. It can be said that the neural network, which is a universal function approximator, attempts to approximate the function Q where $Q(S, A) = Qr$, S is the current state, A is an action that can be taken in that state and Qr is the resulting Q value of taking that action in the state. A Q value refers to the discounted cumulative reward that is derived from that action being taken in that state, the calculation for this is shown in 1.

$$Q = \sum_{t=t_0}^{\infty} (\gamma^{t-t0} r_t) \tag{1}$$

where γ refers to the discount value of the function, which controls how strongly the future reward is considered compared to the reward generated at the current step. This stops the DQN from becoming a greedy algorithm as in many cases a greedy method is not ideal, because the greatest reward can require following a path that has delayed reward.

The loss of the neural network is calculated by using the Bellman equation [14] and results in the loss Eq. 2, known as the temporal difference.

$$\delta = Q(s_t, a) - (r + \gamma(Q_1(s_{t+1}, a))) \tag{2}$$

In DQN, r is the reward, π is the policy (often a max function), and Q_1 is the target network [5]. The deep neural network approximating Q is updated via gradient descent. DQN is off-policy, enabling the exploration of unselected state spaces and preventing premature convergence. Typically, π is a greedy function selecting actions with the highest potential reward. To stabilize the target network during training, Q_1 is updated at intervals based on hyperparameter X. The DQN takes input in the form of an observation array, explained in Sect. 3.4.

Observation Array for DQN Decision Model. The DQN's observation array, a 1D array, contains vital environmental information. It encompasses the current task distribution, server and client CPU/network speed, task processing requirements, input size, and the current task. When the task distribution changes, we update the array by shifting the current task's execution location, modifying the server CPU and network speed to adapt to the new load, and increasing the current task index by 1. Frequent CPU and network speed updates ensure diverse observation arrays, preventing the neural network from consistently choosing the same task and, subsequently, poor performance.

3.5 Proposed Hybrid Learning Model for Training Method

Transfer learning is a way of using a pre-trained model to decrease the training time of a new model. In the case of DQN, this can be done in many ways [19]. However, in this implementation, a simple weight transfer and layer lock will be used as described by H. George, et al. [6] as the goals in each of the networks are very similar. Certain layers' weights will be loaded in from a previously trained DQN, these layers will then be locked so they can no longer be trained. This will improve the training speed in two ways. Firstly there will be fewer trainable parameters and therefore the speed of convergence will increase. Secondly, the layers with loaded weights will remember the previous attributes and features they were taught, meaning the DQN will be already partly trained.

QoS Feedback from Edge Services. The simulation used to emulate the environment in which the DQN will be used consists of the three main objects, Client, Severs, and Tasks as described in 3.2. The task distribution is simply represented as a list of integers with size, number of tasks, therefore the total processing time required for any distribution can be calculated as shown in 6.

$$Server(T) = \frac{8T_i S_n(T_l)}{10^3 S_c(T_l)} + \frac{T_p S_n(T_l)}{S_p(T_l)} \tag{3}$$

$$Client(T) = \frac{T_p S_n(T_l)}{S_p(T_l)} \tag{4}$$

$$TTime(T) = \begin{cases} Client(T) & \text{if } T_l = 0 \\ Server(T) & \text{if } T_l \neq 0 \end{cases} \tag{5}$$

$$Time = \sum_{t=0}^{taskNum} TTime(t) \tag{6}$$

This calculation simplifies the process to enable the generalization of task distribution ideas for deep reinforcement learning. It incorporates task-specific weights to influence execution time. The reward function prioritizes overall reduction, summing all task processing times, rather than just the slowest task. This approach encourages a more substantial reduction, albeit not necessarily reaching the global minimum due to unpredictable variables.

Training the Proposed Hybrid Model. An episode is defined as starting with all tasks distributed on the client-side, then allowing the algorithm to pick new locations for the tasks one by one, ending when all tasks have had their distributions chosen. A reward is generated at the end of each task decision. It is calculated as a percentage increase or decrease from the previous steps time. If it is a decrease, the returned reward is positive, however, if it is an increase, the returned reward is negative to force the DQN to aim to reduce the running time. This percentage is then broken down into the tiers shown in 1 which return their associated reward.

Table 1. Reward Calculation

Result	0	< 1	< 10	< 20	< 30	< 40	< 50	< 70	< 90	> 90
Decrease	0	1	10	15	35	50	60	90	120	150
Increase	0	-2	-20	-30	-70	-100	-120	-180	-240	-300

We used percentages instead of final processing times due to variable minimum processing times in different environments. Reward is categorized into tiers to reduce the need for precise predictions, promoting a stronger distribution. While accuracy may decrease, the focus is on a robust distribution, not the global minimum. The reward function disregards the one's digit, encouraging positive development. Rewards are provided at every offloading action to reduce sparsity and enhance training stability.

Choosing the training policy is vital for DQN training, affecting its exploration level. There are two common policies, the epsilon greedy policy and the BoltzmannQPolicy. The epsilon greedy policy takes in a parameter ϵ with a value range of 0 to 1 and the Q values generated by the neural network. With probability ϵ it selects a random action and with probability $1 - \epsilon$ it selects the action with the highest Q value. This means the amount of exploration is directly correlated to ϵ. The BoltzmannQ policy however generates a probability distribution based on a temperature value T and the Q values generated by the neural network. This has the same effect as when T is large the DQN experiences large amounts of exploration, then when small. However, the exploration is biased toward actions with high Q values.

4 Experiments

4.1 Experimental Setup

For this experiment, all tasks will have the same emphasis, however, in real-world implementations of this framework, weighted tasks can be extremely beneficial as certain algorithms require certain tasks to be finished quicker. Table 2 outlines the environment variables used during the simulation.

Two environments were used for individual DQN training, each with 4 servers and 15 tasks. The first DQN was trained in an environment with 3 servers and 10 tasks; its weights were transferred to a new DQN for training in the second environment. Notably, weights for the input-to-first-hidden-layer and last-hidden-layer-to-output-layer were not copied due to different dimensions. These DQNs underwent 60 iterations with the same hyperparameters.

During transfer learning, various hidden layer locking patterns were tested, including locking the first hidden layer, no layers, all layers, and the middle three layers. The top-performing pattern was selected and retrained for 200 iterations.

Table 2. Environment Variables

Variable	Value
S_c	1–50 MB/s
S_p	1–5 GHz
T_i	100–1000 MB
T_p	0.0001–4 s/GHz
C_p	1–2 GHz

The epsilon-greedy policy was used during DQN training to encourage exploration. The epsilon value decreased linearly from 0.9 to 0 over 10,000 training steps, allowing the DQN to explore broadly initially and then focus on states closer to the ideal state as it improved. Testing employed a simple greedy algorithm, always selecting the action with the highest value (Table 3).

Table 3. Hyperparameters

Parameter	Value
Policy	Epsilon Greedy
Epsilon	0.9–0
Learning Rate	1e−5
Target Network Update	1000
Training Steps	2,000,000
Hidden Layer Size	$400 \times 400 \times 400 \times 300 \times 200 \times 100 \times 50$

4.2 Analysis

The Baseline Approach. We use a brute force method to calculate the global minimum time, aka the global maximum reward. Using this we can then calculate the Root Mean Square Error (RMSE) of our Deep Q method after training. To compare during training it is not feasible to run the brute force algorithm for every data point due to its high computational complexity, therefore, we use the reward generated by the simulation to compare methods.

Hybrid Task-Offloading Model Results. In Fig. 3, raw data from various locking patterns is presented. The network parameters were initially trained in an environment with 3 servers and 10 tasks, then transferred to a new network with different input and output shapes for retraining in a 4-server, 15-task environment. The locking patterns considered were all-lock, first-lock, half-lock, and no-lock. No-lock performed the best, followed closely by all-lock and first-lock, while half-lock was less effective. For subsequent experiments, the no-lock method will be employed.

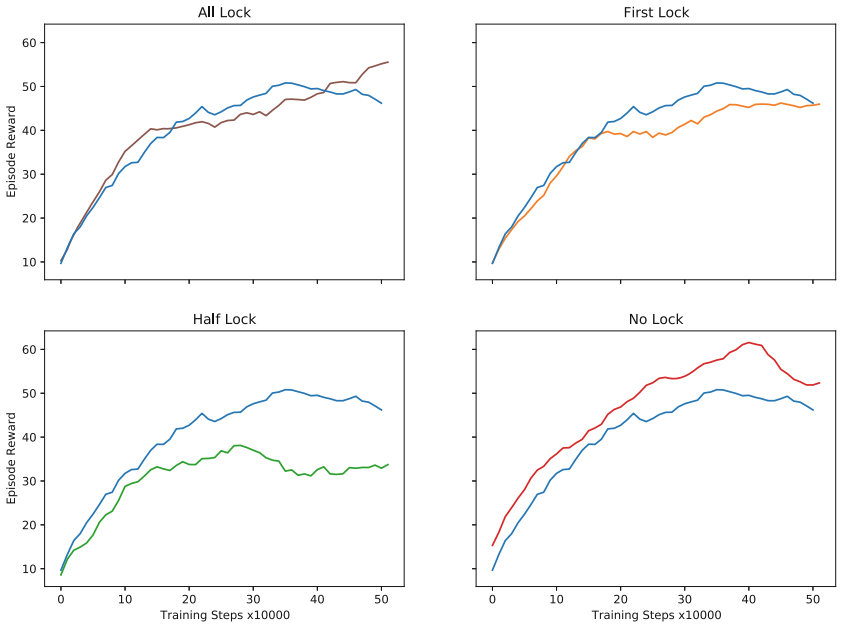


Fig. 3. First 60 Training Steps of Locking Patterns (All-Lock (Brown), First Lock (Orange), Half Lock (Green), No Lock (Red)) Compared with Vanilla DQN (Blue) Training Smoothed (Color figure online)

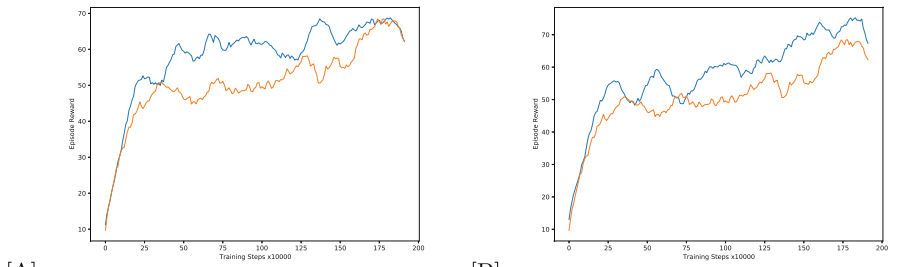


Fig. 4. Transfer Learning DQN Training with No Lock (Yellow) vs Vanilla DQN (Blue) (Color figure online)

Figure 3 compares the locking methods' 60 training steps to the first 60 training steps of the DQN trained in the vanilla DQN method. These graphs further show how the no-locking method performs much better than the other methods.

The DQN was trained using transfer learning with a no-lock method in two runs. Figure 4.A and 4.B display the raw data for both training iterations. The locking method consistently outperformed vanilla DQN training, with the second run finishing closer to 65 than 60, indicating significant improvement.

Comparative Analysis. Distributions are challenging to assess accurately, so they are compared to the brute force algorithm's ground truth. The ground truth averaged a 77% reduction, while the proposed method achieved an average reward of 70, equivalent to a 50–60% reduction in offloading time. These results are robust, given that the brute force algorithm takes about a minute to run, even in relatively small exploration spaces. Table 4 displays root mean square errors, demonstrating that the model performs well in smaller state spaces and that transfer learning and vanilla DQN [7] yield similar results.

Table 4. Root Mean Square Error

Model	RMSE
3×10 Vanilla DQN	0.33
4×15 Vanilla DQN	0.43
4×15 Hybrid Framework	0.43

The DQN, with limited environment information, also achieves a 50–60% reduction in overall running time, while the baseline averages around 80%. The root mean square errors of 0.33 and 0.43 indicate the closeness of the results. Employing DQN for task distribution in Mobile Edge computing with 5G results in an approximate 50% performance boost. Using transfer learning, although end results match, the training curve yields higher rewards, reducing training time compared to vanilla DQN.

5 Conclusion

We addressed the task offloading issue in MECs via 5G. Traditional DQNs and other reinforcement learning approaches struggle with dynamic environments. Our proposal leverages transfer learning to speed training and enhance DQNs' adaptability. We presented a framework for handling task offloading adaptivity using DQNs and transfer learning, explained the simulation environment, and conducted experiments. Results demonstrated that DQNs, with transfer learning, achieve a significant 50–60% reduction in task running time, maintaining learned knowledge across domains for faster convergence and adaptability.

References

1. Chen, X., Zhang, H., Wu, C., Mao, S., Ji, Y., Bennis, M.: Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning. IEEE Internet Things J. **6**(3), 4005–4018 (2018)
2. Chen, X., Liu, G.: Energy-efficient task offloading and resource allocation via deep reinforcement learning for augmented reality in mobile edge networks. IEEE Internet Things J. 1–1 (2021). https://doi.org/10.1109/JIOT.2021.3050804

3. Furht, B.: Handbook of Augmented Reality. Springer, New York (2011)
4. Hasselt, H.: Double q-learning. Adv. Neural. Inf. Process. Syst. **23**, 2613–2621 (2010)
5. van Hasselt, H., Guez, A., Silver, D.: Deep reinforcement learning with double q-learning. CoRR **abs/1509.06461** (2015). http://arxiv.org/abs/1509.06461
6. He, G., Yang, D., Shen, K.: Improving DQN training routines with transfer learning (2019)
7. Hossain, M.S., Nwakanma, C.I., Lee, J.M., Kim, D.S.: Edge computational task offloading scheme using reinforcement learning for IIoT scenario. ICT Express **6**(4), 291–299 (2020)
8. Huang, J.T., Li, J., Yu, D., Deng, L., Gong, Y.: Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 7304–7308. IEEE (2013)
9. Huang, Z., Li, W., Hui, P., Peylo, C.: Cloudridar: A cloud-based architecture for mobile augmented reality. In: Proceedings of the 2014 workshop on Mobile Augmented Reality and Robotic Technology-Based Systems, pp. 29–34 (2014)
10. Kartik, S., Siva Ram Murthy, C.: Task allocation algorithms for maximizing reliability of distributed computing systems. IEEE Trans. Comput. **46**(6), 719–724 (1997). https://doi.org/10.1109/12.600888
11. Kounavis, C.D., Kasimati, A.E., Zamani, E.D.: Enhancing the tourism experience through mobile augmented reality: challenges and prospects. Int. J. Eng. Bus. Manag. **4**, 10 (2012)
12. Ling, H.: Augmented reality in reality. IEEE Multimedia **24**(3), 10–15 (2017)
13. Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., Mordatch, I.: Multi-agent actor-critic for mixed cooperative-competitive environments. arXiv preprint arXiv:1706.02275 (2017)
14. Mnih, V., et al.: Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602 (2013)
15. Shi, W., Dustdar, S.: The promise of edge computing. Computer **49**(5), 78–81 (2016)
16. Singh, S.: Optimize cloud computations using edge computing. In: 2017 International Conference on Big Data, IoT and Data Science (BID), pp. 49–53. IEEE (2017)
17. Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., Liu, C.: A survey on deep transfer learning. In: Kůrková, V., Manolopoulos, Y., Hammer, B., Iliadis, L., Maglogiannis, I. (eds.) ICANN 2018. LNCS, vol. 11141, pp. 270–279. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01424-7_27
18. Zhang, K., Zhu, Y., Leng, S., He, Y., Maharjan, S., Zhang, Y.: Deep learning empowered task offloading for mobile edge computing in urban informatics. IEEE Internet Things J. **6**(5), 7635–7647 (2019)
19. Zhu, Z., Lin, K., Zhou, J.: Transfer learning in deep reinforcement learning: a survey. arXiv preprint arXiv:2009.07888 (2020)

Discretization-Based Ensemble Model for Robust Learning in IoT

Anahita Namvar[1](✉), Chandra Thapa[2], and Salil S. Kanhere[1]

[1] UNSW Sydney, Kensington, NSW 2052, Australia
a.namvar@student.unsw.edu.au, salil.kanhere@unsw.edu.au
[2] Data61 Marsfield, Sydney, NSW 2122, Australia
chandra.thapa@data61.csiro.au

Abstract. The rapid proliferation of Internet of Things (IoT) devices has introduced new challenges in network management and security. While machine learning models hold promise for identifying these devices, they remain vulnerable to adversarial attacks, undermining their accuracy and reliability. This paper addresses the need for robust IoT device identification by proposing a novel approach: a discretization-based ensemble stacking technique. This method harnesses both the protective properties of discretization and the generalization benefits of ensemble methods. Through extensive experimentation, we demonstrate the efficacy of our approach against various adversarial attacks, showcasing its potential to enhance the resilience and accuracy of IoT device identification models in dynamic and uncertain environments.

Keywords: Adversarial Robustness · IoT · Discretization · Ensemble

1 Introduction

The Internet of Things (IoT) has revolutionized the way we interact with our surroundings, connecting a wide range of objects, from household appliances to smartphones and automobiles. However, this connectivity has also introduced several challenges for organizations, particularly with regard to managing access and preventing potentially insecure IoT devices from connecting to their network [1]. To address this challenge, accurate and fast methods are needed to effectively manage the visibility of IoT devices on the network and distinguish compromised devices from legitimate ones.

Machine learning models have emerged as a promising approach for IoT device identification, as they can learn to recognize the unique behavior patterns of each device on the network [2]. However, these models are susceptible to various types of noise and adversarial attacks that can potentially affect their accuracy and performance [3–5]. These attacks can introduce perturbations into input data, leading to misclassifications and undermining the reliability of device identification models. In existing literature, while the application of machine learning for IoT device identification is acknowledged,

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 353–367, 2024.
https://doi.org/10.1007/978-3-031-63992-0_23

there is a noticeable gap concerning the development of a comprehensive and effective defence strategy against adversarial attacks in this context. Particularly as a means of enhancing robustness, there is a lack of exploration into discretization and ensemble methods in the realm of IoT device classification. Discretization making the input space less sensitive to small perturbations. Furthermore, Ensemble methods often help in reducing overfitting and improving generalization, which are important factors for model robustness. To address this gap, the following research questions have been formulated:

Research Question 1: How does the incorporation of discretization methods as defence mechanisms for IoT device identification classifiers enhance their security against both white-box and black-box adversarial attacks?

Research Question 2: How does the development of a hybrid approach that effectively integrates discretization and ensemble methods synergistically improve the robustness of IoT device identification classifiers against diverse white-box and black-box adversarial attacks using real-world IoT network traffic data?

We summarized the key contributions of this paper in the following:

Contribution 1: Empirical Validation of Discretization Effectiveness

We practically validate the efficacy of discretization methods as defence mechanisms by showcasing their ability to fortify IoT device identification classifiers against a spectrum of adversarial attacks, pinpointing the most effective techniques. Through rigorous experiments, it establishes these methods as a valuable additional layer of protection, aligning with the broader contributions that enhance IoT security resilience.

Contribution 2: Synergistic Enhancement via Discretization-Based Ensemble Defence

In response to Research Question 2, this paper introduces a novel and comprehensive approach that synergistically leverages discretization and ensemble techniques for enhancing the robustness of IoT device identification classifiers. This contribution advances the field by presenting a holistic defence methodology that combines the strengths of discretization and ensemble methods to create a more resilient defence mechanism. The proposed hybrid approach not only addresses the challenges of both white-box and black-box attacks but also offers a practical solution for real-world IoT network traffic scenarios.

These contributions collectively advance the knowledge in the field of IoT security by demonstrating the potential of discretization methods as defence mechanisms, empirically validating their efficacy, and proposing a sophisticated hybrid approach that showcases the synergy between discretization and ensemble techniques in enhancing the robustness of IoT device identification classifiers.

The rest of this paper is organized as follows: Sect. 2 provides backgrounds and related works. Section 3 presents the details of our research methodology. Section 4 provides our experiments and results, and Sect. 5 concludes the paper.

2 Background and Related Works

2.1 IoT Device Classification and Adversarial Attacks

The vulnerability of machine learning models to adversarial attacks is mostly addressed in the image domain, and less amount of research has been done on the security of machine learning base device identification models. Namvar, et al. [5] presented the vulnerability of device identification models, including Random Forest (RF), Logistic Regression (LR), Feed Forward (FF), and Decision Tree (DT) to adversarial attacks. However, they did not provide any solution to increase the models' robustness against attacks. Singh and Sikdar [6] investigated the effectiveness of generating attacks against a deep learning-based appliance classification model using smart meter data. The study found that adversarial attacks can significantly degrade the accuracy of the model. To investigate the detectability of attacks in the smart home domain, the authors proposed a visualization method that compares the distribution of true data points and adversarial data points. The findings of the study highlight the need for effective defence mechanisms to counter adversarial attacks on deep learning-based appliance classification models in the smart home domain. Meidan, et al. [7] proposed random forest supervised machine learning model to detect suspicious IoT devices connected to organizational networks. The results showed that the algorithm correctly classified, white-listed device types in 99% of cases and detected non-white-listed device types in 96% of test cases. The authors presented the resilience of their method to adversarial attacks. Kotak and Elovici [8] propose a new approach for generating real-time adversarial examples using heatmaps generated by CAM and Grad-CAM++ that can be applied in the computer network domain. Authors revealed that in many cases, an adversarial example created using a heatmap could deceive the payload-based deep IoT device identification solution with up to 100% accuracy.

2.2 Discretization-Based Ensemble as a Defence Strategy

Discretization, also known as quantization or binning, is the process of converting continuous input variables to discrete values by dividing the input range into a fixed number of bins or intervals [9]. Discretization has been introduced for secure deep learning in image datasets [10]. Until recently, discretization robustness was mostly studied in computer vision [11–13]. However, only one research presented discretization as a defence mechanism in protecting tabular data from adversarial attack [14].

There are several methods for discretization [15], some of them are used in this study Equal Width, Equal Frequency, Entropy-based Binning, and Minimum Descriptive Length. These methods divide a feature's range into bins using different approaches.

Figure 1 illustrates a simplified view of data discretization and shows how it improves robust machine learning against adversarial attacks. The White circle in Fig. 1 represents the original input sample, denoted as X_1. The black circle represents the adversarial sample, denoted as $\widehat{X}_1$, which has been maliciously tampered with by an attacker. The attacker introduces a slight perturbation, ε, , to the input data. This perturbation leads to misclassification. However, let us suppose that we have defined bin boundaries $(C_1, C_2, \ldots, C_n)$ after applying a discretization algorithm. The bin boundaries can be

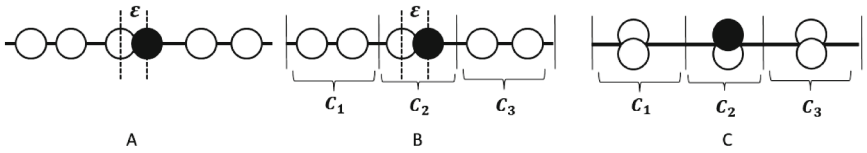


Fig. 1. A simple view of discretization for improving robustness. A. Data before discretization. Continuous variables, and black circle presents an adversarial sample. B. Data is discretized into three equal ranges. All data in the same category have been mapped to an equal value. C. Data after discretization.

seen in Fig. 1. B. After defining bin boundaries, each input data is assigned to the appropriate bin. Finally, the original input data is replaced with the discretized versions in the model. Figure 1.C depicts the discretized samples. Hence, the small changes in the input data cannot fool the model. For example, both X_1 and $\widehat{X}_1$ will be categorized in bin 3, and adversarial attacks cannot simply misclassify input data [16].

Ensemble robustness is another method that can increase the security of machine learning models against adversarial attacks [17–19]. In this study, we illustrate the idea of using an ensemble of different robust discretised classifiers to improve the model's predictive performance in adversarial environment.

3 Methodology

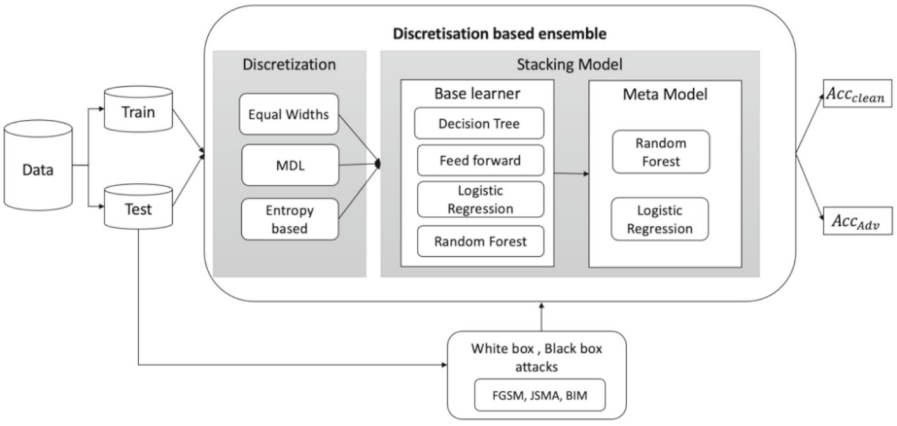


Fig. 2. Discretization-based ensemble robustness.

This section discusses the proposed methodology for the robust learning based IoT device identification mechanism that can improve the resilience of models in an adversarial environment. Figure 2 illustrated the overall methodology of the proposed defence method. We generate FGSM, BIM, and JSMA adversarial attacks in both white and black scenarios. Our defence method specially addresses three different discretization methods of EW, MDL, and EBD collectively enabling the conversion of continuous input

data into discrete representations. Additionally, we utilize stack ensemble to combine different predictions to enhance the model accuracy in adversarial settings.

3.1 Data

In the initial component of the methodology, the study utilizes real-world IoT data to lay the foundation for model development and evaluation. This dataset serves as a crucial source of information, reflecting the complexities of IoT device behaviour in authentic environments. Through rigorous data pre-processing, including cleaning and feature extraction, the dataset is refined for subsequent phases. The process entails a meticulous division into training and testing subsets, enabling the construction and validation of robust machine learning models.

3.2 Discretization

First, we discretized input features into categorical features; we used four state-of-the-art discretization methods including, Equal Frequency (EF), Equal Width (EW), Minimum Description Length (MDL), and entropy-based discretization. These methodologies were chosen due to their ability to effectively discretize continuous data.

In our methodology, each classifier undergoes training with input data that has been transformed using different discretization models. This process results in the creation of multiple base models for each individual classifier, where each base model corresponds to a specific discretization approach. For instance, if we consider three discretization techniques namely, EW, MDL, and EBD applied to the input data, each classifier will be trained separately using these three different discretized datasets. Consequently, for each classifier, a set of three base models emerges, each reflecting the impact of a distinct discretization method.

3.3 Adversarial Attack

In this study, the most state-of-the-art adversarial attacks, including FGSM, BIM, and JSMA, were employed under two scenarios of a white box and a black box to evaluate the robustness of machine learning based IoT device identification models. These attacks can be generated by adding imperceptible perturbations to the input data, which can evade the machine learning models and cause misclassification of IoT devices [5]. For example, an attacker could generate adversarial examples to make a device appear as a legitimate device to avoid detection while it is malicious and performing harmful actions on the network.

3.4 Ensemble

The ensemble method addressed in this study is the stacking ensemble. we address EW, MDL, and entropy discretization techniques since they lead to higher performance on robust accuracy. The stack ensemble model involves two levels of machine learning models: level 0 and level 1. The former consists of single models trained on discretized input features, while the latter is limited to only one meta-model of either RF or LR that combine the outcome of discretized single models.

3.5 Robustness Measures

The "Robustness Measures" component encompasses the quantitative evaluation of model performance under adversarial conditions. Accuracy, a fundamental metric, gauges the model's correctness in predicting device classifications. Meanwhile, we establish the robustness measure as the percentage decrease in model accuracy when comparing the model's performance between a clean and adversarial environment. The Robustness measure (RM) is depicted by Eq. 1.

$$RM = \frac{\|Acc_{Clean} - Acc_{adv}\|_1}{|Acc_{Clean}|} \tag{1}$$

Together, these measures offer a comprehensive insight into the model's capacity to safeguard IoT device identification in dynamic and threat-prone environments.

4 Experimental Results

In this section, we present a case study to demonstrate the practical application and effectiveness of the proposed defence method in IoT device identification. The case study uses a real-world IoT device data set. We focus on the proposed performance metrics to assess the efficacy of our defence mechanism in enhancing the resilience of the ML model against adversarial attacks. The results and analysis highlight the potential of the proposed defence method in securing IoT device classifiers and enhancing their resilience against adversarial threats. The experiments have been run in Python 3.8 on an Intel® Core i5 2.3 GHz CPU, 64 GB of RAM, and running on MAC OS.

4.1 Dataset

We utilized a comprehensive dataset obtained by Sivanathan, et al. [20]. This dataset encompasses a diverse array of network traffic data emanating from 28 distinct IoT devices, encompassing categories such as cameras, lights, motion sensors, appliances, and health monitors. It comprises seven fundamental attributes including traffic flow, flow volume, flow duration, sleep time, NTP interval, DNS interval, and domain count. This dataset forms the cornerstone of our analytical efforts, enabling a holistic exploration of network behaviours exhibited by various IoT devices.

4.2 Adversarial Attack Setting and Evaluation Metric

To evaluate the robustness of the proposed defence mechanism, our study employs three widely recognized adversarial attack techniques: Fast Gradient Sign Method (FGSM), Basic Iterative Method (BIM), and Jacobian-based Saliency Map Attack (JSMA). These attacks are conducted under two distinct scenarios: black box and white box attacks.

In black box attacks, the transferability property of adversarial examples is harnessed to evaluate the resilience of the defence mechanism. We generate adversarial perturbations using substitute models, specifically Logistic Regression (LR) and Feed-forward Neural Network (FF). This approach leads to various combinations, resulting in six different black box attacks: FGSM-LR, FGSM-FF, BIM-LR, BIM-FF, JSMA-LR, and JSMA-FF. For instance, FGSM-LR refers to a black box attack generated using the substitute model of LR. Similarly, FGSM-FF signifies a black box attack created via the substitute model of FF. A complete list of the different combinations of attack and substitute models can be found in Table 1. Such black box attack methods are likely to effectively fool the model due to the transferability property [21, 22]. These adversarial inputs then transferred to the target IoT device classifiers, including Random Forest (RF), Decision Tree (DT), LR, and FF.

In the white box attack setting, the adversary possesses complete knowledge of the learning model, its parameters, and training data. We exclusively employ white box attacks on differentiable classifiers, specifically LR and FF. For non-differentiable classifiers, such as DT and RF, only black box attacks are implemented.

Adversarial examples are crafted using the default settings in IBM's Adversarial Robustness Toolbox [23]. The FGSM attack parameter ϵ is set to o.o1. For a BIM attack, parameters ϵ and α are set to 0.01, and 0.001, respectively. Finally, for the JSMA attack, the parameters, Theta, and gamma are set to 0.01, 0.4, respectively.

In terms of the evaluation metric, we introduce the robustness measure (RM) in Sect. 3.5, represented by Eq. 1. A lower value of RM indicates enhanced robustness. Table 1 presents a comprehensive analysis of the model performance under different black box attack scenarios. The Acc_{adv} column indicates the accuracy of models against black-box attacks, while RM is the robustness measure that quantifies the percentage drop in model accuracy when comparing the model's performance in clean and adversarial environments. The outcome of these black box attacks collectively underscores the fact that no model remains impervious to the influence of adversarial attacks.

Upon analysing the white box attack results presented in Table 2, it becomes evident that the susceptibility of models to adversarial perturbations is not exclusive to black box scenarios. Models such as Logistic Regression and Feed Forward exhibit notable accuracy reductions under various white box attacks, including FGSM, JSMA, and BIM. For instance, Logistic Regression demonstrates a considerable decrease in accuracy across all white box attacks, particularly with FGSM where accuracy drops to 60%. Feed Forward also showcases variations in accuracy reductions depending on the specific attack method.

The results highlight that no model is immune to the impact of adversarial attacks. Under various black and white attacks techniques, including Fast Gradient Sign Method (FGSM) and Jacobian-based Saliency Map Attack (JSMA), accuracy reductions are observed across the board. Some models exhibit more pronounced declines in accuracy and robustness margins than others, indicating varying degrees of susceptibility. While the severity of vulnerability varies depending on the specific attack and model combination, it can be observed that all models experience compromised performance.

Table 1. Black box attacks

Model	Black box Attack	Adv_acc	RM
Decision Tree	FGSM LR	75.78	23.77
	JSMA LR	75.67	23.88
	BIM LR	75.86	23.69
	JSMA FF	74.37	25.19
	FGSM FF	78.11	21.43
	BIM FF	76.61	22.93
LR	JSMA FF	65.21	29.88
	FGSM FF	78.13	15.98
	BIM FF	73.83	20.60
Random Forest	FGSM LR	64.82	35.12
	JSMA LR	73.92	26.01
	BIM LR	66.11	33.83
	JSMA FF	71.52	28.40
	FGSM FF	73.88	26.05
	BIM FF	70.44	29.49
FF	FGSM LR	79.42	13.74
	JSMA LR	65.76	28.58
	BIM LR	78.50	14.74

Table 2. White Box Attacks

Model	Black box attack	Adv_acc	RM
LR	FGSM	60.10%	35.37%
	JSMA	64.82%	30.29%
	BIM	64.98%	30.12%
FF	FGSM	76.00%	17.45%
	JSMA	62.29%	32.34%
	BIM	80.29%	12.79%

4.3 Discretization

In this section, we thoroughly examine the evaluation of how different discretization methods impact the robustness of IoT device classifiers in the adversarial environment. We consider four IoT device identification models namely, Random Forest, Decision

Tree, Logistic Regression, and Feedforward Neural Network. We employ several discretization methods as detailed in Sect. 3.2, including Equal Frequency (EF), Equal Width (EW), Minimum Description Length (MDL), and entropy-based discretization. We examine the models under both white box and black box attack scenarios to comprehensively gauge their resilience. We assess the classifier's robustness to adversarial attacks in terms of accuracy, and RM metric.

First, we show the accuracy of models when the models are not exposed to adversarial manipulation (see Table 3). Original column presents the performance of models in a clean environment where there is no discretization or ensemble mechanism involved. We use this information as a baseline for evaluating the performance of robustness mechanisms. Columns EF, EW, MDL, EBD, EN_LR, and EN_RF correspond to equal frequency, equal width, MDL, entropy-based discretization, discretization-based ensemble with LR meta model, and discretization-based ensemble with RF meta model, respectively, showcasing the performance of models in a clean environment.

Table 3. Models' performance in clean environment

Model	original	EF	EW	MDL	EBD	EN-LR	EN-RF
DT	99.41	50.67	93.61	84.15	85.42	96.27	96.30
RF	99.90	50.91	99.71	99.88	99.80	99.70	99.71
LR	92.99	32.28	99.68	99.88	99.82	99.67	99.59
FF	92.07	50.26	99.21	99.80	99.71	99.60	99.55

Table 4. Black box attacks

Model	Black box attack	EW		MDL		EBD		No Discretisation	
		Adv_acc	RM	Adv_acc	RM	Adv_acc	RM	Adv_acc	RM
Decision Tree	FGSM LR	86.84	7.23	74.51	11.46	73.49	13.97	75.78	23.77
	JSMA LR	85.36	8.81	78.35	6.89	78.41	8.21	75.67	23.88
	BIM LR	87.10	6.95	74.53	11.43	73.69	13.73	75.86	23.69
	JSMA FF	81.34	13.11	71.18	15.41	74.06	13.30	74.37	25.19
	FGSM FF	87.27	6.77	74.35	11.65	75.20	11.96	78.11	21.43
	BIM FF	86.28	7.83	71.55	14.97	73.75	13.66	76.61	22.94

(*continued*)

Table 4. (*continued*)

Model	Black box attack	EW		MDL		EBD		No Discretisation	
		Adv_acc	RM	Adv_acc	RM	Adv_acc	RM	Adv_acc	RM
Random Forest	FGSM LR	96.87	2.85	93.83	6.06	94.91	4.90	64.82	35.12
	JSMA LR	98.57	1.14	98.49	1.39	98.69	1.11	73.92	26.01
	BIM LR	97.04	2.68	96.34	3.54	96.36	3.45	66.11	33.82
	JSMA FF	97.18	2.54	94.77	5.12	96.22	3.59	71.52	28.41
	FGSM FF	96.24	3.48	97.91	1.97	96.75	3.06	73.88	26.05
	BIM FF	96.42	3.30	94.58	5.31	93.25	6.56	70.44	29.49
LR	JSMA FF	95.91	3.78	92.91	6.98	92.89	6.94	65.21	29.87
	FGSM FF	94.77	4.93	96.69	3.19	95.20	4.63	78.13	15.98
	BIM FF	95.24	4.45	93.72	6.17	92.29	7.54	73.83	20.60
FF	FGSM LR	92.54	3.81	91.97	7.85	87.57	12.18	79.42	13.74
	JSMA LR	90.97	5.45	96.36	3.45	92.07	7.66	65.76	28.58
	BIM LR	92.52	3.84	91.95	7.87	87.20	12.55	78.50	14.74

It is apparent that the EF discretization method has led to a significant decrease in the accuracy of the models. The original models have high accuracy, while the model built using the EF discretization method has much lower accuracy. This indicates that the discretization process has negatively impacted the ability of the models to accurately predict outcomes. This is mainly due to the distribution of the data, as the model cannot discretize the data into bins with equal frequency when the frequency in some bins is very high. The maximum number of bins for equal frequency is 17, and the model is unable to increase this limit. Given the poor performance of EF approach in a clean environment, we will not evaluate its robustness capability in the next steps. However, the results indicate that the remaining methods, namely EW, MDL, EBD, EN-LR, and EN-RF accurately identify IoT devices in a clean environment. Consequently, we will evaluate the performance of these models in an adversarial environment. Table 4 presents the accuracy of different discretized models when subjected to different black box attacks.

while RM is the robustness measure. Most of the discretized models exhibit a high level of adversarial robustness, as evidenced by their accuracy in an adversarial environment, compared to their lower accuracy when no discretization mechanism is used. EW is proving to be a robust technique for all classifiers, as it achieves high accuracy in adversarial environments and exhibits less degradation in accuracy when exposed to adversarial attacks (low RM measure). For example, in the Decision Tree model, the Adv_acc for FGSM LR attack is 86.84% in the EW model, whereas it is only 75.78% without discretization. Similarly, in the Random Forest model, the Adv_acc for JSMA LR attack is 98.57% in the EW model, whereas it is only 73.92% in the no discretization model. This can be attributed to its ability to handle data with varying distributions and to its effectiveness in reducing the impact of outliers. EW discretizes the variable into a smaller number of categories; hence, the noise and variability can be reduced, and the model can learn more general and robust patterns. Additionally, equal-width discretization is a simple and easy-to-implement technique that requires minimal computational resources. Overall, the strong performance of equal-width discretization suggests that it is a promising approach for achieving adversarial robustness in this domain.

Table 5. White box attacks

Model	White box attack	EW		MDL		EBD		No Discretisation	
		Adv_acc	RM	Adv_acc	RM	Adv_acc	RM	Adv_acc	RM
LR	FGSM	94.79	4.91	96.50	3.38	94.44	5.39	60.10	35.37
	JSMA	96.50	3.19	98.22	1.66	97.20	2.62	64.82	30.29
	BIM	94.83	4.87	96.46	3.42	94.32	5.51	64.98	30.12
FF	FGSM	93.30	5.96	93.01	6.80	90.39	9.35	76.00	17.45
	JSMA	88.12	11.18	93.17	6.64	85.24	14.51	62.29	32.34
	BIM	94.89	4.35	91.23	8.59	87.84	11.90	80.29	12.79

Results demonstrate the effectiveness of the MDL methodology in increasing the resilience of random forest, feedforward, and logistic regression models to the black box. The MDL methodology achieves this by minimizing the description length of the model, which encourages the selection of simpler and more interpretable models that are less prone to overfitting. In addition to improving model robustness, the MDL methodology provides a principled approach for selecting the best model among a set of candidates based on the tradeoff between model accuracy and complexity. Overall, the MDL methodology is a promising approach for achieving robustness in IoT device classification. The EBD discretization method has emerged as an effective way to increase the robustness of machine learning models to black box attacks, except for the decision tree model. For example, in the case of the random forest model, the EBD approach improved the adversarial accuracy against the JSMA LR attack from 73.92% to 98.69% and against the FGSM FF attack from 73.88% to 96.75%. Overall, the effectiveness of the EBD approach in improving the robustness of models against black-box attacks

appears to be model-dependent. Further analysis to understand the factors that affect the effectiveness of EBD will be left for future research.

Table 5 illustrates the adversarial robustness of discretized models to white-box attacks. All classifiers without input discretization are vulnerable to such attacks. In contrast, IoT device classifiers trained on discretized input data generated by EW, MDL, EBD show less vulnerability to adversarial attacks as their Adv_acc is much greater than that of models without discretization. Also, discretized models present lower RM compared to models with no discretization, which confirms the effectiveness of input discretization with EW, MDL, EBD for creating a robust IoT device identification model in a white box scenario.

4.4 Discretization-Based Ensemble Defence Mechanism

Table 6. Discretization based ensemble-black box attacks

Model	Black box attack	EN-LR		EN-RF		Original model (no robustness)	
		Adv_acc	RM	Adv_acc	RM	Adv_acc	RM
Decision Tree	FGSM LR	91.70	4.75	91.86	4.61	75.78	23.77
	JSMA LR	95.13	1.18	95.83	0.49	75.67	23.88
	BIM LR	93.07	3.32	91.02	5.48	75.86	23.69
	JSMA FF	91.15	5.32	92.54	3.90	74.37	25.19
	FGSM FF	94.50	1.84	92.70	3.74	78.11	21.43
	BIM FF	92.80	3.60	90.17	6.37	76.61	22.93
Random Forest	FGSM LR	97.77	1.94	97.34	2.38	64.82	35.12
	JSMA LR	99.36	0.34	99.59	0.12	73.92	26.01
	BIM LR	97.58	2.13	97.54	2.18	66.11	33.83
	JSMA FF	97.52	2.19	97.28	2.44	71.52	28.40
	FGSM FF	98.20	1.50	98.16	1.55	73.88	26.05
	BIM FF	96.58	3.13	96.55	3.17	70.44	29.49
LR	JSMA FF	95.95	3.73	96.71	2.89	65.21	29.88
	FGSM FF	97.89	1.79	96.49	3.11	78.13	15.98
	BIM FF	96.40	3.28	96.54	3.06	73.83	20.60
FF	FGSM LR	96.03	3.58	95.68	3.89	79.42	13.74
	JSMA LR	96.40	3.21	96.18	3.39	65.76	28.58
	BIM LR	96.39	3.22	95.98	3.59	78.50	14.74

We evaluated the effectiveness of the proposed discretization-based ensemble robustness methodology in enhancing the robustness of IoT device classifiers against adversarial

attack. Two discretisation-based ensembles of EN-LR, EN-RF were built in this study. The ensemble method employed was a stacking method using the Scikit-learn Python library. We achieved this through the aggregation of individually trained models, each employing distinct discretization strategies on input features, resulting in diverse base classifiers for the same IoT device classification; these base models were then used to build a stack ensemble, ultimately yielding a robust classifier tailored to the base IoT device classifier. The main difference between EN-LR and EN-RF is their choice of meta-model. EN-LR uses LR as its meta model, while EN-RF uses RF as its meta-model. For example, to build EN-LR for DT model, The DT with EW discretisation, DT with MDL, and DT with EBD were used as base learners and LR as a meta-model.

Table 6, and Table 7 illustrate the performance of our models in black-box and white-box scenarios, respectively. Each table shows the adversarial accuracy and robustness measures of the models against different adversarial attacks EN-LR, EN-RF, and original models without robustness methodology. Looking at Table 6, we can see that across various black box attacks, both EN-LR and EN-RF consistently outperform the original models in terms of adversarial accuracy. For instance, within the context of DT model, the application of the EN-LR discretization-based ensemble robustness methodology yields a remarkable improvement in model accuracy, achieving accuracy of 91.70% against the FGSM-LR adversarial attack. This achievement is particularly noteworthy when compared to the original DT model's accuracy of 75.78% when exposed to the same adversarial attack. Furthermore, the robustness measures (RM) for EN-LR and EN-RF are generally lower than those of the original models. For instance, analysing the RF classifier robustness to JSMA-LR black box attack, the EN-LR robustness method achieves an RM of 0.34%, EN-RF an RM of 0.12%, whereas the original model's RM is significantly higher at 26.01%. Both robustness approaches exhibit notable improvements in adversarial accuracy and reduced robustness measures. Table 7 reveals insights into the effectiveness of EN-LR, EN-RF discretization based ensemble approaches in enhancing the robustness of the models against white box attacks. The EN-LR and EN-RF models consistently demonstrate higher adversarial accuracy compared to the original models across various white box attacks. For instance, LR model using EN-LR robustness methodology achieves an accuracy of 96.08% against FGSM white box

Table 7. Discretization based ensemble - white box attacks

Model	Black box attack	EN-LR		EN-RF		Original model (no robustness)	
		Adv_acc	RM	Adv_Acc	RM	Acc_Adv	RM
LR	FGSM	96.51%	3.17%	96.18%	3.42%	60.10%	35.37%
	JSMA	98.71%	0.96%	98.94%	0.65%	64.82%	30.29%
	BIM	96.39%	3.29%	95.13%	4.48%	64.98%	30.12%
FF	JSMA	94.62%	5.00%	94.28%	5.29%	62.29%	32.34%
	FGSM	93.50%	6.12%	93.42%	6.16%	76.00%	17.45%
	BIM	94.39%	5.23%	95.23%	4.34%	80.29%	12.79%

attack, outperforming the original model's accuracy of 60% for same attack. It clearly highlights the improvement achieved by the EN-LR approach.

5 Conclusion

In this paper, we introduced a discretization-based stack ensemble methodology to enhance the resilience of machine learning-driven IoT device identification models against adversarial attacks. Our findings highlight the efficacy of discretization as a strategy to improve model robustness across white and black box attack scenarios. Our research's generality and adaptability make it applicable to diverse IoT datasets, increasing its relevance in cybersecurity. In the future, we will delve into the practical implementation of our approach within a new domain. By exploring the application of our method in other contexts, we aim to solidify the real-world utility further.

References

1. Al-Garadi, M.A., Mohamed, A., Al-Ali, A.K., Du, X., Ali, I., Guizani, M.: A survey of machine and deep learning methods for internet of things (IoT) security. IEEE Commun. Surv. Tutor. **22**(3), 1646–1685 (2020)
2. Cvitić, I., Peraković, D., Periša, M., Gupta, B.: Ensemble machine learning approach for classification of IoT devices in smart home. Int. J. Mach. Learn. Cybern. **12**, 1–24 (2021)
3. Akhtar, N., Mian, A.: Threat of adversarial attacks on deep learning in computer vision: a survey. IEEE Access **6**, 14410–14430 (2018). https://doi.org/10.1109/ACCESS.2018.2807385
4. Yuan, X., He, P., Zhu, Q., Li, X.: Adversarial examples: Attacks and defenses for deep learning. IEEE Trans. Neural Netw. Learn. Syst. **30**(9), 2805–2824 (2019). https://doi.org/10.1109/TNNLS.2018.2886017
5. Namvar, A., Thapa, C., Kanhere, S.S., Camtepe, S.: Evaluating the security of machine learning based IoT device identification systems against adversarial examples. In: Hacid, H., Kao, O., Mecella, M., Moha, N., Paik, H. (eds.) Service-Oriented Computing: 19th International Conference, ICSOC 2021, Virtual Event, November 22–25, 2021, Proceedings, pp. 800–810. Springer International Publishing, Cham (2021). https://doi.org/10.1007/978-3-030-91431-8_57
6. Singh, A., Sikdar, B.: Adversarial attack for deep learning based IoT appliance classification techniques. In: 2021 IEEE 7th World Forum on Internet of Things (WF-IoT) , pp. 657–662. IEEE (2021)
7. Meidan, Y., et al.: Detection of unauthorized IoT devices using machine learning techniques. arXiv preprint (2017)
8. Kotak, J., Elovici, Y.: Adversarial attacks against IoT identification systems. IEEE Internet Things J. **10**(9), 7868–7883 (2023). https://doi.org/10.1109/JIOT.2022.3229906
9. Brownlee, J.: Data preparation for machine learning: data cleaning, feature selection, and data transforms in Python. In: Machine Learning Mastery (2020)
10. Buckman, J., Roy, A., Raffel, C., Goodfellow, I.: thermometer encoding: one hot way to resist adversarial examples. In: International Conference on Learning Representations (2018)

11. Sharmin, S., Rathi, N., Panda, P., Roy, K.: Inherent adversarial robustness of deep spiking neural networks: effects of discrete input encoding and non-linear activations. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX, pp. 399–414. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-58526-6_24
12. Panda, P., Chakraborty, I., Roy, K.: Discretization based solutions for secure machine learning against adversarial attacks. IEEE Access **7**, 70157–70168 (2019)
13. Lal, S., et al.: Adversarial attack and defence through adversarial training and feature fusion for diabetic retinopathy recognition. Sensors **21**(11), 3922 (2021)
14. Kireev, K., Kulynych, B., Troncoso, C.: Adversarial robustness for tabular data through cost and utility awareness. arXiv preprint arXiv:2208.13058 (2022)
15. Maslove, D.M., Podchiyska, T., Lowe, H.J.: Discretization of continuous features in clinical datasets. J. Am. Med. Inform. Assoc. **20**(3), 544–553 (2013)
16. Zhou, J., Zaidi, N., Zhang, Y., Li, G.: Discretization inspired defence algorithm against adversarial attacks on tabular data. In: Gama, J., Li, T., Yang, Y., Enhong Chen, Y., Zheng, F.T. (eds.) Advances in Knowledge Discovery and Data Mining: 26th Pacific-Asia Conference, PAKDD 2022, Chengdu, China, May 16–19, 2022, Proceedings, Part II, pp. 367–379. Springer International Publishing, Cham (2022). https://doi.org/10.1007/978-3-031-05936-0_29
17. Kurakin, A., et al.: Adversarial attacks and defences competition. In: Escalera, S., Weimer, M. (eds.) The NIPS '17 Competition: Building Intelligent Systems, pp. 195–231. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-94042-7_11
18. Mohanty, H., Roudsari, A.H., Lashkari, A.H.: Robust stacking ensemble model for darknet traffic classification under adversarial settings. Comput. Secur. **120**, 102830 (2022)
19. Li, D., Li, Q.: Adversarial deep ensemble: evasion attacks and defenses for malware detection. arXiv preprint arXiv:2006.16545 (2020)
20. Sivanathan, A., et al.: Classifying IoT devices in smart environments using network traffic characteristics. IEEE Trans. Mob. Comput. **18**, 1745–1759 (2018)
21. Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z.B., Swami, A.: The limitations of deep learning in adversarial settings. In: 2016 IEEE European Symposium on Security and Privacy (EuroS&P) , pp. 372–387. IEEE (2016)
22. Papernot, N., McDaniel, P., Goodfellow, I.: Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. arXiv preprint (2016)
23. Nicolae, M.-I., et al.: Adversarial Robustness Toolbox v1. 0.0. arXiv preprint arXiv:1807.01069 (2018)

Flute: Enabling a Battery-Free and Energy Harvesting Ecosystem for the Internet of Things

Van Vu Bui[1,2(✉)], Shuaibu Musa Adam[1], Brendan J. Mackenzie[1], Ashok Samraj Thangarajan[1], Mengyao Liu[1], Sam Michiels[1], Nelson Souto Rosa[3], Huynh Nguyen Bao Phuong[2], and Danny Hughes[1]

[1] DistriNet, KU Leuven, 3001 Leuven, Belgium
[2] Department of Engineering and Technology, Quy Nhon University, Quy Nhon, Vietnam
vanvu.bui@kuleuven.be
[3] Centro de Informatica, UFPE, Recife, Brazil

Abstract. Nowadays, most Internet of Things (IoT) devices still operate on batteries, offering lifetimes of just a few years; leading to high maintenance costs and excess waste. Energy Harvesting (EH) in combination with super-capacitor charge storage offers a solution, however, it is considered complex and unreliable, hampering its real-world deployment. To address this problem, we contribute *Flute*, a power management system for building battery-free IoT deployments. Unlike prior work, Flute focuses on transforming existing IoT development boards into battery-free devices. Reliable operation is achieved through a combination of generic power management hardware and self-adaptive software enabling sustainable operation in the face of dynamic power supply and demand. We evaluate our approach through a series of diverse one-week solar EH scenarios and show that Flute efficiently harvests energy and achieves reliable operation for a range of unmodified *IoT development boards* using LTE-M and LoRa. Hence, Flute provides a promising foundation for building a battery-free and EH ecosystem for the Internet of Things.

Keywords: Self-adaptive power management · IoT development boards · Energy harvesting · Battery-free · Plug-and-play power supply

1 Introduction

IoT research aims to create efficient and performant technologies through which embedded devices can communicate, sense and actuate. Two decades of research have resulted in the creation of specialized hardware [2,4,6,8], Operating Systems (OS) [9] and network protocols (See footnote 5) allowing low-cost wireless Micro Controller Units (MCUs) to function as first-class Internet devices.

However, the current ecosystem of IoT Development Boards (dev-boards) has yet to assimilate recent research results in *battery-free and Energy Harvesting [EH] operation* [9]. We address this gap by introducing Flute, a Hardware

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 368–380, 2024.
https://doi.org/10.1007/978-3-031-63992-0_24

(HW) and Software (SW) platform for building battery-free and EH IoT systems. Unlike other technologies, Flute does not offer an integrated solution [5]. It converts many popular IoT boards into battery-free EH devices. Specifically, Flute delivers: super-capacitor charge storage, solar EH and self-adaptive energy management. The Flute hardware is a Plug-and-Play (PnP) power supply and management board for Adafruit Feather format devices. This is complemented by self-adaptive scheduling algorithms[1] which integrates with standard development environments to ensure sustainable operation in the face of dynamic energy supply and demand. We evaluate Flute through a real-world solar EH deployment, using unmodified dev-boards with representative processors (Cortex M0 and Cortex-M33) and diverse networks (LoRa and LTE-M). Our results show that Flute ensures sustainable operation under diverse conditions while optimally exploiting available environmental energy.

The scientific contributions of Flute are: (i.) The first PnP power board for commodity IoT platforms that enable battery-free and EH operation; (ii.) Effective energy management algorithms that adapt to changing energy supply and demand; and (iii.) a real-world evaluation in a variety of representative environmental settings.

The remainder of this paper is organised as follows. Related work and requirements are presented in Sect. 2. The design and implementation of the complete solution are presented in Sects. 3 and 4. Section 5 presents the evaluation of Flute. We conclude in Sect. 6 and discuss future work.

2 Related Work

Battery-free EH IoT platforms use various energy sources, and store charge in capacitors, offering longer operating life than rechargeable batteries. As capacitors energy density is lower than batteries, research has emerged on intermittent computing [1,2,5], isolated energy storage architectures [4,8] and OS support for energy awareness [9,10]. We briefly review each stream of research below.

2.1 Intermittent Computing

Afanasov et al. [1] use an EH system to monitor underground chambers of the Circus Maximus. They identify dynamic energy availability as a key challenge and achieve a maximum of 22% availability over three design iterations. To maximise availability, the authors argue for careful energy bench-marking and eliminating dynamic power consumption.

The 'Battery-free Game Boy' [2] applies hardware [4,8] and OS innovations [9,10] to realise a battery-free hand-held gaming console. However, it provides no communication capabilities and requires consistent sunlight for continuous use, applying a range of intermittent computing techniques when power fails.

BFree is a HW/SW platform enabling battery-free applications development in Python [5]. *BFree* offers a power failure-resilient architecture that makes it

[1] GitHub: Flute hardware & software.

easy for novice developers to use. While simplifying power failure management, it does not ensure continued operation with changing conditions as it relies solely on solar harvesting.

Although these techniques share goals with Flute, such as shielding developers from power failure complexities, the approaches are fundamentally different. Intermittent computing accepts power failure as inevitable and contributes techniques to handle it, while Flute aims to avoid power failures by scheduling software within the available power budget. Additionally, *BFree* [5] and the *Battery-Free Game Boy* [2] are integrated platforms that do not support third-party development boards. In contrast, *Flute* supports a wide range of networks, processors and sensors by separating power concerns from application development.

2.2 Isolated Charge Storage

UFoP [3] is a federated energy storage architecture for intermittent sensors that support variable supply voltage, unpredictable power outages, and uncertain EH performance. It simplifies development by segmenting charge storage for specific peripherals while preventing power-hungry elements from consuming all available energy, rendering the system unusable.

Morphy is a software-defined charge storage architecture for the IoT [8] that uses a re-configurable array of capacitors. Like UFoP, it enables physical charge isolation for specific tasks. It also allows re-parameterization of the capacitor array from series to parallel, enabling rapid wake-up from low voltage sources and extended operation during energy droughts. Morphy also contributes an *energy watchdog*, allowing applications to detect and handle power failures.

This stream of research demonstrates the importance of safeguarding charge for specific tasks, a goal which is shared by Flute. Inspired by *Morphy* [8], Flute also cleanly separates power management hardware from the dev-board.

2.3 OS Support for Energy Awareness

AsTAR is a battery-free IoT platform with prioritised energy-aware scheduler that matches energy demands to available supply [9] and thereby ensures sustainable operation. In contrast to earlier work, AsTAR achieves sustainable operation without prior benchmarking. Evaluation shows that AsTAR efficiently uses all available energy sustainably.

HyPer [6] extends AsTAR with multi-stage energy storage, including a backup battery and super-capacitors for EH storage. It adapts task execution rates to guarantee a specified lifetime while optimally exploiting all available energy. This enables reliability even with frequent energy droughts.

Flute builds on the AsTAR [9] task scheduler, extended to support diurnal operation, dramatically improving schedule consistency in the case of outdoor solar deployments. Going beyond AsTAR, Flute supports a wide range of 3rd-party boards. In the following section, we distill technical requirements for the design of Flute.

2.4 Requirements

We define the following requirements for the design of *Flute*:

1. ***Low power duty cycling:*** Sustainable EH requires aggressive duty cycling. However, the power efficiency of IoT development boards varies dramatically. Flute addresses this problem via a power board that allows the host board to be powered down or placed into deep sleep mode between executions.

2. ***Adaptive task scheduling:*** Intelligent scheduling is required [8,9] to tailor application energy demands to match available energy supply. However, additional support is required to enable more consistent scheduling in the case of diurnal EH patterns. Flute achieves this by extending the AsTAR [9] algorithm.

3. ***PnP operation:*** Flute will support the existing ecosystem of IoT development boards by offering the first PnP battery-free and EH solution for the boards, specifically focusing on the Adafruit Feather due to its extensive range and popularity.

4. ***Developer support:*** Flute contributes energy management libraries that simplify task scheduling, time management, and handling power faults. Reference implementations are available for the Arduino and Zephyr[2] development environments. These goals are in line with those of BFree [5].

3 The Design of Flute

The *hardware* goal of Flute is to convert commodity IoT development boards into battery-free EH devices. Flute's *software* goal is to schedule tasks: *(i.)* as frequently as possible, while *(ii.)* accumulating an optimum charge level and *(iii.)* preserving system operation by throttling execution when environmental energy is lost. This requires a PnP power supply board and software support to manage energy usage. These are described in Sects. 3.1 and 3.2, respectively.

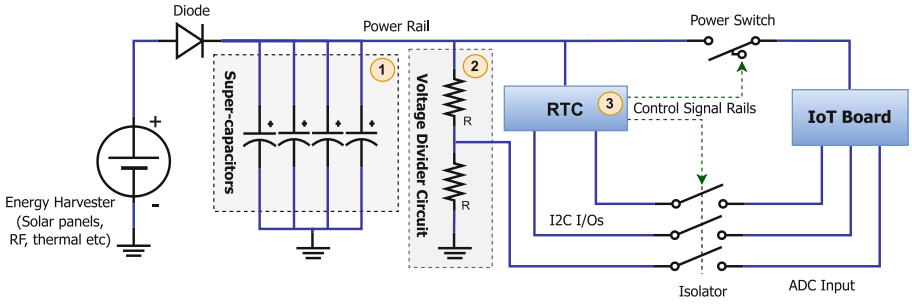


Fig. 1. Block diagram of the Flute PnP power board

[2] Zephyr development tools and IDEs.

3.1 Plug-and-Play Power Board

The primary functions of the Flute PnP power board are: charge storage, monitoring and power control. Figure 1 provides a high-level block diagram of the power board and each block is then explained in the text.

Charge Storage: Flute's charge storage (1) uses an array of up to four reconfigurable super-capacitors. As Flute aims to ensure sustainable operation, these capacitors should be sized to power the device at an acceptable duty cycle during the longest expected *energy droughts* (i.e. periods with no environmental energy available). Capacitors with minimal leakage current and low Equivalent Series Resistance (ESR) are preferred.

Charge Monitoring: Flute uses the ADC of the host development board to monitor the capacitor voltage via a 50% voltage divider circuit (2). Large-value resistors are selected to minimize the power consumption of this circuit element. The remaining useful charge is calculated based on the array size, voltage level and brownout voltage.

Power Control: Most development boards using Flute have onboard regulators with different specifications and efficiencies. To address this heterogeneity, Flute supports two power management strategies. First, an efficient Real Time Clock (RTC) (3) provides a common sleep timer with power consumption under 50nA. The sleep timer can be used in two ways. For boards with acceptable sleep

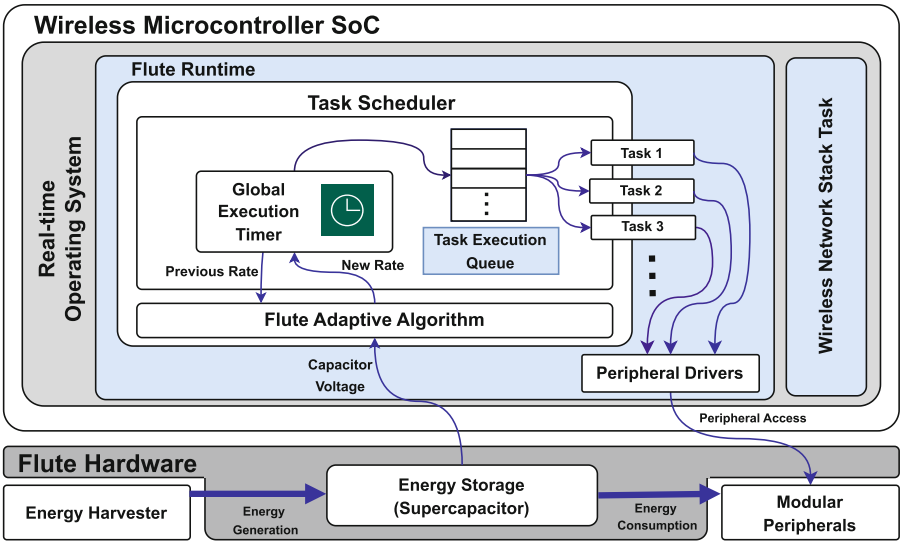


Fig. 2. The Flute Hardware/Software Architecture

power, the MCU can enter deep sleep and the RTC is used to wake it based on an external interrupt. For boards with prohibitive sleep power, the RTC can use its digital power switch to completely power down the board during sleep, mitigating design inefficiencies. To prevent power leakage, I2C and ADC lines are also isolated during power down. The output of the capacitors is regulated using the onboard regulator.

3.2 Energy Aware Software Support

The Flute software stack consists of a generic API and an energy-aware scheduler, as described in the following subsections and shown in Fig. 2.

Flute API: Flute aims not to modify how developers structure their code but to provide simple hooks to schedule tasks, handle energy events and inspect the system's state. The Flute library must first be initialized with suitable parameters to support the selected capacitors and development board.

```
void begin(uint16_t maxV, uint16_t minV, uint16_t maxRate,
    uint16_t minRate, bool powerDown);
```

Where *maxV* and *minV* are the maximum and minimum voltages supported by the board's regulator. *maxRate* and *minRate* are the minimum and maximum task rates (per hour) that are acceptable to the application. Finally, *powerDown* determines if the board is off or in sleep mode during idle scheduler state. In the latter case, the RTC provides a wakeup interrupt. To be scheduled by Flute, a function must observe the following signature:

```
void energyTask(uint32_t currentTime);
```

Flute functions accept one argument, *currentTime*, the RTC time at the moment is called in standard Unix format. A function may be attached to the Flute scheduler as follows:

```
void schedule(*energyTask, uint8_t dutyCycle);
```

Where **energyTask* is a pointer to the scheduled function and *dutyCycle* is how many ticks to wait before calling the function. A *dutyCycle* of 1 will execute the function every time the scheduler ticks. A *dutyCycle* of 10 executes at 1/10th of the tick rate, etc. Flute also provides the following utility functions:

```
void      setTime(uint32_t unixTime);
uint32_t  getTime();
void      sleep();
uint16_t  getSleepPeriod();
uint16_t  getCapacitorVoltage();
```

setTime() enables the application to set a UNIX format time and date in the RTC. *getTime()* gets the time from the RTC. *sleep()* provides a platform-specific deep sleep implementation with wake-on RTC interrupt. *getSleepPeriod()* allows developers to see and reason over the current schedule. Finally, *getCapacitorVoltage* returns voltage of the capacitor array, which is used to calculate available charge.

Energy Adaptive Algorithm: The proposed AsTAR++ algorithm is based on the original AsTAR shown in [7,9]. It manages energy droughts by tracking the duration of power outages using an Exponentially Weighted Moving Average. When an energy drought is detected based on 10 min of no charge gain (e.g. sunset), Flute changes strategies, aiming to schedule tasks at such a rate that 90% of the available charge is used by the expected end of the energy drought (e.g. sunrise). Once charge accumulation resumes, Flute reverts to the standard, conservative, AsTAR strategy. In this way, available energy is more fully exploited overnight.

4 Implementation

We implemented a Flute's prototype for evaluation and to provide an open-source reference design. The hardware details are in Sect. 4.1 while the software is described in Sect. 4.2. Upon publication, all designs, source code and materials will be made open-source.

Listing 1.1. Energy drought listing

```
getAdaptedRate(V_New, V_Prev) {
    IF (V_New - V_Prev) < 0 for last 10 minutes AND V_New <=
        V_Day_Optimum
        Night_Mode = True
    ELSE IF (V_New - V_Prev) > 0
        Night_Mode = False
    IF Night_Mode
       V_Optimum = V_Day_Optimum - (T_Since_Sunset *
          V_Swing_Night / Length_Night_EWMA)
    IF V_New < V_Min
        T_Sleep = Max_Sleep
    ELSE IF V_New > V_Max
        T_Sleep = Min_Sleep
    ELSE IF V_New < V_Optimum
        IF (V_New - V_Prev) < 0
            T_Sleep = T_Sleep * 5
        ELSE
            Sleep_Delta = T_Sleep / 10
            IF Sleep_Delta < 1
                Sleep_Delta = 1
            T_Sleep = T_Sleep - Sleep_Delta
    ELSE IF V_New >= V_Optimum
        IF (V_New - V_Prev) < 0
            Sleep_Delta = T_Sleep / 10
            IF Sleep_Delta < 1
                Sleep_Delta = 1
            IF Night_Mode
                T_Sleep = T_Sleep - Sleep_Delta
            ELSE  T_Sleep = T_Sleep + Sleep_Delta
        ELSE  T_Sleep = T_Sleep/5
    IF T_Sleep < Min_Sleep
        T_Sleep = Min_Sleep
```

```
    IF Night_Mode AND T_Sleep < Min_Night_Sleep
        T_Sleep = Min_Night_Sleep
    RETURN T_Sleep
}
```

4.1 Hardware Implementation

The Flute power board uses a standard connector for AdaFruit Feather format development boards. All boards are powered by four super-capacitors,[3] selected based on high energy density and low ESR. Energy is harvested using two monocrystalline solar panels in parallel[4]. Nodes are enclosed in waterproof cases for indoor and outdoor tests, as shown in Fig. 3.

Fig. 3. Prototype Flute board (left) and finished battery-free EH nodes (right)

The Flute power board was integrated with two representative dev-boards with varied features and power needs as summarized in Git repository(See footnote 1). As shown in our evaluation, Flute manages this heterogeneity without prior benchmarking.

4.2 Software Implementation

We implemented the libraries described in Sect. 3 for Zephyr OS and Arduino. We employed long range networking, 433MHz LoRa and LTE-M, parameterized for low power and extended coverage. The Flute software stack does not constrain developers in configuring their network. LTE-M is a low-power cellular technology that reduces power through local Power Saving Mode (PSM) or extended Discontinuous Reception[5]. With no downlink for this application, the

[3] 5.5V/15F AVX Series-Connected Super Capacitor Modules.
[4] Seeed Studio 0.5W monocrystalline silicon solar panel.
[5] Long Term Evolution for Machines: LTE-M.

PSM was set longer than the sleep time, ensuring immediate return after sending data over UDP.

AsTAR and AsTAR++ use common 1 s *daytimeMaxRate* and 7200 s (2 h) *minRate*. We also set *minV, maxV and optimumV* as shown in the hardware specification Table. The *nightMaxRate* balances sustaining overnight operation until daylight recharge, and keeping voltage high enough to detect dawn and subsequent modes switching. The tuned *nightMaxRate* values for each condition are shown in Table 1. Future hardware revision will directly detect dawn by open-circuiting the solar panel.

5 Evaluation

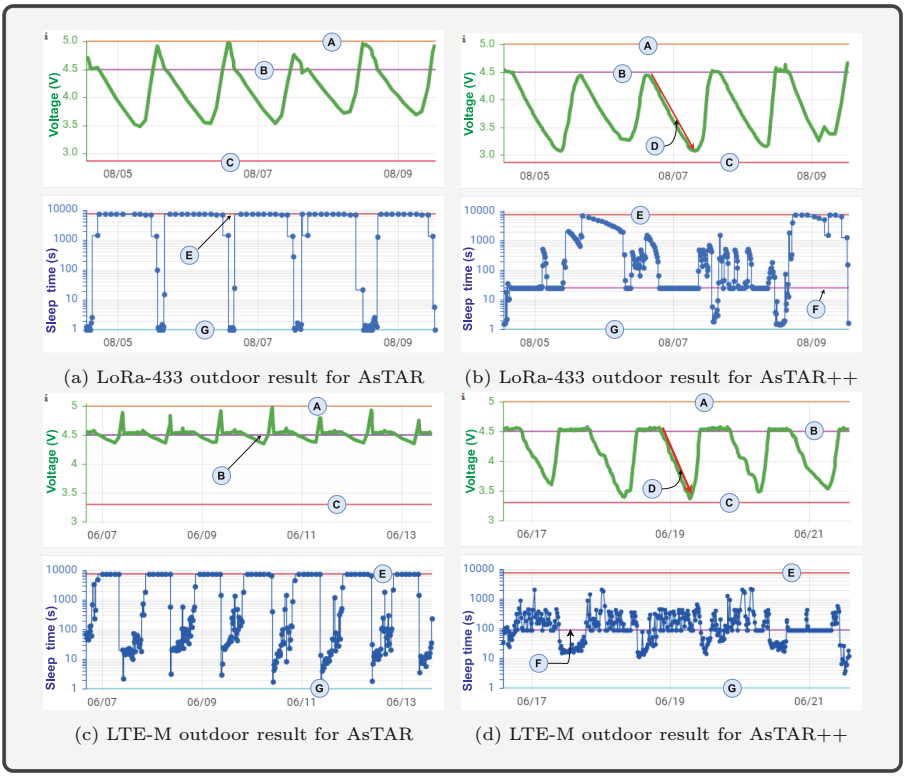
We evaluate the performance of Flute using two IoT daughter-boards described in Sect. 4 through a 7-day test deployed in *Vietnam* and *Belgium* under outdoor and indoor conditions. The boards were connected and ran the Flute software locally. Each of the 7-day experiment was performed at a different time, therefore absolute results of the graphs cannot be directly compared due to changing energy availability. However, the relative trends remain clear. From the results shown in Figs. 4 and 5, label *(A)* denotes the *maximumVoltage*, *(B) optimumVoltage*, *(C) shutOffVoltage* plus a 10% safety margin and *(D) NightOptimumVoltage*, respectively. For the sleep time *(E)* represents *maximumSleepTime*, *(F) nightMinimumSleepTime* and *(G) minimumSleepTime*, respectively.

5.1 Outdoor Operation

Figure 4 shows outdoor performance of baseline AsTAR compared to AsTAR++ for each of the two boards. AsTAR ensures reliability through a conservative approach; throttling software to a minimum of one execution every two hours at night when harvesting energy is unavailable. Conversely, when the sun rises, AsTAR rapidly acquires an optimum charge level and commensurately increases task rates. This daily pattern repeats throughout the experiment. Analysis of the nodes' night-time behaviour reveals the AsTAR inefficiency; while nodes can safely operate down to 2.6V, the capacitor voltage never falls below 3.5V, leaving a large amount of unused charge. While this is reasonable with unpredictable EH scenarios, it is inefficient for diurnal scenarios.

Table 1. User-defined energy drought task rates for AsTAR++

Node	Deployment	nightMaxRate(s)
LoRa-433 (Vietnam)	Indoor	90
	Outdoor	25
LTE-M (Belgium)	Indoor	600
	Outdoor	90



(a) LoRa-433 outdoor result for AsTAR

(b) LoRa-433 outdoor result for AsTAR++

(c) LTE-M outdoor result for AsTAR

(d) LTE-M outdoor result for AsTAR++

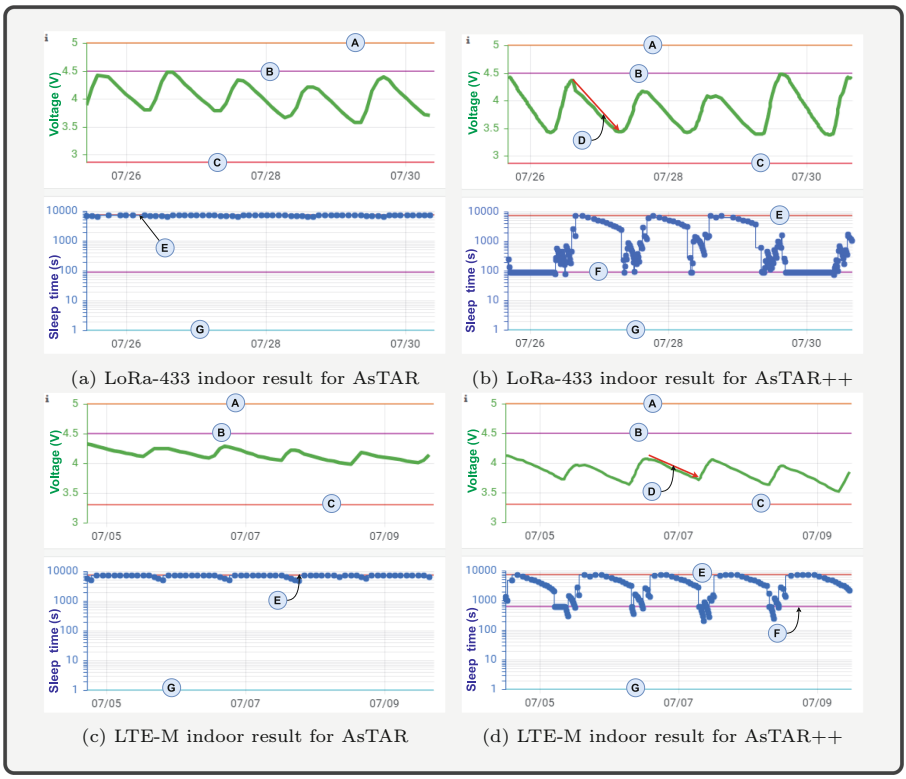
Fig. 4. Outdoor performance graphs for AsTAR and AsTAR++ algorithms

In contrast, AsTAR++ tracks the night-time energy droughts length to safely use most of the available charge overnight. In all cases AsTAR++ reaches approximately 1.1x the brownout voltage of the daughter-board (2.86 V for LoRa and 3.3 V for LTE-M). This way, it efficiently uses the daytime charge, increasing messaging rates while preserving reliability with changing energy availability.

5.2 Indoor Operation

Figure 5 shows Flute indoor performance across the two locations. The indoor nodes received sunlight for just a few hours daily. Specifically, the nodes were deployed in summer of 2023 with many days of rain and inside of laboratories opening briefly without artificial lightening. Annotations *(A)* through *(G)* remain the same.

The graphs show that AsTAR and AsTAR++ algorithms ensure sustainable operation even with reduced amount of harvested solar energy when indoors. Additionally, AsTAR++ significantly outperforms AsTAR through more aggressive overnight energy use, *without* negatively impacting device availability.



(a) LoRa-433 indoor result for AsTAR

(b) LoRa-433 indoor result for AsTAR++

(c) LTE-M indoor result for AsTAR

(d) LTE-M indoor result for AsTAR++

Fig. 5. Indoor performance of AsTAR and AsTAR++ algorithms

As shown in Table 2, with AsTAR, LoRa nodes achieve an average *0.51* messages per hour indoor and *2.47* outdoor. AsTAR++ improves messaging rate by over 16x for indoor and 5x for outdoor, respectively. With AsTAR, LTE-M nodes achieve an average *0.53* messages per hour while indoor and *6.84* outdoor. Again, AsTAR++ improves messaging by over 1.7× for indoor and 2.2× for outdoor. Critically, Flute availability remains unaffected, delivering 100% using both algorithms. While the deployment environments evaluated are simpler than ones tackled by Afanasov et al. [1], our results demonstrate that in *certain scenarios* battery-free EH platforms can deliver very high levels of availability.

Table 2. 1-week Flute message rate metrics for AsTAR and AsTAR++ across two deployment locations

Node Type/Algorithm		AsTAR	AsTAR++	Improvement (%)
		Avg. messaging per hour	Avg. messaging per hour	Avg. messaging per hour
LoRa-433	Indoor	0.51	8.7	**1605.88**
	Outdoor	2.47	15.65	**533.6**
LTE-M	Indoor	0.53	1.45	**173.58**
	Outdoor	6.84	22.22	**224.85**

6 Conclusions

Contributions of this paper facilitate the vision of a battery-free EH IoT by enabling PnP conversion of commodity dev-boards into battery-free EH platforms. A week-long evaluation of our approach shows a promising performance profile. Specifically, Flute sustainably supports simple wireless sensing applications in both indoor and outdoor scenarios running on multiple networks and processors using solar power alone. Moreover, with diurnal operation, Flute outperforms the original AsTAR algorithm by a factor of 2x to 17x while always maintaining a 10% safety margin, well over the brown-out voltage.

7 Future Work

Our future work will focus on three scientific tracks: *(i.)* Improving self-adaptive energy management by broadening its scope beyond the application layer and integrating it into the OS and network. *(ii.)* Implementing approaches to simplify developer construction of complex multi-tasking schedules. *(iii.)* Broadening Flute evaluation to include more EH methods like thermal, RF and kinetic. Additionally, as an engineering track, we intend to establish large-scale Flute deployments in multiple locations for a year-long study across seasons.

Acknowledgements. This work is partially funded by VLIR-UOS (IUC-QNU/KU Leuven, VN2022IUC -044A101), the KU Leuven Research Fund (ReSOS, C3/20/014), and VLAIO (TRUSTI, HBC.2021.0742).

References

1. Afanasov, M., et al.: Battery-less zero-maintenance embedded sensing at the mithræum of circus maximus. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, NY, USA, 2020. Association for Computing Machinery
2. de Winkel, J., Kortbeek, V., Hester, J., Pawełczak, P.: Battery-free game boy. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. **4**(3), 1–34 (2020)

3. Hester, J., Sitanayah, L., Sorber, J.: Tragedy of the Coulombs: federating energy storage for tiny, intermittently-powered sensors. In: Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems, pp. 5-16, Association for Computing Machinery. NY, USA (2015)
4. Hester, J., Sorber, J.: Flicker: rapid prototyping for the batteryless Internet-of-Things. In: Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems. Association for Computing Machinery NY, USA (2017)
5. Kortbeek, V., Bakar, A., Cruz, S., Yildirim, K.S., Pawełczak, P., Hester, J.: BFree: enabling battery-free sensor prototyping with python. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. **4**(4) (2020)
6. Thangarajan, A.S., Yang, F., Joosen, W., Hughes, D.: Deterministic 40 year battery lifetime through a hybrid perpetual sensing platform (HyPer). In: Proceedings of the 10th International Conference on the Internet of Things. Association for Computing Machinery NY, USA (2020)
7. Yang, F., et al.: AsTAR: Sustainable Battery Free Energy Harvesting for Heterogeneous Platforms and Dynamic Environments. pp. 71–82. Junction Publishing. USA (2019)
8. Yang, F., Thangarajan, A.S., Michiels, S., Joosen, W., Hughes, D.: Morphy: software defined charge storage for the IoT. In: Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems, pp. 248–260. Association for Computing Machinery, NY, USA (2021)
9. Yang, F., Thangarajan, A.S., Ramachandran, G.S., Joosen, W., Hughes, D.: AsTAR: sustainable energy harvesting for the internet of things through adaptive task scheduling. ACM Trans. Sen. Netw. **18**(1) (2021)
10. Yıldırım, K.S., Majid, A.Y., Patoukas, D., Schaper, K., Pawelczak, P., Hester, J.: InK: reactive kernel for tiny batteryless sensors. In: Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems, pp. 41–53. Association for Computing Machinery NY, USA (2018)

Exploring Optimal Placement of Head-Based Hierarchical Marking Menus on Smartphones

Sabiha Tahsin Soha[1], Aritra Mazumder[1], Forhad Rabbi[1], Farida Chowdhury[1], and Khalad Hasan[2(✉)]

[1] Shahjalal University of Science and Technology, Sylhet, Bangladesh
{frabbi-cse,farida-cse}@sust.edu
[2] University of British Columbia, Kelowna, Canada
khalad.hasan@ubc.ca

Abstract. Studies have shown that hierarchical marking menus, which allow for the navigation of menu items through a hierarchy, are faster at accessing items compared to traditional menus. Though hierarchical marking menus have been studied in non-touch-based input methods for smart devices (such as gaze-based input with VR), little is known about optimal menu placement in hierarchical structures for smartphones. In this paper, we investigate the optimal placement of subsequent marking menus within a menu hierarchy utilizing head-based input on smartphones. We examine how to position a second menu by considering the distances and directions from the first menu while examining two selection techniques: Dwell and Border Crossing. Results from a user study involving menu item selection tasks reveal that users are faster at selecting items when the second menu partially overlaps the first menu and appears in the direction where the target in the first menu is located. Furthermore, results showed that while Dwell was slower than Border Crossing, it resulted in significantly fewer errors in selecting items in the hierarchical menu. Based on these findings, we provide design guidelines for placing menus in the hierarchical marking menu setting.

Keywords: Marking Menus · Hierarchical Menus · Smartphones

1 Introduction

Hierarchical marking menus present a second menu with items relevant to the selected item in the initial menu after users select an item in the initial menu. Prior studies showed that hierarchical marking menus are effective in expanding the number of menu options available to users by allowing them to select items from multiple submenus through the use of compound strokes with a cursor [11]. These menus have demonstrated the potential to decrease selection time when compared to conventional menus [5] and support the gradual transition from novices to experts [11]. Though prior work investigated factors that can affect

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 381–392, 2024.
https://doi.org/10.1007/978-3-031-63992-0_25

user performance such as the number of menus and items in each menu, little is known about the optimal menu placement of hierarchical marking menus.

Hierarchical marking menus have been implemented with both touch-based inputs [12,21] and non-touch-based inputs [17,23] on various devices. For example, touch-based marking menus have been used on desktops [9], and smartphones [21]. Prior work explored marking menus that are built for smartphones [7], using the smartphone screen for interaction in wearable devices [19], and using the touch facility to help learn mid-air gesture set [9]. However, little is known about the ideal placement of menus within a hierarchy.

Smartphones provide users with the ability to access information while on the move. As a result, people rely on smartphones in various situations, including when their hand is occupied with carrying objects. Previous studies indicated that touch input during such one-handed cases could have a negative impact on users' smartphone performance and experience [10] Consequently, researchers proposed alternative input methods, such as head-based input and eye-based input, for interacting with smartphones in such situations. In such a one-handed situation, people hold their smartphones in front of their faces to facilitate access to on-screen content. In this holding posture, the device's front camera can detect the user's face and track their head movements, which opens up possibilities for novel non-touch-based input for spatial interactions and user interfaces [24]. Despite previous research on marking menus with smartphones, there is still limited understanding regarding the optimal placement of hierarchical marking menus on smartphones for non-touch input such as head movement.

In this paper, we investigated the optimal placement of a subsequent marking menu within a hierarchical menu structure with head-based input on smartphones. Specifically, we studied menu placement techniques that involve positioning a subsequent menu by considering the distances and directions from the previous menu. Additionally, we explored the effectiveness of two selection techniques, Dwell and Border Crossing, within the context of a hierarchical menu structure. We conducted a user study where results showed that users could quickly and accurately select menu items when the second menu partially overlapped the first menu and was positioned in the direction of the target in the first menu. Furthermore, we observed that while Dwell was slower than Border Crossing, it resulted in significantly fewer errors when selecting items. Based on the results, we provided recommendations for the design and optimization of the placement and selection techniques that can significantly enhance users' experience accessing items on hierarchical menus.

2 Related Work

2.1 Marking Menu

The original marking menu, introduced by Kurtenbach and Buxton in 1993 [11] uses a "mark" to indicate the menu item to be selected. To choose an item, the user draws a mark through it. Researchers explored several extensions of the marking menu in the past. For instance, Wave menu [1] is designed to improve

the novice mode of the traditional hierarchical marking menu by facilitating the sub-menu inside the previous menu. The Flower menu [2] and the Zone and Polygon menu [25] leverage the marking menu to contain a larger number of items, which can be accessed by straight and curved gestures. OctoPocus [3] extends the idea of marking menus for learning command sets that use gesture sets, making it faster than a hierarchical marking menu.

2.2 Selection on Marking Menu

Prior work explored touch and non-touch *Selection Techniques* with marking menus. For instance, Francone et al. [7], Salkanovic et al. [26], Zheng et al. [21] leverage touch input on the screen to interact with marking menus. Researchers also explored extended-input capabilities to interact with marking menus. For instance, Lepinski et al. [12] proposed a technique that goes beyond a single-touch input, leveraging multi-touch input where five fingers are used to determine the command. Non-touch based input has also been investigated with marking menus. A popular selection method for any gaze-based navigation system is Dwell [18], where users need to keep the pointer (e.g., cursor) static for a certain amount of time. Border Crossing [23] selection is another technique that lets users select an item by crossing the border of an item in a pie menu. However, overshooting, false activation or incorrect selection can occur if they are not handled properly. To this end, we looked into the Dwell and Border Crossing *Selection Techniques* while keeping the marking menu size and depth constant.

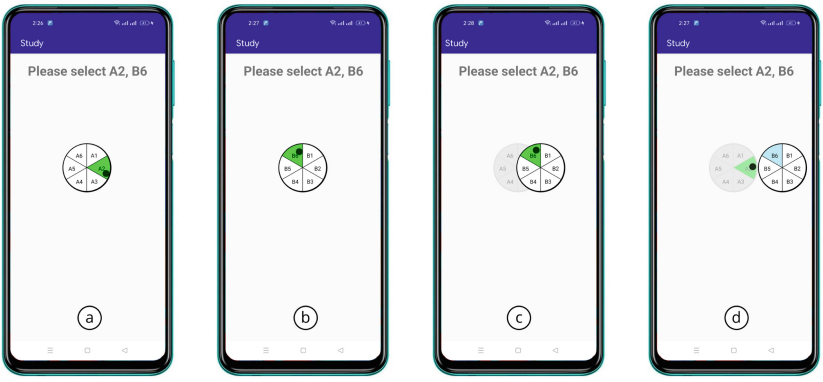


Fig. 1. (a) The cursor is moved on top of a target item in the marking menu. (b) In *In place* condition, the second menu appears on top of the first menu. However, in the (c) *Partial overlap* condition, the center of the second menu appears on the border of the first menu, thus partially overlapping the first menu. (d) In the *Disjoint* condition, the second menu appears in a location without overlapping the first menu.

2.3 Head Based Interaction

Head-based interaction has been explored in various fields. For instance, head-based controls have been studied as keyboard inputs for extremely small key sizes [8]. Gaze-based interactions, as well as gaze and head combined interactions, have also been explored [16]. With a study, Qian et al. [20] found that head-pointing worked faster than the combined eye and head input in wearable virtual reality. Prior work also explored head-based interaction for people with disabilities [14]. Although many studies have explored head gesture interaction techniques on smartphones, to the best of our knowledge, no study has investigated head-based interaction for marking menu navigation on smartphones. Therefore, in our work, we have explored head-based interaction to navigate marking menus.

3 Design Space

There are several factors that affect the placement of the subsequent menu within a hierarchical menu structure, which we discuss below:

3.1 Menu Distance

Distance refers to the relative position of the second menu to the first menu when the menus are displayed one after another. We discuss the possible *Distance* options below:

In-Place: The in-place menu placement style ensures that the second-depth menu appears exactly where the first menu appeared. We included this style in our study since no prior work had explored its use with head-based inputs. With this style, we anticipated that users would require less cursor movement to make selections as the second menu appears on top of the first menu (Fig. 1b).

Partial Overlap. The partial overlap style of menu placement is characterized by the appearance of the second-level menu in a way that partially overlaps with the first-level menu. Specifically, the center of the second menu appears on the circumference of the first menu, resulting in the second-level menu touching the center of the previous menu, as illustrated in Fig. 1c. We anticipate that this style minimizes the cursor movement between menus since the second-level menu overlaps with the prior menu.

Disjoint. The Disjoint menu placement style ensures that the second marking menu appears without overlapping the first menu. Here, the border of the second menu touches the border of the first menu (Fig. 1d). We anticipate that this style will help in reducing the error rate for the head-based input as well.

3.2 Menu Direction

Menu Direction refers to the positioning of the second menu in a diagonal direction from the first menu. We considered the following two menu directions:

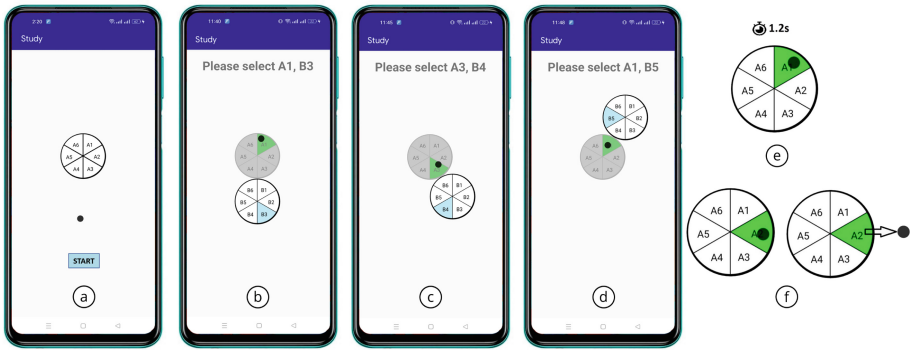


Fig. 2. (a) A trial begins when the start button is selected by moving the cursor on top of it. The fixed menu direction (b) shows the second menu in a specific direction from the first menu. The dynamic approaches (c-d) show the second menu appearing in a direction based on the previously selected item. In (c), the bottom-right item was selected in the first menu, and the second menu appeared in the same direction. In (d), the top-right item was selected in the first menu and the second menu was displayed in the corresponding direction. (e) Dwell selection technique, where the cursor is required to remain on a menu item for 1.2 s for selection, and (f) Border Crossing, where the cursor moves from inside the menu to the outside through a menu item for selection.

Fixed. In a fixed direction condition, the second menu is always displayed at a predetermined position, regardless of the selected item in the first menu. For our study, we placed the second menu at the bottom of the first menu for the fixed placement condition (Fig. 2b). However, the separation between the first and second menu displays is determined by the *Distance* condition.

Dynamic. In the dynamic direction, the second menu appears dynamically based on the item selected in the first menu. In this condition, the second menu is positioned with an angle so that its center is aligned with the angle bisector of the angle formed by the borders of the selected menu item. The separation between the first and second menus is determined by the distance choice, which can be either Disjoint or Partial Overlap. Note that the dynamic direction does not affect when the distance method is set to In Place (Fig. 2 (c–d)).

3.3 Selection Techniques

Selection Technique refers to invoking a specific item from the menu. It plays an essential role in any task to execute an action (e.g., opening a browser with a click). We chose two of the most popular selection techniques that do not rely on any external hardware to make them feasible on smartphones.

Dwell. In this *Selection Technique*, users are required to keep the cursor inside a marking menu item for a certain amount of time in order to select it. We found that eye gaze- and head-based systems typically use a dwell time ranging between 0.5 s and 1.5 s [22]. After a pilot study, we kept the dwell time at 1.2 s. To trigger a selection with dwell, we used a timer that started when the circular

cursor entered a menu item, and the item was selected when the cursor remained within the item for the predefined dwell time (i.e., 1.2 s).

Border Crossing. In the Border Crossing *Selection Technique*, a menu item is selected when the cursor moves out toward the outer border of the item of the marking. We anticipate that the technique would be useful compared to Dwell considering the fatigue and discomfort that can arise from a prolonged head fixation on a single menu item (Fig. 1f).

4 User Study

Our study has two main goals: (G1). Explore the most suitable positioning strategy for the second marking menu concerning the first menu when an item in the latter is selected. (G2). Identify appropriate *Selection Technique* for invoking items from hierarchical marking menus.

4.1 Participants and Study Implementation

Twelve participants (8 males, all right-handed), aged between 20 and 25, were recruited from the local university. The average age of the participants was 22.75 years (SD. 1.05). They had little or no prior experience with head-based interfaces; however, had an average of 9.42 years of experience using smartphones.

We developed an Android application for the user study, which was run on a Realme 9 Pro+ smartphone (screen resolution 1080×2400 pixels). To track a user's head, we used MediaPipe Face Detection [15], which can extract 3D face landmarks from a camera input in real time. We leveraged OpenCV Perspective-n-Point (PnP) [4] pose computation libraries to determine head rotation angles in horizontal and vertical directions. The horizontal head movement is then mapped to control the cursor to the left and right on the screen and the vertical head movement to move the cursor up and down - thus allowing users to move the cursor horizontally, vertically, and diagonally. The cursor movement is mapped to move 6 dp (i.e., 18 px) horizontally for each degree of head movement. In contrast, the smaller range of vertical angles and the larger number of pixels in that direction result in a higher amount of cursor movement per degree of head movement, specifically 24 dp (i.e., 72 px) for each angle in the vertical direction. A similar mapping was used in a prior work [24].

4.2 Menu Placement and Selection Techniques

We devised the following placement techniques for the second marking menu based on two directions and three menu distances. (a) *Fixed + Disjoint (FD)*: The second menu appears right below the first menu. The position of the second menu always remained fixed in this condition. (b) *Fixed + Partial Overlap (FPO)*: The second menu appears below the first menu, as we saw in FD. However, in this case, the center of the second menu is located at a distance equal to

the radius of the menu r, resulting in partial overlap with the previous menu. (c) *Fixed + In Place (FIP)*: The second menu appears at the same position as the menu from the previous depth. The position of the second menu remains fixed in this combination, thus, completely overlapping the first menu. (d) *Dynamic + Disjoint (DD)*: The second menu appears at a fixed distance of $2r$ from the previous menu, and its center is oriented towards the direction of the selected target item in the first menu. and (e) *Dynamic + Partial Overlap (DPO)*: With this technique, the center of the second menu appeared at a distance of r. The second menu always appeared in the direction of the previous targeted item, partially overlapping the previous menu. Importantly, Dynamic + In-Place was not investigated as it would behave exactly like Fixed + In-Place. As mentioned before, we investigated two *Selection Techniques*: Dwell and Border Crossing. We anticipate that the selection time with Border Crossing will be shorter than Dwell because Dwell requires additional 1.2 s for selection. However, Border Crossing requires users to be cautious when moving the cursor and might be more susceptible to errors than Dwell.

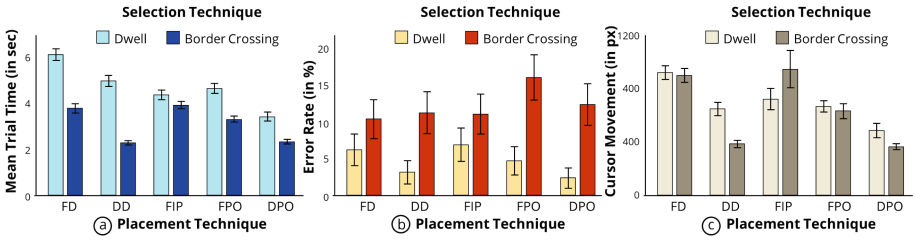


Fig. 3. Mean (a) trial time (in sec), (b) error rate (in %), and (c) cursor movement (in px) for *Placement Techniques* clustered by *Selection Techniques*.

4.3 Task, Procedure and Design

Our study involved tasks in which participants had to select items from marking menus. Wenmin Li et al. [13] suggested using a maximum of eight items on marking menus and having two depths to yield the best performance. Consequently, we had 6 items in a marking menu in two depths. Initially, the first menu was visible without displaying any target item. We decided to use a menu with a 120dp diameter to ensure that a second menu can be placed to the sides in Dynamic directions - ensuring that no part of the second menu extends beyond the screen. A start button was placed below the target close to the edge of the screen (Fig. 2a). Participants could initiate a trial by selecting the start button using a circular cursor controlled with the head movement. This action makes the target item highlighted in sky blue on the menu. We also included textual instructions (e.g., "Please select A1, B3") on top of the screen. If a participant selected a wrong item, an error message ("Wrong Selection") appeared at the top of the screen. Conversely, if the correct target items were selected, a "Successful Trial" message was displayed. When the cursor was placed over a non-target

or the target item, it was highlighted with red or green color, respectively. The menu items were arranged in a 360° circular layout in both depths as shown in Fig. 2. Thus, each item on the menu had an angle of 60°. The labels for the menu items started with a letter ('A' and 'B' for the two depths) and ended with a number, ranging from 'A1' to 'A6' and 'B1' to 'B6'. A trial started once participants selected the *Start* button by moving the cursor over it. A trial ended when either the participants selected both of the target items correctly (i.e., successful trial) or they selected a wrong target item in any of the menus (i.e., error trial). The error trials were then re-queued into the remaining trial pool. We instructed the participants to keep their hands steady while allowing head movement, as we were specifically interested in utilizing head movement for the interaction.

The study used a within-subjects design with two primary independent variables: *Placement Technique* with 5 levels (FD, FPO, FIP, DD, DPO) and *Selection Technique* with 2 levels (Dwell, Border Crossing). Each trial was repeated 10 times for each participant - yielding a total of 5 *Placement Techniques* × 2 *Selection Techniques* × 10 *Repetitions* × 12 Participants = 1,200 trials in total. We counterbalanced the *Selection Techniques* across participants and randomly selected the *Placement Techniques* to conduct the study.

4.4 Measures and Results

The application recorded user performance-related metrics, including trial time, error rate, and cursor movement for all the trials. The trial time was measured from the time when the users clicked the start button to the time when the trial ended successfully. An error was recorded when the participants selected a non-target item in the first or second menu. Cursor movement was calculated by measuring the total path covered by the cursor during a trial. Additionally, we obtained subjective feedback from participants regarding the techniques.

We removed 16 outliers (1.3%) from the data where the trial times were 3 SD away from the mean. We then performed repeated measures ANOVA and Bonferroni adjusted post hoc pairwise comparisons to analyze trial time and cursor movement. Post-hoc pairwise comparisons were Bonferroni adjusted (α-level = 0.05). Friedman tests with Wilcoxon tests were used for post hoc pairwise comparisons to analyze error rates. We define the error rate by the number of trials with an incorrect selection divided by the total number of trials.

Trial time. RM-ANOVA showed a significant effect of placement technique on trial time ($F_{4,36} = 16.44$, $p < 0.001$, $\eta^2 = 0.65$) We found that DPO ($M = 2.74s$) was the fastest, followed by DD ($M = 3.65s$), FPO ($M = 3.83s$), FIP ($M = 4.25s$) and FD ($M = 5.05s$). Post hoc pairwise comparisons showed that DPO was significantly faster than all other techniques, and DD and FPO were significantly faster than FD. There were no other pairwise statistically significant differences. RM-ANOVA also showed a significant effect of the *Selection Technique* on trial time ($F_{1,9} = 32.41$, $p < 0.001$, $\eta^2 = 0.78$) Post hoc pairwise comparisons showed that Border Crossing ($M = 3.15s$) was significantly faster than Dwell ($M = 4.66s, allp < 0.001$). This is understandable as Dwell was

taking an additional 1.2 s for selection, while Border Crossing does not need a further delay to select an item from the marking menu.

Error Rate. The overall error rate was 9% (i.e., the percentage of trials that contained one incorrect selection in the first or the second menu). Figure 3b shows the mean error rates across different *Placement Techniques* and *Selection Techniques*. We observed error rates of 8%, 7%, 9%, 11% and 8% for FD, DD, FIP, FPO, and DPO placement techniques, respectively. However, we didn't find any significant effect of placement technique on the error rate ($\chi^2(4, N = 12) = 8.021, p = 0.08$). Results showed that the *Selection Technique* had an effect on the error rate ($\chi^2(1, N = 12) = 8.33, p < 0.01$). Pairwise comparisons showed that Dwell (5%) caused significantly fewer errors than Border Crossing (12%).

Cursor Movement. Results of the RM-ANOVA revealed a significant effect of placement techniques on cursor movement ($F_{4,36} = 20.13$, $p < 0.001$, $\eta^2 = 0.69$). DPO ($M = 379.57px$) was found to be the placement technique with the least cursor movement, followed by DD ($M = 517.76px$), FPO ($M = 606.79px$), FIP ($M = 699.32px$). FD was the technique that required the most cursor movement to select items on the marking menus ($M = 895.77px$). Post hoc pairwise comparisons indicated that DPO required significantly less cursor movement than all other techniques except for DD. Results also revealed that DD and FPO needed less cursor movement than FD (all $p < 0.05$). No other statistically significant differences were observed between the techniques. RM-ANOVA on the *Selection Techniques* showed no significant effect of the *Selection Technique* on cursor movement ($F_{1,9} = 2.56$, $p = 0.15$, $\eta^2 = 0.21$).

Subjective Feedback. We asked participants to provide us with their feedback on the menu *Placement* and *Selection Techniques*. We received positive feedback on the Dynamic + Partial Overlap (DPO) as it requires less cursor movement to select items compared to the disjoint condition. One participant commented, *"Selecting items (with DPO) is faster and requires less head movement"*. Another participant expressed preferring Dynamic + Partial Overlap (DPO) as he could leverage continuous cursor movement to make item selections on both menus: *"Partial overlap because it allows the user to keep the cursor in the same direction for item selection"*.

5 Discussion and Future Work

G1 Placement Technique: The results of our study indicated that the Dynamic + Partial Overlap (DPO) *Placement Technique* outperformed other techniques in terms of trial time, cursor movement, and error rate. This was due to the fact that DPO rendered the second menu in the same direction and partially overlapped where participants had selected the target item in the first menu. This resulted in less cursor movement and faster access to the target item in the second menu compared to other techniques. It was also evident to the participants that with DPO, the second menu would partially overlap the previous one and be in the

same direction, requiring minimal cursor adjustment. Our study revealed that the Fixed Disjoint (FD) *Placement Technique* resulted in higher trial time and cursor movement compared to other techniques. This was because the second menu appeared in a fixed direction, below the first menu, which might not align with the targeted item in the first menu, leading to more cursor adjustment.

The in-place menu placement technique (i.e., FIP) positions the second menu in the same location as the first menu, which was expected to decrease cursor movements and improve item selection time. However, throughout our study, we observed that participants found this placement problematic when using both selection techniques. Similarly, with Border Crossing, participants had to move the cursor out of the menu for item selection, back inside, and then outside again for item selection in the second menu, leading to a higher mean trial time.

The dynamic menu *Placement Techniques*, namely DD and DPO, generally performed better than other techniques regarding trial time and cursor movement. As previously discussed, with dynamic placement techniques, the subsequent menu is placed adjacent to the selected item in the previous menu. This factor leads to less cursor movement and hence a shorter trial time.

G2 Selection Technique: Our results showed that while Border Crossing is faster on average, Dwell is significantly more accurate for item selection. It is worth noting that the trial time includes an additional 2.4 s required for the Dwell technique to select target items in two depths (1.2 s in each depth). This result differs from the findings of a previous study [23] that utilized eye movement for menu selection. We believe that head movement is faster and more stable than eye movements, which helps Border Crossing to be faster than Dwell.

Our study results demonstrate that the Dwell technique has a lower error rate than the Border Crossing technique. Firstly, participants were observed to be more careful when using the Dwell technique, as it required them to keep the cursor static on the target item. Secondly, the Border Crossing technique involved moving the cursor from the inside of the menu to the outside, and we observed that participants occasionally moved the cursor a significant distance, unintentionally triggering the selection in both the first and second menus. Our results align with prior work [6] that showed the Dwell technique to be less error-prone than other hands-free selection techniques. We had anticipated that the Dwell technique would involve less cursor movement than the Border Crossing. However, our study tasks required participants to move from the first menu to the second, resulting in a comparable amount of movement for both techniques.

We here suggest potential areas for future research. Firstly, to investigate menu placement and selection techniques, we only examined two menus, each containing six items based on [13]. Future studies could explore ways to increase the number of menus and menu items in the menu hierarchy. Secondly, we focused on head-based inputs as a type of non-touch input. Other non-touch input methods, such as eye-based or voice inputs, could be investigated in the future. Finally, further research is needed to explore the potential of head control-based marking menus with placement techniques for common applications including as gaming, texting, or others.

6 Conclusion

In this paper, we explored the optimal placement of subsequent menus within a hierarchical marking menu structure on smartphones. Our results indicated that partially overlapping and dynamically placed second menus could improve user performance in selecting items. Moreover, we observed that although the Dwell selection technique had a longer selection time, it led to fewer errors in comparison to Border Crossing. The results of this study can benefit the design of marking-menu based user interfaces for small-screen mobile devices, such as smartphones, and improve their efficiency and usability.

References

1. Bailly, G., Lecolinet, E., Nigay, L.: Wave menus: improving the novice mode of hierarchical marking menus. In: Baranauskas, C., Palanque, P., Abascal, J., Barbosa, S.D.J. (eds.) INTERACT 2007. LNCS, vol. 4662, pp. 475–488. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74796-3_45
2. Bailly, G., Lecolinet, E., Nigay, L.: Flower menus: a new type of marking menu with large menu breadth, within groups and efficient expert mode memorization. In: Proceedings of AVI 2008, pp. 15–22. ACM, NY, USA (2008). https://doi.org/10.1145/1385569.1385575
3. Bau, O., Mackay, W.E.: Octopocus: a dynamic guide for learning gesture-based command sets. In: Proceedings of UIST 2008, pp. 37–46. ACM, NY, USA (2008). https://doi.org/10.1145/1449715.1449724
4. Bradski, G., Kaehler, A.: Opencv. Leanging OpenCV **3**, 120 (2000)
5. Callahan, J., Hopkins, D., Weiser, M., Shneiderman, B.: An empirical comparison of pie vs. linear menus. In: Proceedings of CHI 1988, pp. 95–100. ACM, NY, USA (1988). https://doi.org/10.1145/57167.57182
6. Esteves, A., Shin, Y., Oakley, I.: Comparing selection mechanisms for gaze input techniques in head-mounted displays. IJHCS **139**, 102414 (2020). https://www.sciencedirect.com/science/article/pii/S1071581920300185
7. Francone, J., Bailly, G., Lecolinet, E., Mandran, N., Nigay, L.: Wavelet menus on handheld devices: stacking metaphor for novice mode and eyes-free selection for expert mode. In: Proceedings of AVI 2010, pp. 173–180. ACM, NY, USA (2010). https://doi.org/10.1145/1842993.1843025
8. Gizatdinova, Y., Špakov, O., Tuisku, O., Turk, M., Surakka, V.: Gaze and head pointing for hands-free text entry: applicability to ultra-small virtual keyboards. In: Proceedings ETRA 2018. ACM, NY, USA (2018). https://doi.org/10.1145/3204493.3204539
9. Henderson, J., Mizobuchi, S., Li, W., Lank, E.: Exploring cross-modal training via touch to learn a mid-air marking menu gesture set. In: Proceedings of MobileHCI 2019. ACM, NY, USA (2019). https://doi.org/10.1145/3338286.3340119
10. Kane, S.K., Wobbrock, J.O., Smith, I.E.: Getting off the treadmill: evaluating walking user interfaces for mobile devices in public spaces. In: Proceedings of MobileHCI 2008 MobileHCI 2008 , pp. 109–118. ACM, NY, USA (2008). https://doi.org/10.1145/1409240.1409253
11. Kurtenbach, G.P.: The Design and Evaluation of Marking Menus. Ph.D. thesis, CAN (1993). uMI Order No. GAXNN-82896

12. Lepinski, G.J., Grossman, T., Fitzmaurice, G.: The design and evaluation of multitouch marking menus. In: Proceedings of CHI 2010, pp. 2233–2242. ACM, NY, USA (2010). https://doi.org/10.1145/1753326.1753663
13. Li, W., Wan, X., Shi, Y., Yao, N., Wang, C., Gao, Z.: Depth and breadth of pie menus for mid-air gesture interaction. IJHCI **37**(2), 131–140 (2021). https://doi.org/10.1080/10447318.2020.1809245
14. LoPresti, E., Brienza, D.M., Angelo, J., Gilbertson, L., Sakai, J.: Neck range of motion and use of computer head controls. In: Proceedings of ASSETS 2000, pp. 121–128. ACM, NY, USA (2000). https://doi.org/10.1145/354324.354352
15. Lugaresi, C., et al.: Mediapipe: a framework for building perception pipelines (2019)
16. Mardanbegi, D., Hansen, D.W., Pederson, T.: Eye-based head gestures. In: Proceedings of ETRA 2012, pp. 139–146. ACM, NY, USA (2012). https://doi.org/10.1145/2168556.2168578
17. Mine, M.R.: Virtual environment interaction techniques. UNC Chapel Hill (1995)
18. Paulus, Y.T., Remijn, G.B.: Usability of various dwell times for eye-gaze-based object selection with eye tracking. Displays **67**, 101997 (2021). https://doi.org/10.1016/j.displa.2021.101997
19. Pollock, J., Teather, R.J.: CountMarks: multi-finger marking menus for mobile interaction with head-mounted displays. In: Stephanidis, C., Kurosu, M., Degen, H., Reinerman-Jones, L. (eds.) HCII 2020. LNCS, vol. 12424, pp. 241–260. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-60117-1_18
20. Qian, Y.Y., Teather, R.J.: The eyes don't have it: an empirical comparison of head-based and eye-based selection in virtual reality. In: Proceedings SUI 2017, pp. 91–98. ACM, NY, USA (2017). https://doi.org/10.1145/3131277.3132182
21. Salkanovic, A., Štajduhar, I., Ljubic, S.: Floating hierarchical menus for swipe-based navigation on touchscreen mobile devices. In: Kurosu, M. (ed.) HCII 2020. LNCS, vol. 12182, pp. 509–522. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-49062-1_34
22. Stampe, D.M., Reingold, E.M.: Selection by looking: a novel computer interface and its application to psychological research. In: Eye Movement Research, Studies in Visual Information Processing, vol. 6, pp. 467–478. North-Holland (1995). https://doi.org/10.1016/S0926-907X(05)80039-X
23. Urbina, M.H., Lorenz, M., Huckauf, A.: Pies with eyes: the limits of hierarchical pie menus in gaze control. In: Proceedings of ETRA, pp. 93–96 (2010)
24. Vo, T., Jannat, M.E., Ahlström, D., Hasan, K.: Faceui: leveraging front-facing camera input to access mid-air spatial interfaces on smartphones. In: Proceedings of GI 2022, pp. 46 – 58. Canadian Information Processing Society (2022). https://doi.org/10.20380/GI2022.06
25. Zhao, S., Agrawala, M., Hinckley, K.: Zone and polygon menus: using relative position to increase the breadth of multi-stroke marking menus. In: Proceedings of CHI 2006, pp. 1077–1086. ACM, NY, USA (2006). https://doi.org/10.1145/1124772.1124933
26. Zheng, J., Bi, X., Li, K., Li, Y., Zhai, S.: M3 gesture menu: design and experimental analyses of marking menus for touchscreen mobile interaction. In: Proceedings of CHI 2018, pp. 1–14. ACM, NY, USA (2018). https://doi.org/10.1145/3173574.3173823

SDN-DMQTT: SDN-Based Platform for Re-configurable MQTT Distributed Brokers Architecture

Fatma Hmissi(✉) and Sofiane Ouni

ENSI, Cristal Laboratory, Manouba University Campus - 2010, Manouba, Tunisia
fatma.hmissi@ensi-uma.tn, sofiane.ouni@insat.rnu.tn

Abstract. The MQTT (Message Queuing Telemetry Transport) protocol is widely recommended for IoT (Internet of Things) communication. It uses a single broker for data exchange, which is typically placed in the cloud. However, this centralized approach is not effective for scaling, leading to mist and edge broker distribution. In distributed infrastructures, administrators are forced to manually reconfigure all MQTT communication to meet performance metrics. However, manual reconfiguration has several drawbacks, e.g., it is a time-consuming process and prone to error. To remedy these issues, this paper proposes combining Software-Defined Networking (SDN) and the MQTT protocol. This proposal enables dynamic reconfiguration of MQTT communication on distributed MQTT-based infrastructure. In the current paper, a real-case study of experiments on a real test-bed is carried out to justify the necessity of the proposed approach.

Keywords: IoT · SDN · MQTT · Dynamic Reconfiguration

1 Introduction

Nowadays, the Internet of Things (IoT) plays an essential role in the world. IoT aims to provide several conveniences and benefits for people's daily lives. To achieve this, connected objects interact and communicate among themselves. This kind of communication between objects (or things) requires IoT-based application layer protocols. These protocols address numerous vital issues in IoT object communication, including unreliable and complex network environments, limited capacities of storage and memory, and restricted processor capability. The focus of attention and research on IoT produce distinct IoT-based application layer protocols. Papers [1–10] survey the key IoT communication protocols for the application layer, e.g., MQTT (Message Queuing Telemetry Transport), CoAP (Constrained Application Protocol), AMQP (Advanced Message Queuing Protocol), and XMPP (Extensive Messaging and Presence Protocol). However,

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 393–411, 2024.
https://doi.org/10.1007/978-3-031-63992-0_26

MQTT has become the choice for IoT. Recent comparative analyses show that MQTT is the preferred protocol for IoT communication because of its advantages. For example, MQTT provides better reliability than other protocols.

MQTT follows the publish-subscribe (pub/sub) architecture for IoT communication. In MQTT-based IoT communication, IoT objects are classified into the following two types: publisher and subscriber. Publishers are producers of data, and subscribers are consumers of data. MQTT employs a broker as a central point to perform and control communication between publishers and subscribers. Publishers connect to the broker and can publish messages on topics. While subscribers may subscribe to particular topics and get published messages.

A number of cloud-based MQTT brokers are available on several cloud platforms [17,18] such as Microsoft Azure IoT Suite. A cloud MQTT broker is an MQTT broker (e.g., Mosquitto [11], HiveMQ CE [13], VerneMQ [14]) placed and running on the cloud [12]. It enables many different IoT objects to communicate together via MQTT, even when they are in different physical geographic locations. However, the number of connected devices is going to increase substantially in the coming years (it is expected that there will be 29.42 billion connected objects globally by the end of 2030 [15,16]). In this trend, the quick increase of the IoT causes a large amount of data and information to be gathered by an enormous number of connected things. Hence, cloud MQTT brokers are considered major problems for MQTT communication. Existing cloud MQTT brokers receive huge amounts of data from a large number of objects and resend it to other objects in a single cloud [19]. Respectively, the employment of MQTT brokers in the cloud causes an unnecessary delay and requires high bandwidth and energy consumption during the MQTT communication process. The additional delay in MQTT communication has a negative impact on the general effectiveness of IoT applications.

To deal with this problem, many researchers have paid attention to looking for promising ways to reduce the delay of MQTT communication. The geographic distribution of MQTT brokers as close as possible to IoT objects is considered a vital solution for this problem. Several solutions, such as those in [19,30] suggested MQTT broker distribution over edge and mist levels. Nevertheless, distributing MQTT brokers over edge and mist levels introduces some side effects associated with the complexities of management (or reconfiguration) and control of MQTT communication.

To address this challenge, a potential solution is needed to control and manage communication efficiently over the distributed infrastructure of MQTT brokers. Within the current available techniques, Software-Defined Networking (SDN) [21–23] is considered the most important option for this challenge. Therefore, the SDN-DMQTT platform is built. The proposed platform is an SDN-based architecture. It hides and simplifies the complexities of MQTT communication management within the distributed physical infrastructure of MQTT brokers. SDN-DMQTT mainly focuses on the following goals:

– Build a programmable distributed MQTT-based IoT architecture capable of dealing with dynamic scalability and adaptability.

- Provide a global view of the distributed physical infrastructure of MQTT brokers.
- Control MQTT communication and brokers remotely.
- Access and reconfigure distributed brokers remotely to deal with dynamic changes in MQTT communication.
- Manage the entire MQTT communication via software from a centralized location.
- Automate certain aspects to handle demand for reconfiguration of MQTT communication against dynamic changes in the environment and/or events that are unexpected.

This paper is structured as follows: Sect. 2 depicts the current state of the art for SDN-based MQTT solutions, figuring out the novelty of the proposal. Section 3 describes the proposed Software-Defined Networking for Distributed MQTT (SDN-DMQTT) platform. Section 4 introduces a functional design of SDN-DMQTT. Section 5 presents a case study. Finally, Sect. 6 concludes this work.

2 State of Art

This section summarizes the most recent contributions found in the literature regarding the combination of the MQTT protocol and SDN technology. The presented works use SDN to enhance the MQTT protocol.

2.1 MQTT Enhancements Using SDN

Shahri et al. [24] proposed a new system called Real-Time MQTT (RT-MQTT), based on SDN, to enhance MQTT with real-time communication services. The architecture includes OpenFlow(OF)-controllerr, OpenFlow-Data Base (OF-DB), OF-switches, Real-Time-Network Manager (RT-NM), and MQTT application components. RT-MQTT uses the user properties field in standard MQTT messages to specify real-time requirements. RT-NM examines messages to identify real-time reservation requests, extracting and registering attributes in OF-DB. The OF-Controller creates deterministic communication channels and updates the flow tables of OF-Switches.

Benson et al. [27,28] proposed a SDN-based middleware system to ensure timely and reliable delivery of mission-critical data from MQTT sources to MQTT consumers in congested networks. The system includes an SDN controller, a coordinator service, SDN switches, brokers, publishers, and subscribers. The coordinator service determines network flow, while the SDN controller configures switch flows and group tables based on priority and drop rate policies. The subscriber sends a subscription packet to the desired SDN switch, while the broker forwards published data to the subscriber.

Fawwaz et al. [29] proposed an SDN-based distributed MQTT broker architecture for the Mobile Edge Computing platform for smart city applications. The

new architecture aims to avoid wasting edge computing resources and reduce network traffic and the latency of data delivery. The proposed architecture consists of dynamically placing MQTT distributed brokers to support the mobility of MQTT applications. When a device changes its location, Fawwaz et al. [29] propose to configure or create a distributed broker on edge resources (SDN switches). Edge resources are chosen in such a way that they should reduce the latency of data delivery and have many computing resources. In this proposal, brokers that are not used for any communication are completely removed from edge resources. The architecture of the proposed system includes distributed brokers (containerized on edge-enabled SDN switches) managed by an SDN controller and Distributed Broker and Edge (DBE) manager, an MQTT proxy, and a central broker. The proposed system uses MQTT proxy servers to synchronize clients and distributed brokers, providing a global view of existing distributed brokers and related topics. The DBE manager manages the table of delegated topics and distributed brokers assigned to them, enabling the proxy to deliver topics to the appropriate distributed broker container and messages to the central broker. The DBE manager also collects and manages edge device computing resources, such as Central Processing Unit (CPU), memory, and storage, using data from the SDN controller to optimize placement. The SDN controller monitors edge device status and statistics, ensuring efficient management of computing resources on the local network.

All of the above approaches show notable advancements essential to IoT MQTT-based communication, such as the timely and reliable delivery of messages, reduced delay, and effective bandwidth usage, demonstrating the practicality of SDN in an IoT MQTT-based system.

2.2 Novelty of SDN-DMQTT

The previous state-of-the-art survey shows that there is a certain need to explore the capabilities of SDN to enhance communication for MQTT-based applications. Nevertheless, the contribution of Fawwaz et al. [29] is the only work that compares directly to our proposal in the sense that it enables controllable and reconfigurable MQTT communication. However, this work includes various drawbacks. The most important ones are listed below.

First, solution of Fawwaz et al. [29] is interesting for edge broker distribution. However, Fawwaz et al. [29] were not considered mist broker distribution. In addition, Fawwaz et al. [29] have changed the MQTT publish-subscribe model to enable controllable and reconfigurable MQTT communication by introducing a proxy into the MQTT communication. Where all communication should pass through a proxy. A proxy is switch-based software that receives messages from MQTT clients and then routes them to the appropriate broker. However, this modification adds complexity to the traditional MQTT-based publish-subscribe model by introducing an intermediate layer that manages message routing between clients and brokers. Moreover, the architecture proposed by Fawwaz et al. [29] does not include an application layer or northbound Application Programming Interfaces (APIs). Hence, other applications (such as load

balancing across brokers and real-time guarantees) that require reconfiguration of MQTT-based communication are not addressed in this architecture. They are unable to communicate with the controller or provide it instructions on how to set up these communications.

Therefore, SDN-DMQTT proposed in this paper solves issues of the state-of-the-art. The proposed solution aims to allow dynamic reconfiguration of MQTT communication on mist and edge MQTT broker distribution while keeping the simplicity of the traditional MQTT-based publish-subscribe model. Also, SDN-DMQTT addresses several optimization applications. Each application has the ability to communicate with the controller and provide instructions on how to set up MQTT communications.

3 SDN-DMQTT Overview

This section illustrates and describes the proposed platform. In the first place, it present the principle and aim of the proposal. On the other hand, it detail how the platform works by detailing its architecture.

3.1 Description

It is highly recommended to use the MQTT protocol for communication between IoT objects. MQTT-based communication consists of any exchange of data that takes place between IoT objects. MQTT uses a publish-subscribe pattern and a single broker for communicating data among customers based on topics of interest. Nevertheless, such a centralized approach does not scale effectively. Therefore, there is a need for mist and edge broker distribution. In these distributed infrastructures, reconfiguration of MQTT communication is required to meet a number of performance metrics, such as the requirement to extend communication to real-time.

Example. For example, considering a distribution of brokers for a smart city application, as can be seen in Fig. 1. Broker (b1) carries out communications from publisher (p) to the consumer who is interested (s) based on topic 1. Assume that the administrator requires enhanced communication between s and p by reducing the time of messages' delivery. The requirement can be addressed by minimizing the length of the pathways that the messages published by p would follow. This optimization can be accomplished by re-configuring MQTT communication, i.e., using broker b2 instead of b1.

The reconfiguration of MQTT communication is done manually by administrators, who shut down the publishers and subscribers, reconfigure their parameters, and then bring them online. In this case, subscribers are responsible for broker allocation. Obviously, this approach is very complex and has several drawbacks. This approach consumes a lot of time due to administrator interaction is prone to error. Furthermore, it is infeasible for an IoT network with billions of objects that need to communicate between them. Moreover, it is unsuitable for high-availability applications. To cope with these issues, automation and

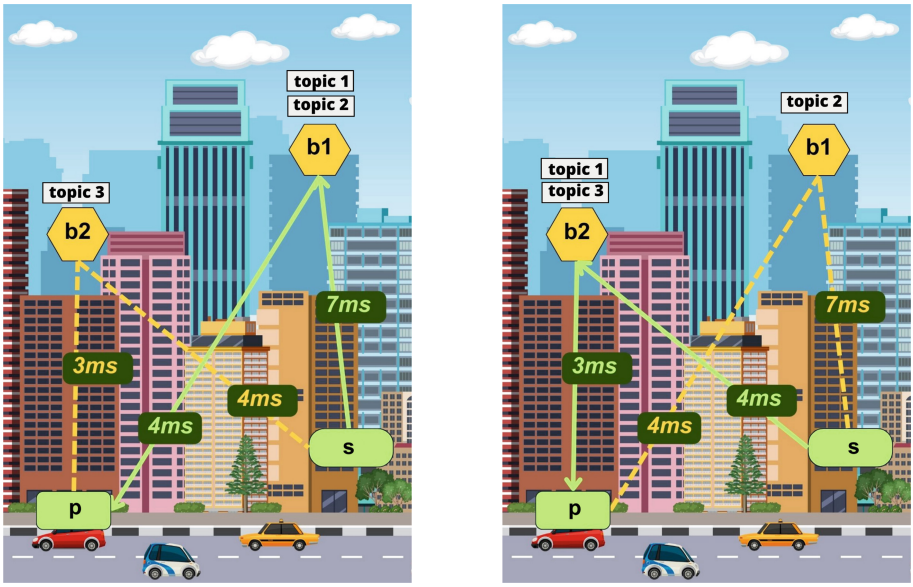


Fig. 1. An example that shows MQTT communication before and after reconfiguration

orchestration, programmability, and dynamicity, and visibility are required. To fit these characteristics, SDN-DMQTT is proposed. SDN-DMQTT leverages the capabilities of SDN technology and the MQTT protocol. The proposed approach remotely controls and reconfigures MQTT communication via a controller without requiring physical access to the MQTT components. In dynamic reconfiguration, the publishers and subscribers are connected automatically to the adjacent MQTT distributed brokers via the controller. In addition, the distributed MQTT brokers are allocated dynamically (topics are dynamically assigned to distributed MQTT brokers). The controller on the control plane dynamically controls and reconfigures MQTT communication on the infrastructure layer. MQTT applications in the application layer (e.g., monitoring of communication, load balancing, load sharing, extending communication with real-time requirements) may be managed and programmed by the controller using the Northbound API (e.g., REpresentational State Transfer (REST) API). The control layer is decoupled from the infrastructure layer, and it is logically centralized, having a global view of the whole MQTT communication. The controller guides the IoT devices to connect to their target brokers, and directs brokers to enable MQTT communication via the Southbound API (MQTT is used as a Southbound API).

3.2 Architectural Details

The proposal is an SDN-based, three-layered architecture. The proposed architecture includes an application layer, a control layer, and an infrastructure layer. Figure 2 depicts a schematic view of this architecture.

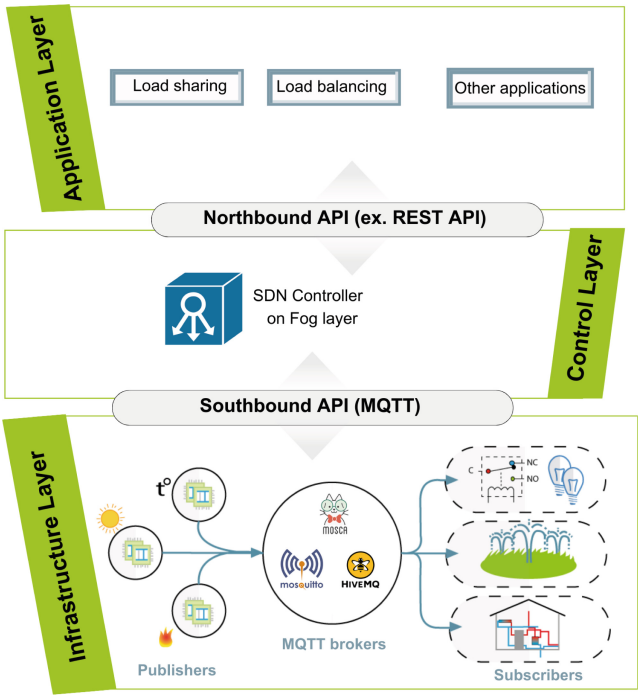


Fig. 2. Architecture of the proposed approach.

Application Layer is the highest layer in the proposed platform. Within that layer all the attributes, functions and rules of MQTT communication are defined. Applications demand the information of a distributed MQTT broker in order to react to it. These applications are able to make decisions concerning the allocation of distributed brokers based on changes, e.g., changes in the resources of brokers and the real-time location of topics. For example, when the processing time of a distributed broker increases, applications are able to change MQTT communication to a new broker dynamically. The application layer takes over all the applications like load balancing between distributed brokers, enforcement of real-time requirements, and broker monitoring (active broker on network, queue size, CPU, battery, topic location). These applications communicate with the SDN controller using the so-called northbound interface (e.g., REST). A northbound interface is an application programming interface (API) or protocol that enables an SDN controller to communicate with an application. Here, the SDN controller exposes northbound APIs, which are used by applications that can run on the same machine as the controller or on a different machine.

Control Layer is the middle layer in the proposed architecture. The control layer is responsible for controlling and reconfiguring communication over distributed MQTT infrastructures. For that purpose, it is made up of SDN controllers. An SDN controller is a central control entity in our proposal. This entity

works as the brain of SDN-DMQTT. It occurs as software on a centralized server at the fog level. The SDN controller provides two main tasks on SDN-DMQTT. First, it provides a centralized view and control of the MQTT broker distribution on the Edge and Mist layers and exchanges this information with upper-layer applications. The next task of the SDN controller is the management of MQTT communication according to the commands of the application layer.

To achieve these tasks, the SDN controller includes the following modules:

- Brokers monitoring module: the SDN controller uses this module to keep track of the distributed brokers. The controller implements this module to discover active brokers that are distributed at the edge and mist levels. Some existing works, such as in proposed a solution for brokers discovery [30]. The controller exchanges the list of discovered brokers with the top layer to give applications a global view of existing distributed brokers. In addition, it collects edge and mist device status and statistics via the SDN controller. These devices are nodes that deploy MQTT brokers. Furthermore, it collects status and statistics of the distributed MQTT broker, e.g., queue size, number of connected clients, and number of archived messages. The broker's information is used to decide about the optimization of topic placement or location. This module gets devices computing resources, such as CPU, memory, battery, and storage.
- Module of Topic Discovery: The controller provides this module to discover and retrieve topics assigned to activate MQTT brokers.
- Communication Management Module: This module allocates or frees brokers by assigning or disassociating topics to them. Therefore, it controls a handover procedure for moving topics from one broker to another. Moreover, it informs the client of the topics' locations.

Infrastructure Layer is the lowest layer in SDN-DMQTT. In the infrastructure layer, all the components of distributed MQTT are placed and connected physically: publishers, subscribers, and distributed brokers.

Publishers receive information from the controller about where to publish the data on a topic. Moreover, subscribers receive information from the controller about where to subscriber to a topic. Furthermore, the distributed brokers in the IoT network process a specific set of topics under the control of the SDN controller. In another world, distributed brokers maintain information from the SDN controller about their authorized topics. An authorized topic is used by a broker to filter published messages and forward them to their corresponding subscribers. In SDN-DMQTT, brokers are authorized only to filter and forward messages for topics assigned by the controller.

4 Functional Design

To demonstrate the feasibility of the SDN-DMQTT platform, a prototype was proposed. Figure 3 designates the proposed prototype.

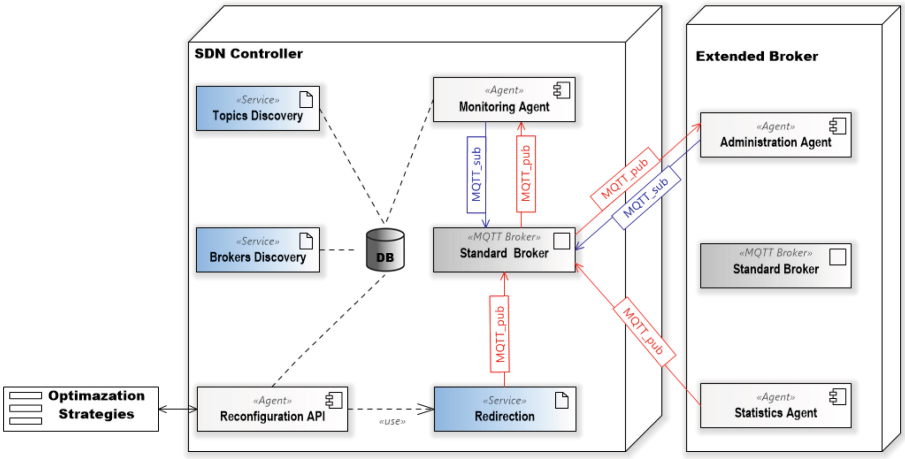


Fig. 3. Functional Design of SDN-DMQTT.

4.1 Extended Broker

It runs on the infrastructure plane, and it is responsible for delivering messages to the correct subscribers. Moreover, it is configurable by the SDN controller with which it interacts using the MQTT protocol. The components of the extended broker are shown in Fig. 3. The extended broker includes a standard MQTT broker (i.e., Mosquitto, HiveMQ, and VerneMQ) to perform the primary purpose of a broker. The standard MQTT broker operates as an intermediary, facilitating communication between publishers and subscribers. In addition, the proposed broker uses an administration agent to be notified of administration events (namely topic authorization and topic unauthorization) of the SDN controller via a standard MQTT broker. This agent configures the standard broker according to the command from the SDN controller. It is designed as an MQTT subscriber that subscribes to an administration topic in order to receive commands from an SDN controller via MQTT_sub. Furthermore, the extended broker comprises a statistics agent. This agent examines and reports on the extended broker. It is often used to track information about the CPU, memory, battery, connected clients, and number of retained messages. This agent delivers its report to the SDN controller periodically via MQTT_pub interface.

4.2 SDN Controller

The SDN controller works as the brain of SDN-DMQTT. It is in charge of discovering active brokers, controlling them, and discovering topics. Furthermore, the administrator employs it to dynamically set up the MQTT brokers and clients with the necessary communication information in an evolutionary manner. The functional design of the SDN controller is shown in Fig. 3. In particular, the designed controller is a black box that uses an MQTT-based API to interact with extended brokers. Following are the specifics. The designed SDN controller uses a simple MQTT API to monitor and manage extended brokers. Therefore, it encompasses these common tools: an MQTT broker, a monitoring agent, and a redirection service. The SDN controller uses a standard MQTT broker that acts as a go-between for the SDN controller and all the extended brokers. The monitoring agent presented in the diagram is an important component of the SDN controller used to monitor and surveil extended brokers. This agent subscribes to monitoring topics, e.g., myapi/brokerId/status, via MQTT_sub interface to receive statistics on extended brokers. The redirection service is used to move a topic from a broker, set a topic to a broker, and inform clients of topics' locations. Thus, this service publishes data on administration topics to indicate administration commands. This service is activated by the reconfiguration API. In addition, the SDN controller employs a database to store an organized collection of structured information about locations of topics and monitoring data of extended brokers and their status (is it an active endpoint or not). The SDN controller regularly discovers the active brokers on the network and the used topics via the broker and topic discovery service.

The SDN controller provides reconfiguration Application Programming Interfaces (or APIs) that administrators can use to build their own applications. An API is a set of programming instructions and standards for accessing data collected from the monitoring agent, brokers, and topic discovery services and translating administration requests to low-level language (or MQTT API). These instructions are a collection of requests for actions. In the following, a list of the API actions supported by the SDN controller and their expected input parameters is provided.

- The action named *redirectTopic(t_topic,a_broker,b_broker)* which takes the name of topic t, a reference to broker a, and a reference to broker b as parameters. This action updates the location of topic t from broker a to broker b. Therefore, it calls the service redirection to move topic t from broker a and set it to broker b. When the SDN controller successfully processes this action, the broker b can use the topic t to filter messages. However, broker a can't use it.
- The action named *setTopicLocation(t_topic,a_broker)* which has two parameters: a string that identifies a topic and a reference to a broker. This action is called to add the topic t to broker a. After the execution of this action, the broker a can use the topic t to filter messages.
- The *getBrokerTopics(a_broker)* action is passed a reference of a broker as a parameter. This action returns a list made up of the topics used to filter

messages on a_broker. If there are no topics, then the action returns an empty list.
- *getBrokerStatus(a_broker)* is an action that requires a broker reference as an input. It returns the statistics, such as remaining memory and energy, of an extended broker stored on the database.
- *getTopicLocation(t_topic)* returns the reference of the broker that filters messages using topic t. The argument may be a string that indicates the name of topic t. If the argument is an empty string, a list of all topics and their locations is returned.
- The *getBrokers()* action has no parameter as input and returns the list of all active MQTT brokers on the network.

5 Case-Study: Use of SDN-DMQTT for Load Sharing

SDN-DMQT is designed to control and reconfigure MQTT communication remotely. In this section, we provide a case study of how SDN-DMQT can be used for automatic control and reconfiguration of MQTT communication. Using SDN-DMQTT, the administrators are totally focused on the design and development of their application without being aware of how the MQTT components are controlled and managed. This section discuss a case study of many applications that can use SDN-DMQTT to optimize MQTT communication. The paragraph that follows represents the fundamental objective of this case study.

Distributed brokers placed on mist or edge nodes. However, brokers have a capacity and bandwidth limits to receive and to send messages. When the load on a broker achieves the maximum capacity of transmission medium, it will not be able to process the whole workload. Thus, delay of message communication increases when broker becomes overloaded. To solve this issue, a method known as "load sharing" is required. Load sharing aims to reduce burden of an overloaded broker. More specifically, it prevents load of a single MQTT broker of exceeding a specific threshold. However, when the broker's load reaches the threshold this method shift a particular amount of load to under loaded broker. This approach optimizes the performance of MQTT communication and availability of brokers. Load sharing is committed to reduce load of brokers to meet the delay requirements of MQTT communication. SDN-DMQTT allows administrators to automate a load sharing algorithm. This SDN-DMQTT based algorithm automatically control loads of distributed brokers and triggers the reconfiguration process of MQTT's communication when either of brokers ' loads achieve the threshold. Based on SDN-DMQTT, algorithm 1 proposes a load sharing algorithm with stable and minimal complexity and efficient load distribution influence. As defined in algorithm 1, the load sharing mechanism uses the following primitives of the controller API: getBroker(), getBrokerStatus(b), getTopics(b), getTopicLoad(t_topic,broker), and redirectTopic(t_topic,a_broker,b_broker).

Algorithm 1. A load sharing algorithm based on SDN-DMQTT

```
B : a set of active brokers
B_o : a set containing overloaded brokers
B_u : a set containing under loaded brokers
T : a simple threshold
S_b : a set of topics assigned to broker b
l_b : load of broker b
l_s : load of topic s
procedure LoadSharing(T)
    B ← getBroker()
    B_o ← {∅}
    for all b ∈ B do
        l_b ← getBrokerStatus(b).load
        if l_b > T then
            B_o ← B_o ∪ {b}
        end if
    end for
    B_u ← B_o ∩ B
    for all b ∈ B_o do
        while getBrokerStatus(b).load > T do
            S_b ← getTopics(b)
            s ← S_b[1]
            l_max ← getTopicLoad(s, b)
            for i = 2 to |S_b| do
                l_s ← getTopicLoad(S_b[i], b)
                if ls > l_max then
                    s ← S_b[i]
                    l_max ← l_s
                end if
            end for
            b_s ← b
            i ← 1
            while i <= |B_u| or b_s = b do
                l_i ← getBrokerStatus(B_u[i]).load
                if l_max + l_i < T then
                    b_s ← B_u[i]
                    redirectTopic(s, b, b_s)
                end if
                i ← i + 1
            end while
        end while
    end for
end procedure
```

5.1 Experiment Test-Bed

This section describes the test bed that has been set up to conduct our experiments. For this study, a test network was performed in a local computing environment, as shown in Fig. 4.

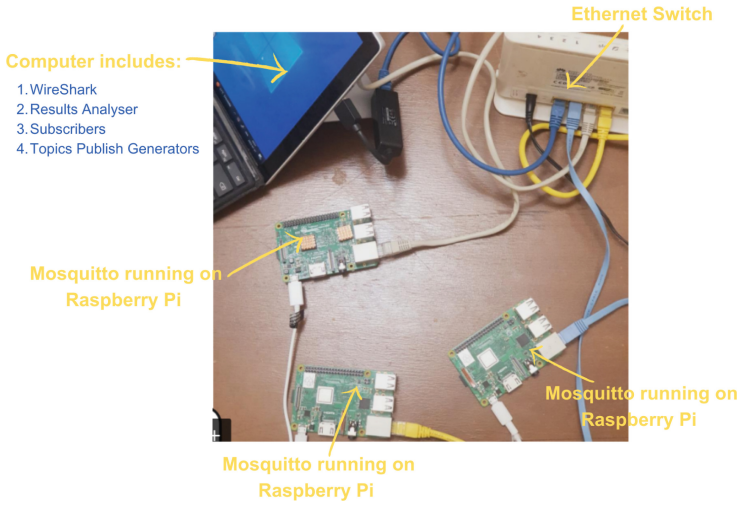


Fig. 4. Experimental environment.

As illustrated in Fig. 4, the hardware tools used in experiments are three Raspberry Pi [31], one Switch Ethernet, and one computer. For the experiments, a Raspberry Pi 3 Model B is employed. The Raspberry Pi is a tiny embedded IoT device with all the necessary features for operating as a computer. It uses less energy. On the Raspberry Pi, the Raspbian 10.13 buster [32] operating system is set up and installed for use in real-world applications. The memory is a 16-GB SD card. And the switch Ethernet is used for creating an isolated network without outside traffic and interference.

For all experiments, the following five software tools were used, as depicted in Fig. 5:

- Mosquitto is a free and open-source implementation of a broker for the MQTT protocol. It provides data to clients who are connected to it.
- Subscribers are MQTT clients. When a client subscribes to data, it means that it will receive the latest information from the broker as soon as it becomes available.
- Topics Publisher Generator is a Paho Python client. It publishes messages according to Poisson processes with a random arrival rate. In other words, this generator is typically employed to produce MQTT messages at a specific rate but is actually completely random.

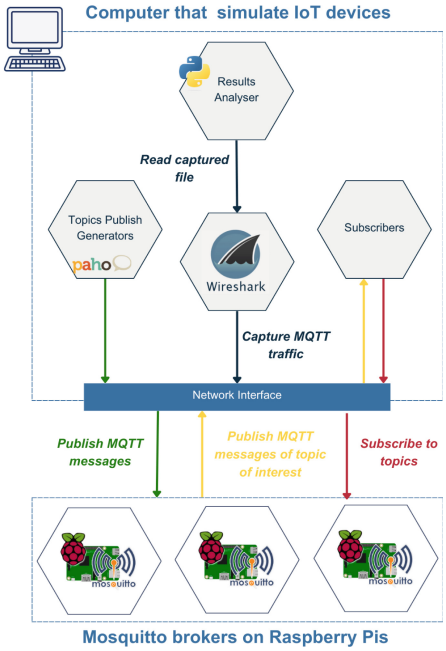


Fig. 5. Test-bed and evaluation model.

- WireShark [33] is a free, open-source packet analyzer for network maintenance, protocol development, and education. WireShark was downloaded on the computer side to capture all messages sent and received by the computer, and a special MQTT filter was configured because the focus of this expriment was only on MQTT packets.
- The Results Analyzer is a Python program. It analyzes the network scan log from a WireShark to show the delay of MQTT communication and the load of brokers.

5.2 Test Scenarios and Results

In this section, two different scenarios will be evaluated and discussed in order to understand the threshold-based load sharing mechanism. The first scenarios focus on analyzing and understanding the importance of load sharing and distribution across brokers when more than one broker's load reaches a specific threshold. The second scenario shows the benefits of the proposed mechanism. Test scenarios are given below.

Influence of Broker's Load on Delay of Communication. The aim of this test scenario is to study how the broker's load could increase the delay of MQTT communication. In the test scenario, a producer that publishes messages on a specific topic and a consumer that subscribes to the same topic are implemented. So, on the computer side, an MQTT client called "Publisher" was created using the Python programming language to publish messages on a given topic tag to one of the Mosquitto brokers on the Raspberry PI Board side. The publisher uses the Topic Publisher Generator to generate random published messages as Poisson arrivals, increasing the traffic load $\lambda \in \{400,500,1000, 2000, 3000, 4000, 5000,6000,10000\}$ messages/s. To receive data, a second MQTT client called "subscriber" was developed to subscribe to the publishing topic and receive data. In this test, the delay in actual communication was measured. For verification, WireShark packet capture software was used to capture, save, and analyze the received data (Fig. 6).

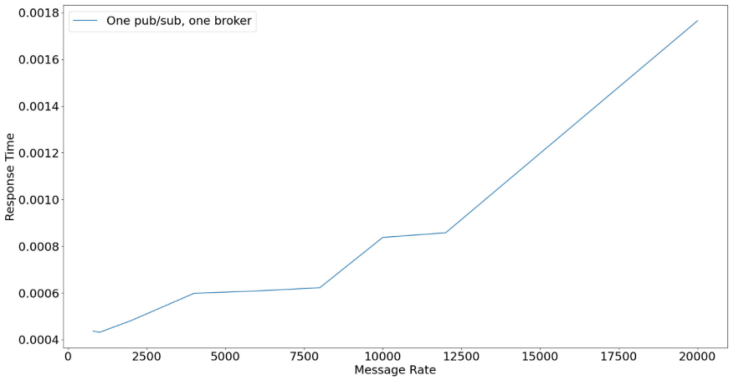


Fig. 6. Delay of 1st communication

Figure 6 represents the delay of communication against the load of the broker. Results indicate that the delay in communication increases with the broker's load. Figure 6 illustrates that with a load of up to 10,000 messages per second, an average delay of less than 0.001 s (1 millisecond) was obtained. However, an important increase in the delay of communication was noticed from the load of 12000 messages per second. From this scenario, It can be concluded that the load on a broker negatively affects the delay of MQTT communication when it reaches a specific threshold. Since, the delay gets going to increase significantly. Therefore, it's highly recommended to use a load-sharing algorithm.

Benefit of Load Sharing on Delay of Communication. This scenario aims to show that load sharing is required to reduce a broker's response time (and consequently reduce the delay of communication) when a broker is overloaded. In addition, it shows that it is highly recommended to use SDN-DMQTT for load

sharing mechanisms. As MQTT is based on a topic pattern, communication is classified per topic, and broker loads are organized into topics. Therefore, the delay of communication per topic is presented in this experiment.

This experiment focuses on justifying the benefits of using a load-sharing algorithm on the delay of each communication. Therefore, three topic-based communication channels were set up. Each communication includes a publisher that periodically publishes messages about a given topic (Top/Topic1, Top/Topic2, or Top/Topic3) as a Poisson arrival. And it includes a subscriber who is interested in receiving messages about topics.

Initially, all communication was set up to go through a single broker. Figure 7 shows that the delay increases remarkably for each communication after 19 s. This behavior is due to the significant increase in broker' load (see Fig. 8 that illustrates broker' load organized per topic.

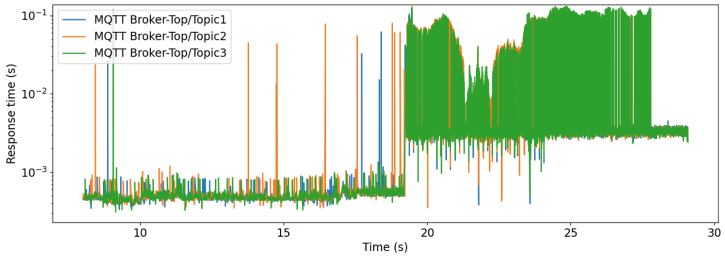


Fig. 7. Delay of communication before load sharing.

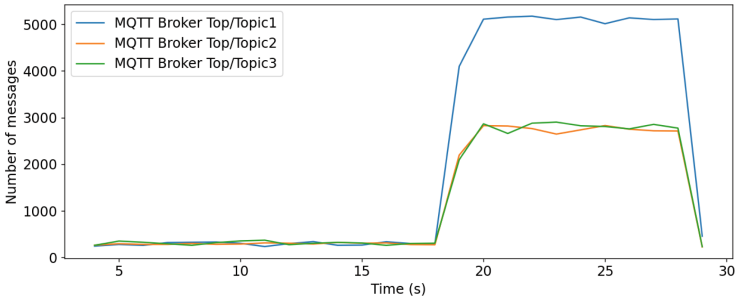
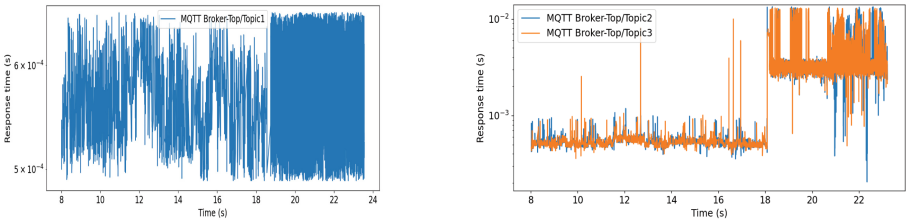


Fig. 8. Load of a broker before load sharing.

Afterwards, the delay of this communication was studied by distributing the current broker's load with another broker. Therefore, the SDN-controller will reconfigure the first communication (Top/Topic1-based communication) to go through an other broker. Comparing delay of communication that were done

without load sharing (See Fig. 7), it is obvious that delay of communication achieve better performance when load sharing is employed (See Fig. 9). Thus, in first case without load sharing, the response time delay will reach 0.1 s. However with applying the Algorithm 1 of the SDN controller, the reconfiguration will share the load between brokers. Therefore, reducing the workload per broker will lead to reduce the response time to 0,01 s for broker 2 as showed in Fig. 9 (b) and the response time to 0.001 s for the broker 1 (Fig. 9 (a)).



(a) Delay of 1st communication

(b) Delay of 2nd and 3rd communication.

Fig. 9. Delay of MQTT communication after load sharing.

6 Conclusion

MQTT broker distribution refers to the process of distributing MQTT brokers across multiple nodes in a network. This distribution enables better scalability, reliability, and load balancing of MQTT message traffic. It ensures that the MQTT brokers can handle a larger number of clients and messages by distributing the workload among multiple servers. Despite the various advantages of MQTT broker distribution, ensuring dynamic reconfiguration of communication becomes more challenging as the number of brokers and connections increases, making it crucial to propose robust mechanisms for dynamic reconfiguration. Therefore, SDN-DMQTT was proposed. SDN-DMQTT is a novel platform that leverages Software-Defined Networking (SDN) to enable seamless reconfiguration of MQTT communication within the distributed architecture of MQTT brokers. By harnessing the power of SDN, this platform offers enhanced flexibility and scalability for managing MQTT communication across multiple brokers, making it an ideal solution for large-scale IoT deployments. With SDN-DMQTT, organizations can easily adapt their MQTT communication infrastructure to meet evolving requirements and optimize communication performance.

References

1. El Alami, H., Sidna, J., Baina, A., Najid, A.: Analysis and evaluation of communication Protocols for IoT Applications. In: Proceedings of the 13th International Conference on Intelligent Systems: Theories and Applications (SITA 2020), November 2020. https://doi.org/10.1145/3419604.3419754

2. Chen, Y., Kunz, T.: Performance evaluation of IoT protocols under a constrained wireless access network. In: 2016 International Conference on Selected Topics in Mobile & Wireless Networking (MoWNeT), Cairo, Egypt, pp. 1–7 (2016). https://doi.org/10.1109/MoWNet.2016.7496622.
3. Iqbal, F., Akhtar, S.M., Anwar, R.: A survey of application layer protocols of internet of things. Int. J. Comput. Sci. Netw. Secur. (IJCSNS) **21**(11), 301–311 (2021). http://paper.ijcsns.org/07_book/202111/20211141.pdf
4. Bayılmış, C., Ebleme, M., Çavuşoğlu, Ü., Küçük, K., Sevin, A.: A survey on communication protocols and performance evaluations for internet of things. Digit. Commun. Netw. **8**, 1094–1104 (2022). https://www.sciencedirect.com/science/article/pii/S2352864822000347
5. Naik, N. Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP. In: 2017 IEEE International Systems Engineering Symposium (ISSE), pp. 1–7 (2017)
6. Ouadghiri, M., Aghoutane, B., Farissi, N.: Communication model in the internet of things. Procedia Comput. Sci. **177**, 72–77 (2020). https://www.sciencedirect.com/science/article/pii/S187705092032278X, The 11th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN 2020)/The 10th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (ICTH 2020)/Affiliated Workshops
7. Al-Masri, E.: Investigating messaging protocols for the internet of things (IoT). IEEE Access. **8**, 94880–94911 (2020)
8. Silva, D., Carvalho, L., Soares, J., Sofia, R.: A performance analysis of internet of things networking protocols: evaluating MQTT, CoAP, OPC UA. Appl. Sci. **11** (2021). https://www.mdpi.com/2076-3417/11/11/4879
9. Donta, P., Srirama, S., Amgoth, T., Annavarapu, C.: Survey on recent advances in IoT application layer protocols and machine learning scope for research directions. Digit. Commun. Netw. **8**, 727–744 (2022). https://www.sciencedirect.com/science/article/pii/S2352864821000845
10. Jerrin Simla, A., Chakravarthy, R.: Review on application layer protocol for iot enabled agricultural intrusion detection. In: 2021 International Conference On Artificial Intelligence And Smart Systems (ICAIS), pp. 1170–1175 (2021)
11. Light, R. Mosquitto: server and client implementation of the MQTT protocol. J. Open Source Softw. **2**, 265 (2017). https://doi.org/10.21105/joss.00265
12. Mishra, B., Mishra, B., Kertesz, A.: Stress-testing MQTT brokers: a comparative analysis of performance measurements. Energies **14** (2021). https://www.mdpi.com/1996-1073/14/18/5817
13. Try hivemq HiveMQ. https://www.hivemq.com/ Accessed 17 Aug 2023
14. Company, O.: VerneMQ - A MQTT broker that is scalable, enterprise ready, and open source. (vernemq.com). https://vernemq.com/
15. Hnatyuk, K.: 120+ internet of things (IoT) statistics: market size, usage, growth & security. (MarketSplash) (2023). https://marketsplash.com/internet-of-things-statistics/#:~:text=The%20number%20of%20IoT%20devices%20is%20forecasted%20to%20reach
16. Katigbak, R.: The economy of things: the next value lever for telcos. (IBM Blog,2023,7). https://www.ibm.com/blog/the-economy-of-things-the-next-value-lever-for-telcos/
17. Hejazi, H., Rajab, H., Cinkler, T.: Lengyel, L. Survey of platforms for massive IoT, January 2018

18. Babun, L., Denney, K., Celik, Z., McDaniel, P., Uluagac, A.: A survey on IoT platforms: communication, security, and privacy perspectives. Comput. Netw. **192** 108040 (2021). https://www.sciencedirect.com/science/article/pii/S1389128621001444
19. Park, J., Kim, H., Kim, W.: DM-MQTT: an efficient MQTT based on SDN multicast for massive IoT communications. Sensors **18** (2018). https://www.mdpi.com/1424-8220/18/9/3071
20. Hmissi, F., Ouni, S.: An MQTT brokers distribution based on mist computing for real-time IoT communications, 13 July 2021. PREPRINT (Version 1). https://doi.org/10.21203/rs.3.rs-695717/v1
21. Zemrane, H., Baddi, Y., Hasbi, A.: SDN-Based Solutions to Improve IOT: Survey, October 2018
22. Sharma, R.: A review on software defined networking. Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol. 11–14 (2021)
23. Mohammed, A., Khaleefah, R., Hussein, M., Abdulateef, I.: A review software defined networking for internet of things, July 2020
24. Shahri, E., Pedreiras, P., Almeida, L.: Extending MQTT with real-time communication services based on SDN. Sensors **22** (2022). https://www.mdpi.com/1424-8220/22/9/3162
25. Almeida, L., Shahri, E., Pedreiras, P.: Enhancing MQTT with real-time and reliable communication services. In: IEEE International Conference on Industrial Informatics (INDIN), vol. **22** (2021)
26. Tamri, R., Rakrak, S.: The SDN-MQTT for an interoperable smart home. In: E3S Web Of Conferences, vol. **229** p. 01031 (2021)
27. Benson, K., et al.: FireDeX: a prioritized IoT data exchange middleware for emergency response. In: Proceedings Of The 19th International Middleware Conference, pp. 279–292 (2018). https://doi.org/10.1145/3274808.3274830
28. Scalzotto, L., et al.: An implementation experience with SDN-Enabled IoT data exchange middleware. In: Proceedings Of The 19th International Middleware Conference (Posters). pp. 21–22 (2018). https://doi.org/10.1145/3284014.3284025
29. Fawwaz, D., Chung, S., Ahn, C., Kim, W.: Optimal distributed MQTT broker and services placement for SDN-edge based smart city architecture. Sensors **22** (2022). https://www.mdpi.com/1424-8220/22/9/3431
30. Hmissi, F., Ouni, S.: TD-MQTT: transparent distributed MQTT brokers for horizontal IoT applications. In: 2022 IEEE 9th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), pp. 479–486 (2022)
31. Ltd, R. Buy a raspberry Pi 4 model B. (Raspberry Pi). https://www.raspberrypi.com/products/raspberry-pi-4-model-b/
32. Debian - News - Updated Debian 10: 10.13 released. (www.debian.org), https://www.debian.org/News/2022/20220910
33. Foundation, W. Wireshark. (Wireshark.org.2016). https://www.wireshark.org/

Mobile Augmented Reality and Applications for Mobile Computing

Quality Evaluation of Image Segmentation in Mobile Augmented Reality

Shneka Muthu Kumara Swamy(✉) and Qi Han

Colorado School of Mines, Colorado, USA
{smuthukumaraswamy,qhan}@mines.edu

Abstract. Mobile Augmented Reality (AR) facilitates a seamless interactive experience between actual and virtual environments. AR employs segmented images for various purposes such as object recognition, occlusion boundary estimation, and foreground-background separation. However, evaluating the quality of segmented images in mobile AR is challenging due to the limited resources of mobile devices. Existing solutions employ neural networks with many layers, making it difficult to deploy them on mobile devices. To address this issue, we propose techniques to modify the inputs so that we can reduce the number of layers in the neural network, making it possible to deploy for mobile devices. This idea is incorporated into our proposed SegQNet. It utilizes deep learning techniques based on convolutional neural networks (CNNs) to evaluate the quality of overlaid segmentation in mobile AR. SegQNet achieves high accuracy without the need for ground-truth images or expensive computations. Our experiments on Android smartphones demonstrate that SegQNet outperforms two state-of-the-art methods without incurring significant overhead.

Keywords: Augmented Reality · Quality of Segmentation · Neural Network for Mobile Devices

1 Introduction

Mobile Augmented Reality (AR) has gained immense popularity in recent years, offering the ability to overlay digital images onto real-world environments. Image segmentation plays a vital role in AR applications, providing a more immersive and realistic user experience. In particular, AR uses segmented images for object recognition, occlusion boundary estimation, and foreground-background separation [14,16,17]. For example, when a user is navigating through a crowded urban environment with multiple routes to choose from. The user can benefit from an overlaid map and a segmented view of different real-world objects such as streets, buildings, and sidewalks.

Despite the significant progress made in enhancing image segmentation techniques for many applications, the evaluation of segmentation quality remains

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 415–427, 2024.
https://doi.org/10.1007/978-3-031-63992-0_27

a challenging issue for mobile augmented reality (AR). The currently available methods for assessing the quality of segmentation are not entirely suitable for mobile AR since they either rely on ground truth images that may not always be accessible or require computationally intensive operations that mobile devices may not support. Therefore, a fast and resource-efficient quality evaluation method suitable for mobile AR is needed.

Evaluating the quality of segmented images in mobile AR is faced with several challenges. First, mobile devices have limited computational resources, and any quality assessment process should not compete with the primary AR application for resources. Consequently, evaluation methods must be designed to be efficient in their use of resources. Second, the storage capacity for each Android app is restricted, making it impractical to store a large deep-learning model. Therefore, a lightweight model needs to be developed to complete the task. Three, AR applications may require quantitative quality measurement, so a rough quality classification may be inadequate. Last, AR scenes are complex as they may contain occlusion, background, and foreground objects.

This paper tackles these challenges by presenting a fast and efficient method for evaluating segmentation quality suitable for mobile AR. To demonstrate that our method can work for mobile AR, we implement it on Android smartphones. More specifically, our main contributions in this work are as follows.

1. We have designed SegQNet, a no-reference model that utilizes a lightweight convolutional neural network (CNN) to determine the quality of segmented images. SegQNet distinguishes itself from existing work in several ways. 1) SegQNet does not require many layers in the neural network, making it possible to deploy on mobile devices. 2) Unlike some previous studies [12,22] that were limited to medical images, SegQNet can handle all kinds of images. 3) SegQnet has the ability to process images of varying sizes, eliminating the need to pre-process images to conform to a specific size as required in existing methods [19]. 4) SegQNet does not need ground truth images.
2. We have evaluated SegQNet's performance on a PC with a GPU and compared it to two other approaches – MultiScale Net [19] and Pull the Plug [12]. The results show that SegQNet achieves an average loss that is comparable to MultiScale Net but uses less storage and computation. SegQNet also outperforms Pull the Plug in terms of average loss.
3. We have implemented SegQNet, MultiScale Net, and Pull the Plug on Android smartphones. Results show that SegQNet is more efficient in terms of CPU and energy usage and incurs less delay compared to MultiScale Net.

The rest of the paper is organized as follows. Section 2 provides a summary of the related work; Sect. 3 discusses our model for evaluating the quality of segmented images overlaid in AR; In Sect. 4 we present the performance of SegQNet and comparison with two baseline approaches using GPU in a PC; In Sect. 5 we demonstrate the working of our model in smartphones and show it performs better than baseline approaches. We conclude our paper in Sect. 6.

2 Related Work

Image segmentation is susceptible to six types of errors [8], including dilation, erosion, boundary localization, object localization, boundary approximation, and inner mask.

Objective segmentation quality evaluation methods can be classified into full reference (i.e., requiring the comparison against ground truth images), no reference (i.e., no need for ground truth images), and partial reference methods (i.e., no need for ground truth images, but need some kind of base images). The most recent full reference methods [18] considered human visual perception rather than solely comparing pixels.

Several no-reference approaches utilize machine learning methods to predict segmentation quality. Some methods use segmentation properties such as image edges, boundary, and coverage to evaluate quality [12], but this approach was designed for medical images that have their own unique characteristics. In contrast our approach works for general images. Other methods use only the original and segmented image and implement neural networks for determining the quality. Examples include neural networks [13,19], transfer learning [13], a three cascaded layer of double nets with five convolutional layers [19]. However, these methods are not developed for resource-constrained mobile devices, resulting in high storage and execution time, which is not suitable for smartphones. In addition to the segmented and the original image, some other no-reference neural network methods [10] assume the presence of spatial uncertainty maps. Yet other neural network methods such as [6] predict a newly defined quality metric. Instead, we adopt the commonly used evaluation metric Intersection over Union (IOU) and do not assume the presence of spatial uncertainty maps.

Segmentation can be done for single or multiple regions. Thus, quality can also be determined for single or multiple regions. For example, [5] considers the quality of only the region of interest to the viewers. Our work considers the quality of multiple-region segmentation. Unlike single region segmentation, which only considers the proper segmentation of a single object, multiple region segmentation considers multiple objects as well as the background.

No-reference segmentation quality evaluation that uses supervised learning algorithms can further be categorized as classification and regression methods. Classification methods [11,13] determine the quality as 'Good', 'Ok', and 'Bad'. They determine the quality by setting a threshold for each class in the method (e.g.,'Good' is from 0.8 to 1.0, 'Ok' is from 0.4 to 0.8). These thresholds vary from user to user and are specific to applications. In contrast, regression methods [12, 19] generally return a value from 0 to 1, with 0 being the worst quality. Thus, we consider the quality of segmentation as a regression problem in this paper.

3 Neural Network-Based Evaluation of Image Segmentation Quality

In this section, we present our design of Segmentation Quality Network (SegQNet) that evaluates the quality of segmented images overlaid in AR scenes.

SegQNet adopts CNN [15], a commonly used neural network for letter identification, object detection, and the optical flow of images. A CNN has an input layer, a hidden layer followed by fully connected layers, and an output layer. The input layer consists of images to be processed. The hidden layer performs convolution operations on these images using a kernel of fixed size. Fully connected layers are then used for invariance and fine-tuning of the network. The output layer provides the final result as a value in our quality evaluation.

SegQNet (Fig. 1) consists of two convolutional layers and two fully connected layers. We added an adaptive average pool layer to make sure that the network can handle images of different sizes. In addition, we increase the kernel size of the network from 5×5 to 10×10 for performing a reduced number of operations. We next present the details of the input and output layers in SegQNet.

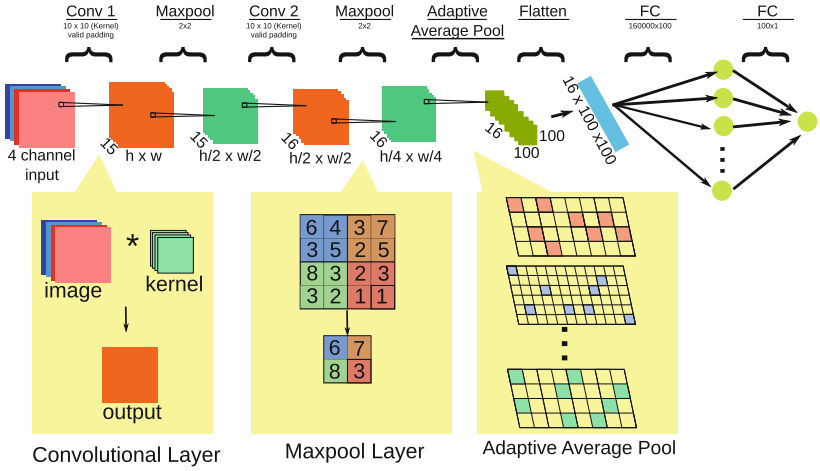


Fig. 1. Neural Network Architecture of SegQNet. The top part shows the overall architecture and the bottom part shows a snippet of the working of each layer.

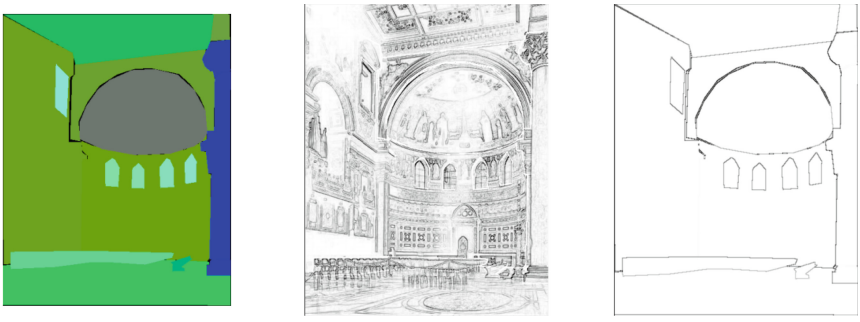
Input Layer. Since the segmentation of an image chooses which edges to keep, we use the edge images of both the original and the segmented image to determine the inputs to SegQNet. There are various methods to determine the edge of a given image, such as Sobel, Canny, and Laplacian filters. A Sobel filter works under the principle that the presence of an edge can be determined by the color changes in the adjacent pixels. The color changes are obtained by the gradients of an image (a.k.a. edge intensity), which is the difference between the adjacent pixel values. We use the edge intensity to determine whether an edge is present or not. We use a Sobel Filter instead of Canny or Laplacian filters. This is because Canny or Laplacian filters operate on the output of the Sobel filter and choose the most probable edges where misclassification may occur.

The Sobel filter is applied to both the original and segmented images to obtain their edge intensity and orientation. The edge orientation, which refers

to the direction of the gradient, is also used as an input to SegQNet. We expect the orientation of the segmented edge and the original edge to be nearly identical with proper segmentation.

The difference between the edge intensities of the original and segmented image and vice versa are calculated. Figures 2 and 3 show the difference between the original and the segmented image and vice versa. The difference in edge intensity between the original and segmented images could be zero, positive, or negative. Zero difference suggests the presence of edges in both the original and segmented images. A positive difference indicates an edge exists in the original image but not in the segmented image. A negative difference indicates the presence of an edge in the segmented image but not in the original image. Since this negative difference becomes positive when we compute the difference between the edge intensities of the segmented and the original images, we discard negative differences to simplify the representation.

Figures 2b and 3b show examples of white pixels, which indicate the absence of edges or edges in the segmented image that were not present in the original image. In Fig. 3b, a white line indicates an extra edge in the Dilated image, that should actually not be present (Inside the block shown in red). By comparing Figs. 2b and 2c, we can see that the intersection of all white pixels represents places where there are no edges, while the difference represents the extra edges. We use these two inputs with the orientation and feed them to the CNN which has the ability to utilize spatial correlation to learn the quality.



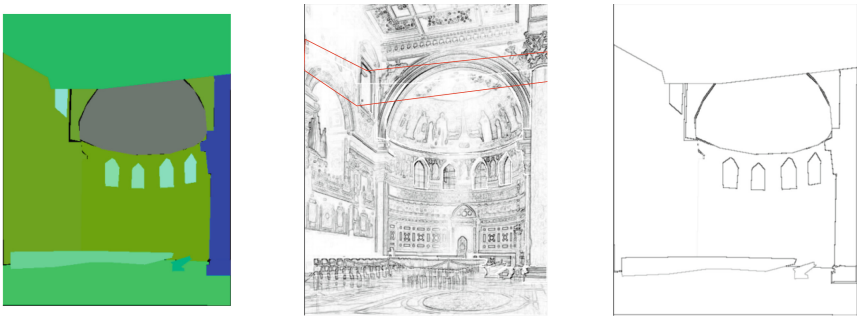
(a) Original image segmented

(b) original_sobel - segmented_sobel

(c) segemented_sobel - original_sobel

Fig. 2. Input to SegQNet when the ground truth image segmented is given as input where original_sobel is the original image edge intensity and segmented_sobel is the segmented image edge intensity ((b) and (c) are inverted for better visualization)

Based on earlier studies [23], it has been observed that neural networks primarily learn low-level characteristics such as edges and colors in the initial layers. Given that our input already utilizes edge images, we can employ a neural network with fewer layers. Moreover, our proposed methodology of finding the difference between edge intensity of the original and segmented images enables



(a) Dilated image segmented

(b) original_sobel - dilated_sobel

(c) dilated_sobel - original_sobel

Fig. 3. Input to SegQNet when the dilated image segmented is given as the input, where original_sobel is the original image edge intensity and segmented_sobel is the segmented image edge intensity ((b) and (c) are inverted for better visualization)

SegQNet to identify the existence of extra edges, which is an essential aspect in evaluating image quality. Hence, we constructed SegQNet utilizing a simpler architecture like LeNet.

In summary, we provide the original image edge orientation, the segmented image edge orientation, the original image edge intensity subtracted from the segmented edge image intensity, and the segmented image edge intensity subtracted from the original image edge intensity as inputs to SegQNet.

Output Layer. This layer provides a single value as the image quality. Traditional metrics for segmentation quality are DICE (Jaccard Score) [9], precision and recall, Mask Intersection over Union (IoU) [21], and Boundary IOU [8]. More recent research has designed a new metric that considers human visual perception. However, all these methods are full-reference method (i.e., need ground truth images). Therefore, we decided to use IOU, the most commonly used metric [2], which ratio of the number of pixels in the segmented image with errors introduced that match the ground truth image over the total number of pixels in both images.

4 Performance Evaluation on a PC with GPU

We first evaluate the performance of SegQNet on a PC with GPU and compare that with baseline approaches. We utilize the ADE20k dataset [1,24,25] in our evaluation. This dataset includes ground truth segmented images as well as original images. We selected the ADE20k dataset also because it contains multi-object segmentation as well as both foreground and background segmentation. Furthermore, this dataset offers extensive coverage of both indoor and outdoor images, which is where AR is predominantly used. The dataset contains about 434,826 object instances, 2,693 object classes, and 9.9 object classes per image. ADE20k dataset has 20,210 images in the training set, 2,000 images in the val-

idation set, and 3,000 images in the test set. We use the same training-test set partition as specified in the dataset for our evaluation.

When considering the right approach to compare against, we do not consider the neural network algorithms [11,13] that treat quality analysis as a classification problem. This is because we treat quality evaluation as a regression problem and we would like to provide a quantitative value for the quality. Methods [6] that define a new quality metric are not fair to be compared against, either. Therefore, we choose two most relevant algorithms as our baselines: (1) MultiScale Net [19] that uses a neural network model, and (2) Pull the Plug [12] that uses a traditional machine learning model.

- **MultiScale Net.** Two neural network-based models [19] was developed for the quality evaluation of segmented images. The first is a double-network model that has two cascaded VGG16 nets with the original and segmented images as inputs. The second is a MultiScale Net with six or eight cascaded VGG16 layers with original and segmented images in the actual and also cropped sizes as inputs. The results demonstrate the superiority of the MultiScale Net over the double-net model, hence we implement the MultiScale Net for our comparison. However, Multiscale Net uses VGG architecture that can only accept images of one size (i.e., 224×224). Images of 224×224 and the corresponding segmented images are fed to the first two cascaded layers. Also, according to the original Multiscale Net paper, for the other four layers, the original and the segmented image must be cropped first and then interpolated back to 224×224 for VGG. We utilize the same details as described in the paper in our implementation for fairness of comparison. Since their paper does not mention the size of the smaller images, we consider the common cropping sizes, 112×112 and 168×168. It is important to note that, we use the same images as input to our SegQNet, but do not make modifications to the size of the image as our network can work with images of different sizes.
- **Pull the Plug.** Pull the Plug [12] was developed for a single object in the foreground. In contrast, our work considers multiple object segmentation. For a fair comparison, we randomly choose one object out of all the objects in the images for quality evaluation.
  In addition, Pull the Plug considers a bounding box and convex hull around the segmented object. While this is valid for most of the error types considered, it is not always true for erosion. As applying erosion to a small object can make the object disappear completely from the scene. The diminished object creates a bounding box and convex hull with a minimal area close to zero. Therefore, we do not include these sets of images in our experiments of Pull the Plug.

For comparison, we generate negative samples by modifying the segmented images to include the six common types of errors described previously. We trained and tested all three algorithms (i.e., our SegQNet, MultiScale Net, and Pull the Plug) using NVIDIA 3090Ti GPU and 20-core Intel I7 processor.

We trained SegQNet on the different types of errors in segmentation. To provide a reasonable comparison, we trained MultiScale neural network and Pull

the Plug as mentioned in their papers. Table 1 compares the average error of the three different quality evaluation methods: SegQNet, MultiScale Net, and Pull the Plug.

Our findings demonstrate that SegQNet's accuracy in evaluating image quality is equivalent to MultiScale Net, but surpasses Pull the Plug by a significant margin. However, it is critical to consider the complexity of the neural network models utilized, specifically MultiScale Net and SegQNet. SegQNet achieves comparable accuracy to MultiScale Net for an input image size of 224×224 (a commonly used size in machine learning algorithms), but with much lower memory requirements (30 times less), a smaller number of trainable parameters (10 times less), and substantially fewer GFLOPs (45 times less) as shown in Table 2. We attribute the better performance of SegQNet mainly to the inherent differences in the architecture. Multiscale Net uses many layers and is likely to determine the edge of the images after initial layers [23] and hence requires more layers. On the other hand, in our approach, edge detection is done first, hence SegQNet requires fewer layers, resulting in smaller GFLOPS, and less memory requirement.

Table 1. Average error of quality evaluation

Method	Average Error
SegQNet	0.07
MultiScale Net	0.07
Pull the Plug	0.41

Table 2. Complexity of the Neural Network models of size 224×224

Method	Total Memory(MB)	Total trainable params	No. of mult-add (GFLOP)
MultiScale Net	2164.64	214,967,169	92.626
SegQNet	71.70	16,030,232	2.11

Further, as shown in Table 3, when processing images with resolutions similar to those used by ARCore (640×480) [3], SegQNet has lower complexity compared to MultiScale Net.

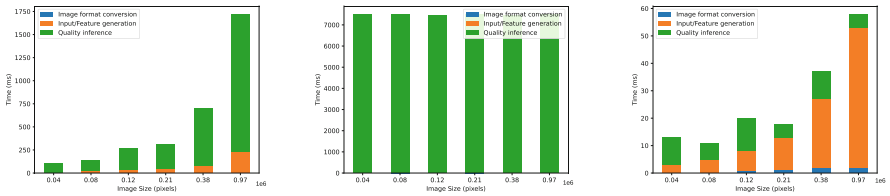
Table 3. Complexity of the Neural Network models of size 640×480

Method	Total Memory(MB)	Total trainable params	No. of mult-add (GFLOP)
MultiScale Net	2164.64	214,967,169	92.626
SegQNet	113.55	16,030,232	5.05

5 Performance Evaluation on a Smartphone

Since the main aim of this work is to enable quality evaluation for mobile AR, we implemented SegQNet, MultiScale Net, and Pull the Plug on OnePlus 6T smartphones. We used DeepLabv3 [7] provided by PyTorch for image segmentation

first before running each of the quality evaluation methods. To ensure that the neural network loaded onto phones is lightweight, we utilize various optimization techniques such as quantization, pruning, operator fusion, kernel optimization, and model compression on both SegQNet and MultiScale Net. These methods are implemented in PyTorch Lite and do not compromise the accuracy of the neural network. These optimizations are crucial for enabling the models to run efficiently on mobile devices without sacrificing their performance.



(a) SegQNet (b) MultiScale Net (c) Pull the Plug

Fig. 4. Latency breakdown comparison

Latency. Fig. 5a and Fig. 5b highlight that the time taken to execute the quality evaluation utilizing SegQNet is roughly 83.3 times shorter than the segmentation process itself. This result is significant since it is undesirable for quality evaluation to consume more time than the segmentation algorithm itself.

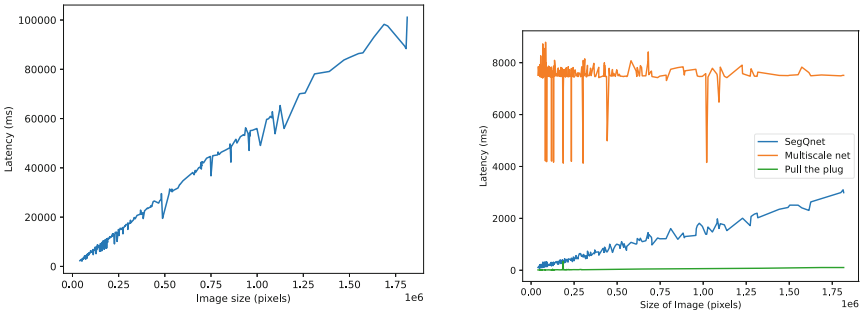
Figure 5b also shows that the time taken by SegQNet is about seven times less than MultiScale Net, so SegQNet can achieve comparable performance as MultiScale Net but with much less time. In fact, MultiScale Net takes more time than the segmentation itself for images of smaller sizes. However, SegQNet takes more time than Pull the Plug, especially for larger images. This can be attributed to the number of convolution operations performed in SegQNet. However, the additional few milliseconds of the extra time taken by SegQNet significantly improves its accuracy, i.e., by six times when compared with Pull the Plug as shown in Table 1. By examining Fig. 5b, it is evident that SegQNet's quality determination process takes up to 3 s for a few large images. However, these images are typically high-resolution and not frequently used. In fact, the MultiScale Net architecture decreases all images to 224×224 using VGG16. By restricting the image size to this small dimension, SegQNet requires less than 250 milliseconds to assess the quality. Additionally, when we consider the resolution employed by ARCore to process video for AR applications, SegQNet's execution time is less than 300 milliseconds.

Figure 4a shows the breakdown of the time for quality evaluation. It takes very little time to convert Bitmap to 'Mat'. This processing time is incurred because of the need to convert from Bitmap, the type for reading an image in Java, to 'Mat', the required type for OpenCV, and back to Bitmap, the type required for PyTorch. Most of the processing time in SegQNet is dedicated to the neural network and the computation of edge intensity and orientation of

images. Due to the neural network's significant role and the absence of image scaling, the SegQNet processing time grows as the input image size increases, as the number of convolution operations also increases.

Figure 4b depicts the breakdown of the time to run MultiScale Net. This network has the same conversion latency as SegQNet. The 'Input/feature generation' is the time taken to rescale the images to the size required by the network. Most of the time taken by this approach is to run the network.

Figure 4c is the breakdown for time to run Pull the Plug. In this case, more amount of time is taken to get the features than to run the trained model. This is because to get the features, the algorithm needs to traverse through the image, whereas for deriving inference, it runs through the linear regression model. It is also intuitive that the latency to calculate the features increases with the size of the image.



(a) (b)

Fig. 5. (a) Image segmentation Latency (b) Quality evaluation Latency

Energy consumption. Battery usage for performing segmentation for 1500 images uses about 95% of the smartphone battery, this involves rendering the image to the screen as well. According to the paper [20], the maximum amount of energy/battery utilization in an AR app is due to camera sampling and rendering. The battery utilized to render the image to the screen is not the consideration of the paper, and hence we measure the battery utilization after removing the rendering. Table 4 compares the battery utilization of three different quality evaluation methods for 1,000 images. SegQNet battery utilization is comparable to Pull the Plug, and it is about 12 times less than that of MultiScale Net.

CPU Usage. We recorded the total CPU usage for 1,000 images and observed that SegQNet used 50% of CPU for a short period of time in the beginning and then stayed at about 30%. The similarity of CPU usage is because SegQNet and MultiScale Net do not vary in the complexity of each operation performed but vary only in the number of instances the operations are performed. However, there is a clear difference in the total latency involved. The time duration that the CPU runs at 30% for SegQNet is similar to Pull the Plug but significantly less than MultiScale Net.

Table 4. Battery Utilisation of quality evaluation

Method	Battery Utilization
SegQNet	4 %
MultiScale Net	48 %
Pull the Plug	1 %

Memory Usage. The models trained by the neural network are optimized for phones and then stored in phones. After the training, the memory required by SegQNet is about 60 MB, which is about 14 times less than MultiScale Net. We applied a quantization process to reduce the model size of MultiScale Net, but it still takes about 500 MB, which is quite demanding for a smartphone. We use ONNX [4] to store the Linear Regression model of Pull the Plug, and it takes only 600 B. Therefore, Pull the Plug takes the least space among the three, but it is also at the cost of the lowest accuracy in quality evaluation.

6 Conclusion

In this paper, we presented our techniques for determining the quality of overlaid segmented images in mobile AR apps. We determine the quality based on the most common six types of errors that may appear in the segmentation: dilation, erosion, object localization, boundary approximation, boundary localization, and inner mask errors. We developed SegQNet, a lightweight CNN that has an average error of 0.07. Our experiments show that SegQNet achieves approximately six times less absolute error when compared to Pull the Plug. While SegQNet shows comparable accuracy with MultiScale Net, SegQNet incurs much less overhead. Experiments on smartphones demonstrate that overall SegQNet outperforms existing approaches when considering performance and cost trade-offs. SegQNet cannot determine under- or over-segmentation. This is acceptable considering that recent advances in segmentation techniques [8] have eliminated under- or over-segmentation as a type of error. With SegQNet being fast and resource-efficient, We are in the process of developing a framework that integrates real-time quality evaluation with adaptive segmentation in mobile AR.

References

1. Ade20k dataset. https://groups.csail.mit.edu/vision/datasets/ADE20K/
2. Papers with code - semantic segmentation. https://paperswithcode.com/task/semantic-segmentation

3. Recording and playback introduction, arcore, google developers. https://developers.google.com/ar/develop/recording-and-playback
4. Sklearn-onnx: Convert your scikit-learn model into onnx, https://onnx.ai/sklearn-onnx/
5. Cai, Z., Liang, Y., Huang, H.: Unsupervised segmentation evaluation: an edge-based method. Multimedia Tools Appl. **76**(8), 11097–11110 (2017)
6. Chen, H., Peng, B., Al-Huda, Z., Li, X.: Metric learning with feature embedding for segmentation quality evaluation. In: 2021 16th International Conference on Intelligent Systems and Knowledge Engineering (ISKE), pp. 269–275. IEEE (2021)
7. Chen, L.C., Papandreou, G., Schroff, F., Adam, H.: Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587 (2017)
8. Cheng, B., Girshick, R., Dollár, P., Berg, A.C., Kirillov, A.: Boundary IOU: Improving object-centric image segmentation evaluation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 15334–15342 (2021)
9. Crum, W.R., Camara, O., Hill, D.L.: Generalized overlap measures for evaluation and validation in medical image analysis. IEEE Trans. Med. Imaging **25**(11), 1451–1461 (2006)
10. DeVries, T., Taylor, G.W.: Leveraging uncertainty estimates for predicting segmentation quality. arXiv preprint arXiv:1807.00502 (2018)
11. Feng, T., Peng, B., Al-Huda, Z., Jia, H.: Segmentation quality evaluation algorithm based on classification. In: 2021 16th International Conference on Intelligent Systems and Knowledge Engineering (ISKE), pp. 254–261 (2021). https://doi.org/10.1109/ISKE54062.2021.9755428
12. Gurari, D., Jain, S., Betke, M., Grauman, K.: Pull the plug? predicting if computers or humans should segment images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 382–391 (2016)
13. Huang, C., Wu, Q., Meng, F.: Qualitynet: segmentation quality evaluation with deep convolutional networks. In: 2016 Visual Communications and Image Processing (VCIP), pp. 1–4. IEEE (2016)
14. Kim, W., Seok, J.: Indoor semantic segmentation for robot navigating on mobile. In: 2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN), pp. 22–25. IEEE (2018)
15. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
16. Li, X., Tian, Y., Zhang, F., Quan, S., Xu, Y.: Object detection in the context of mobile augmented reality. In: 2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR), pp. 156–163. IEEE (2020)
17. Perry, J., Fernandez, A.: MineNet: a dilated CNN for semantic segmentation of eye features. In: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (2019)
18. Shi, R., Ma, J., Ngan, K.N., Xiong, J., Qiao, T.: Objective object segmentation visual quality evaluation: quality measure and pooling method. ACM Trans. Multimedia Comput. Commun. Appl. (TOMM) **18**(3), 1–19 (2022)
19. Shi, W., Meng, F., Wu, Q.: Segmentation quality evaluation based on multi-scale convolutional neural networks. In: 2017 IEEE Visual Communications and Image Processing (VCIP), pp. 1–4. IEEE (2017)
20. Wang, H., Xie, J.: User preference based energy-aware mobile AR system with edge computing. In: IEEE INFOCOM 2020-IEEE Conference on Computer Communications, pp. 1379–1388. IEEE (2020)

21. Wang, Z., Wang, E., Zhu, Y.: Image segmentation evaluation: a survey of methods. Artif. Intell. Rev. **53**(8), 5637–5674 (2020)
22. Zaman, F.A., Zhang, L., Zhang, H., Sonka, M., Wu, X.: Segmentation quality assessment by automated detection of erroneous surface regions in medical images. In: Computers in Biology and Medicine (2023)
23. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8689, pp. 818–833. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10590-1_53
24. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ade20k dataset. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 633–641 (2017)
25. Zhou, B., et al.: Semantic understanding of scenes through the ade20k dataset. Int. J. Comput. Vision **127**(3), 302–321 (2019)

Reading the Minds Eye: Detecting Trauma-Vulnerability in Individuals by Analyzing Attention Through Eye-Tracking

Taseen Mubassira, Mehedi Hasan(✉), and Sadia Sharmin

Bangladesh University of Engineering and Technology, Dhaka, Bangladesh
mhasan912@gmail.com, sadia@teacher.cse.buet.ac.bd

Abstract. In this work, we propose a novel approach for detecting and predicting trauma vulnerability in individuals by monitoring their attention to threat images using eye-tracking. We utilized a combination of statistical analysis, machine learning models and cognitive psychological theories. To do so, initially, an online survey was conducted with 183 participants to gather information about individuals' previous experiences with trauma. Based on the survey results, an image set was prepared, consisting of four categories: positive, neutral, general threat and trauma-relevant threat. 39 volunteers, categorized into trauma-exposed and non-trauma-exposed groups based on clinical diagnosis and previous treatment, then participated in an eye-tracking experiment, where they were presented with 10 slides containing 4 images from each category; each slide lasting for 6 s. Eye gaze patterns were recorded and processed during this period, which was used to extract different features that were later used for the machine learning models to train on. Our findings revealed that trauma-exposed group showed more attentional bias towards the specific trauma-relevant threat images than the non-trauma-exposed group. Results from training the models suggest that it is highly likely that predicting trauma vulnerability in individuals is possible and this demands further research.

Keywords: Trauma Detection · Eye Tracking · Machine Learning · Survey

1 Introduction

The importance of mental health cannot be overstated, as it has a direct impact on our overall well-being and the quality of our lives. It shapes our thoughts, emotions, and behaviors, playing a crucial role in how we handle stress, establish relationships, and navigate decisions. When mental health is neglected or compromised, it can lead to a range of challenges, including emotional instability, impaired cognitive function, strained relationships, and decreased productivity. Therefore, prioritizing mental health is essential for maintaining a

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 428–444, 2024.
https://doi.org/10.1007/978-3-031-63992-0_28

balanced and fulfilling life. Trauma, within the sphere of mental health, refers to distressing events or circumstances that profoundly disrupt an individual's emotional and psychological equilibrium. The study of trauma highlights the critical link between mental health and traumatic experiences. Trauma can have long-lasting effects on an individual's psychological well-being, often resulting in conditions such as PTSD, anxiety, and depression [13]. The psychological impact of trauma varies from person to person, and not everyone who experiences trauma will develop a diagnosable mental health disorder [4,15]. Asides mental health impacts, trauma can contribute to a range of physical symptoms, such as sleep disturbances, chronic pain, headaches, gastrointestinal issues, and compromised immune function [9]. These physical manifestations can further exacerbate the emotional and psychological distress associated with trauma. One study even found out that an enduring effect of trauma exposure can be passed to offspring transgenerationally, which affects the expression of genes [18].

A comprehensive study was conducted across 24 countries worldwide, encompassing a diverse group of 68,894 adults from six continents. The aim was to evaluate the exposure of individuals to various types of traumatic events, with a total of 29 categories considered. The findings indicated that more than 70% of respondents reported experiencing at least one traumatic event; 30.5% were exposed to four or more [1]. Additionally, a separate survey involving 10,641 adults in Australia revealed that 57% of the population had encountered a traumatic event at some point in their lives. Notably, the study unveiled that men had a higher likelihood of experiencing most types of trauma, including multiple traumas, except in cases of sexual assault. Furthermore, the research highlighted that factors such as being female, younger in age, having limited educational background and residing in economically disadvantaged areas were associated with an increased probability of developing post-traumatic stress disorder (PTSD) after experiencing trauma. However, when considering a combination of factors, the specific nature of the trauma itself, particularly instances like sexual assault, emerged as a more influential predictor of PTSD development [10].

When it comes to trauma, individuals react and respond in unique ways, demonstrating a range of vulnerabilities and resilience. The concept of trauma vulnerability acknowledges that not everyone responds to traumatic events in the same manner [2]. Understanding why some people are more susceptible to the negative effects of trauma while others exhibit greater resilience requires consideration of various factors. It is important to emphasize that trauma vulnerability does not imply weakness or deficiency. It is a recognition that individuals have different strengths, resources, and capacities to cope with and recover from traumatic events. Understanding trauma vulnerability can inform the development of targeted interventions and support systems that help mitigate the negative effects of trauma and promote healing and resilience. The motivation of this study was to design a state of the art method that would be able to identify trauma-vulnerability in individuals, which would help in early intervention and prevention of traumatization, building resilience and coping skills and preventing long-term societal costs.

Numerous studies have demonstrated the connection between trauma and visual content. One of the recent studies showed that, a perceptual processing bias for trauma related stimuli may contribute to the involuntary triggering of intrusive trauma memories in PTSD [7]. In this context, understanding trauma vulnerability becomes crucial as it not only contributes to our comprehension of how individuals respond to digital stimuli but also informs the development of targeted interventions and support systems. Trauma, often exacerbated or triggered by digital content, can have far-reaching psychological implications, making it imperative to delve into its dynamics to ensure a healthier digital experience for all.

1.1 Motivation

In the context of Bangladesh, a developing nation with very limited awareness of mental health issues, a substantial number of trauma cases remain undiagnosed. Consequently, mental health disorders persist unchecked, often manifesting as conditions such as anger and depression. Even when overt signs of trauma exposure become apparent, individuals often hesitate to seek treatment due to the financial constraints associated with therapy and medication. Furthermore, discussing mental health remains a social taboo in some communities, making it challenging to access professional help. Against this backdrop, the primary driving force for this research endeavor is to investigate the feasibility of preemptively identifying individuals susceptible to specific types of trauma, with the aim of enabling early intervention to prevent the development of Post-Traumatic Stress Disorder (PTSD). Moreover, the methodology employed for this purpose is designed to be cost-effective, ensuring accessibility for a broad spectrum of the population, particularly in the resource-constrained context of a developing country like Bangladesh.

This study explores the following research questions in the context of Bangladesh.

- **RQ1:** Is it feasible to ascertain whether an unassuming individual, without a prior diagnosis, is predisposed to a specific form of trauma?
- **RQ2:** Is it possible to use visual attention to do this task?
- **RQ3:** What level of accuracy can be attained using a cost-efficient approach for accomplishing this?

This study aims to answer these queries by leveraging the innate human inclination to unconsciously direct attention towards visual stimuli. By combining data from both our survey and experimental phase, we train machine learning models to forecast an individual's vulnerability to trauma.

1.2 Attentional Bias

Visual attention is a cognitive process that allows individuals to selectively focus their visual perception on specific aspects of the environment while ignoring irrelevant or less salient information. It involves a combination of bottom-up processing, where attention is automatically captured by salient or novel stimuli,

and top-down processing, where attention is guided by internal goals, expectations, and prior knowledge. When presented with different visual stimuli, such as a series of images, individuals may demonstrate distinct attentional patterns. Each person's attentional biases are influenced by a range of factors, including their unique cognitive makeup, personality traits, interests, and emotional states. These biases result in variations in the allocation of attention and the duration of gaze toward specific visual elements [16]. Psychologically, attentional patterns can provide insightful information about an individual's cognitive and emotional processes. By analyzing these biases, we can gain a deeper understanding of cognitive mechanisms such as selective attention, perception, and memory. A lot of research has established that attentional bias is linked to an individual's threat processing [19]. The objective of the current study is to leverage this fact to try to detect trauma vulnerabilities in individuals.

2 Related Works

Numerous researches in the past have suggested attentional bias in people diagnosed with PTSD. In a study conducted by Wshah, Skalka, Price, et al. in 2019 [17], the researchers aimed to develop computational methods to effectively identify at-risk patients and provide better early interventions. The study demonstrated that an ensemble model, comprising a combination of support vector machines, naive Bayes, logistic regression, and random forest algorithms, accurately predicted elevated PTSD symptoms with an area under the curve (AUC) of 0.85. Remarkably, only seven self-reported items or features were necessary to achieve this AUC. The self-reported features were collected through surveys administered via smartphones.

In another study conducted by Kimble et al. [5], they explored the differences in attentional patterns between depression and post-traumatic stress based on relevant literature. The researchers tracked the participants' eye movements while they viewed a series of 30 neutral but ambiguous and complex pictures on a computer sceen. Their findings revealed that hypervigilance scores and negative views of the world were predictors of both the number of fixations and the are of the picture covered by the participants' gaze. Based on their findings, the researchers concluded that there are distinct gaze patterns associated with pos-ttraumatic stress but not depression following trauma.

Eye tracking technology was utilized by Kimble et al. [6] to examine visual orientation and attention patterns in 19 veterans of the Iraq war. The objective was to investigate their responses to traumatic and neutral stimuli. The results indicated that veterans who reported higher levels of PTSD symptoms exhibited larger pupil sizes in response to all negatively valenced pictures compared to veterans with lower PTSD symptoms. Additionally, those with higher PTSD symptoms spent more time fixating on the negatively valenced stimuli.

A more detailed study was conducted by Thomas et al. [12] in 2013. 55 individuals were divided into three groups: clinical PTSD diagnosed, subthreshold

symptoms of PTSD, and a group with no trauma exposure. The researchers aimed to investigate the attentional patterns of these groups by tracking and recording their eye gaze while viewing sets of four images presented for 6 s. The images were categorized into positive, neutral, negative, and either general threat or trauma-relevant threat images. The results revealed that both the clinical PTSD and subthreshold symptom groups exhibited a higher attentional bias towards trauma-relevant threat images compared to the group with no trauma exposure. However, no significant group differences were observed for general threat images. Additionally, a time course analysis of the attentional patterns was conducted. The analysis demonstrated that participants with clinical symptoms of PTSD displayed evidence of avoidance towards threat images. This finding suggests that individuals with clinical symptoms of PTSD tend to actively avoid engaging with trauma-relevant threat stimuli.

3 Methodology

Our study involved several sequential steps that aimed to explore the effects of traumatic experiences on individuals. The initial phase encompassed an online survey, which included a comprehensive questionnaire that was designed to gather a wide range of information from the participants. This questionnaire sought to gather general participant information while also delving into specific details concerning their past traumatic experiences. Subsequently, leveraging the gathered information on traumas, our focus shifted towards the creation of an image set specifically tailored to the study's objectives, which would serve as the stimulus to the experiment. With the image set prepared, we progressed to the main experiment, wherein we tracked and recorded participants' eye-gaze patterns while they viewed a slideshow of images. This system allowed us to record and analyze participants' gaze fixation points, saccades (rapid eye movements between fixations), and overall eye-gaze patterns during the image presentation. By analyzing the recorded eye-gaze patterns, we aimed to uncover patterns of visual exploration, including areas of heightened attention and potential avoidance behaviors, related to the traumatic experiences reported in the initial online survey. Lastly, we used the data collected to train and evaluate various predictive models to assess the effectiveness of these models in predicting whether an undiagnosed individual is vulnerable to a certain type of trauma (Fig. 1).

3.1 Online Survey

To gather relevant data, we developed a comprehensive online questionnaire. The questionnaire aimed to capture general information about individuals, such as their age, occupation, and living conditions. Additionally, we included specific inquiries regarding their past traumatic experiences such as domestic violence, road accident, animal attack, abortion/miscarriage, emotional abuse, robbery, childhood bullying, physical abuse, sudden unexpected death of loved one, sexual assault, suicide attempt, divorce etc. We also inquired about their engagement

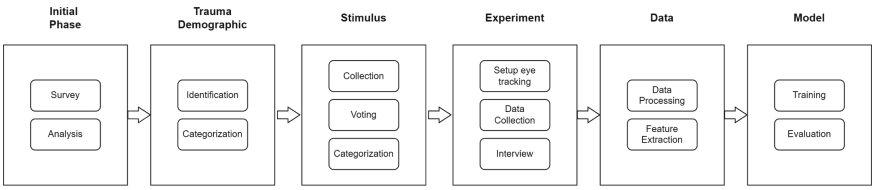


Fig. 1. Flow chart of methodology

in therapy or counseling to distinguish participants with a history of trauma exposure. Approximately 180 individuals participated in the survey (Figs. 2 and 3).

3.2 Stimuli

The stimulus set consisted of 40 images carefully selected from a larger collection. The selection process was guided by the results obtained from the initial survey. The images were categorized into four distinct categories: positive, neutral, general threat, and trauma-relevant threat. To determine the category assignment of each image, a smaller study was conducted in which individuals rated the images. The rankings provided valuable insights into how the images were perceived and were then appropriately categorized (Fig. 4).

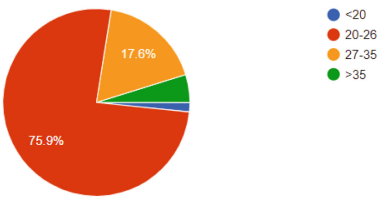
3.3 Participants

The study consisted of 39 individuals, comprising 21 males and 18 females. To facilitate the analysis, we divided the participants into two distinct groups:

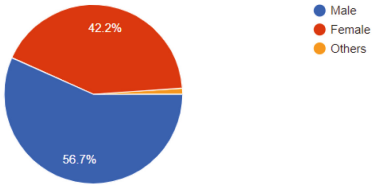
- **Trauma Exposed Group:** This group consisted of 18 participants who had experienced various forms of trauma. Among the, 5 individuals had been clinically diagnosed with trauma-related conditions and had received medication for their past traumas. Additionally, 13 participants had undergone some form of behavioral therapy at some point of their lives to address their trauma experiences.
- **Non-Trauma Exposed Group:** The remaining 21 participants had no significant history of trauma. These participants reported that they had not experienced any major traumatic incidents nor had they received any diagnoses related to trauma

3.4 Apparatus

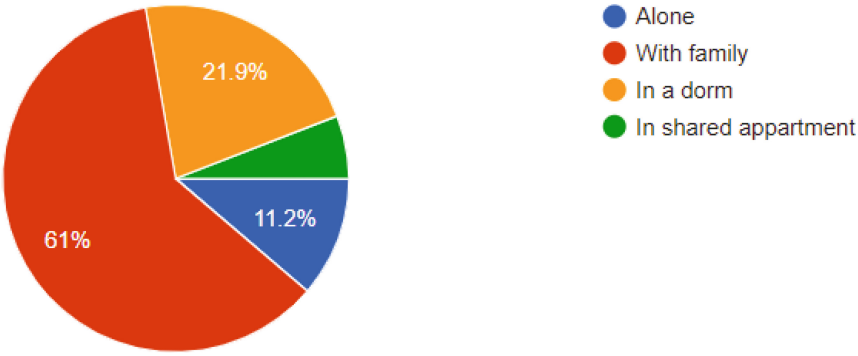
WebGazer [8], an online eye tracker that leverages the built-in webcams commonly found in laptops and mobile devices was used to track and record the participants' eye movements. It incorporates a self-calibration mechanism by observing how web visitors interact with the web page and trains a mapping between eye features and screen positions.



(a) Age Distribution



(b) Gender Distribution



(c) Living Condition

Fig. 2. Results from the survey

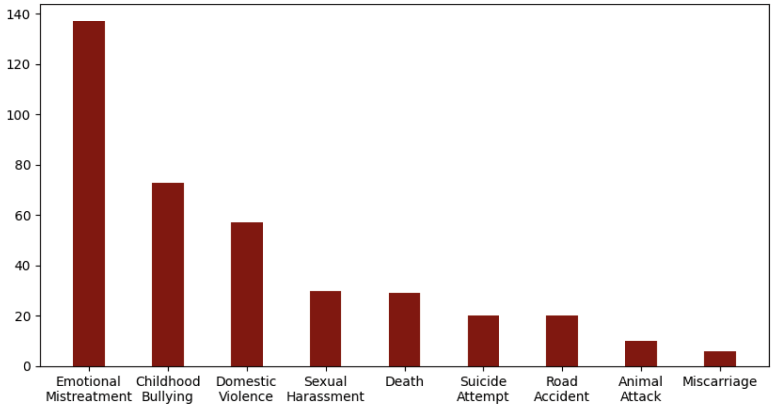


Fig. 3. Major Traumas in people

3.5 Experiment

Questionnaire. Prior to commencing the experiment, each participant was engaged in a preliminary questionnaire session. The purpose of this session was

(a) General Threat (b) Trauma Relevant

(c) Positive Image (d) Negative Image

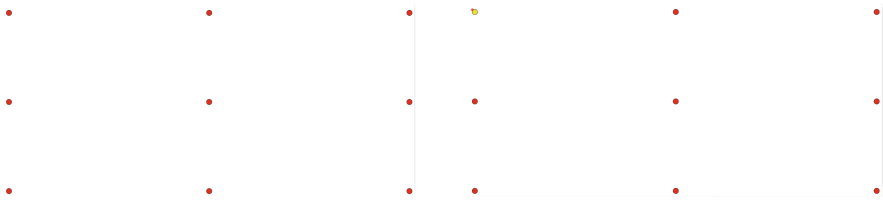
Fig. 4. Example of different categories of images

to gain insights into the participant's specific trauma history, the measures they had taken to address it, and how it had affected their life. This initial interaction aimed to establish a better understanding of each participant's unique experiences and provide context for the subsequent experiment.

Environment. Participants were positioned in front of a display screen with a with their faces directed towards a webcam mounted on the screen. The display screen's back faced a wall to ensure that there was no visual distractions behind the screen. Participants who wore corrective glasses were kindly requested to remove them for the duration of the experiment. This step was taken to enhance the precision of eye tracking measurements.

Calibration. The first phase of the experiment was a calibration process to establish accurate mapping between participants' eye positions and specific locations on the screen. A white screen displaying nine dots arranged across the screen was presented to the participants. They were then instructed to fixate their gaze on one of the red dots and proceed to register five mouse clicks on that particular dot. It was emphasized that they should refrain from blinking during the registration of clicks. Each click made by the participant was utilized to establish a precise mapping between their current eye position and the cor-

responding location of the dot on the screen. Once five clicks were successfully recorded, calibration for that specific point was accomplished. The same process was then repeated for the remaining eight dots on the screen (Fig. 5).



(a) 9 red dots on the screen (b) Top-left point is calibrated

Fig. 5. Calibration

Accuracy. To evaluate the accuracy of the calibration, additional mappings between participants' eye positions and specific regions on the screen were performed. The screen was divided into four distinct areas: top left, top right, bottom left and bottom right. Each of these regions was colored red for a duration of 3 s and the participants were requested to fixate their screen on that particular region. If the predicted participants gaze also fell on that region, the predicted gaze was considered accurate. The total time period where the predicted gaze was accurate was calculated to measure the accuracy of eye tracking. An average accuracy of 88%, with a peak accuracy of 95% was achieved during calibration. This validates the reliability of the eye tracking mechanism.

Procedure. The core component of the experiment involved presenting participants with a slideshow comprising four images per screen. A total of 10 slides were shown, resulting in a presentation of 40 images overall. Each screen consisted of four distinct categories of images: positive, general threat, trauma-relevant threat and neutral images. Each screen was displayed for a duration of 6 s, which is a little short of the human attention span [11].

3.6 Data Collection

For each of the four categories of images displayed, two key pieces of information was collected:

- **Looked At Times:** The timestamps when the gaze point entered the image
- **Looked Away Times:** The timestamps when the gaze point moved away from the image

Based on these two timestamps, various parameters were measured that would later be used for the training purposes of the models. Each set of images yielded seven specific features; with four categories of images, a total of 28 features were extracted. The seven features for each set of images are:

- *Total Fixation Duration:* Cumulative duration during which participants' gaze remains fixated on the particular image
- *Total Fixation Count:* The number of times participants' gaze fixates on that specific image. It represents the frequency of attentional shifts towards the image.
- *Average Fixation Duration:* Mean duration of individual fixations on that particular image. A longer average fixation duration suggests that participants are more engaged or immersed in the content of the image, whereas a shorted duration may indicate a more rapid scanning or less focused attention.
- *Average Gap Duration Between each Fixation:* The temporal intervals or pauses that occur during participants' gaze transitions within the image. A shorter average gap suggests that participants quickly move their gaze from one fixation to the next without much interruption which further suggests that participants look at the specific image more quickly which could be a result of potential bias towards that image.
- *Attention Over Time:* The 6-second time window was divided into three equal time windows of 2 seconds each:
  - Initial (0–2 seconds): This captures the initial fixation point of each participant, providing insights into the image that initially draws their attention
  - Middle (2–4 seconds): This captures the shifting of participants' eye gaze between different images after the initial attraction.
  - Final (4–6 seconds): This captures the last point of attention for each participant.

4 Results

The variables were the seven features for each category of images. Firstly, the data was divided into two groups, each corresponding to either the trauma exposed group or the non-trauma exposed group. The variables were calculated by taking an average of all the participants' data in each group. Then, the variables were summed over all the four types of category of images to make a comparison between them.

4.1 Percentage of Total Fixation Duration of Each Image Type

Table 1 presents the total fixation duration of two distinct groups of individuals towards the four image categories. The attention distribution of the trauma exposed group to these categories is displayed in the third column. The trauma exposed group allocated approximately 31.2% of their fixation time

to trauma relevant threat images, which is the highest proportion among the categories. Conversely, the non-trauma exposed individuals exhibited a lower attentional focus on trauma relevant threat images, with only 24.4% of their fixation time directed towards this category This pattern of attentional bias is consistent with the findings of existing literature on the topic [12]. Furthermore, the most attended category for the non-trauma exposed group is the positive images, accounting for 28.6% of their total fixation time. Interestingly, the trauma exposed individuals also displayed considerable attention towards positive images, with 27.5% of their fixation time directed to this category, making it the second most attended category for them. This suggests a bias towards positive images in both groups of people. However, for the non-trauma exposed individuals, the difference in total fixations between positive and general threat images is minimal, only 1.1%. This indicates that their attention is evenly divided between general threat and positive images. On the other hand, the trauma exposed individuals exhibit a greater disparity in fixation times between these categories. This finding suggests that non-trauma exposed individuals display a bias toward negative images [14]. In both cases, participants showed the least amount of fixation time on neutral images.

Table 1. Percentage of Total Fixation Duration

Image Category	Trauma Exposed	Non-trauma Exposed
Positive	27.5%	28.6%
Neutral	16.7%	19.5%
General Threat	24.6%	27.5%
Trauma Relevant Threat	31.2%	24.4%

4.2 Percentage of Average Fixation Duration

In Table 2, when ranking the images based on the average fixation duration of the trauma-exposed group, we observed that trauma-relevant threat images had the highest average duration (30.9%), followed by general threat images (26.1%), positive images (23.5%), and neutral images (19.5%). This ranking contradicts the ranking based on total fixation duration, where positive images ranked second and general threat images ranked third. However, the attentional pattern of the non-trauma-exposed group remained consistent when comparing average fixation duration and total fixation duration. The non-trauma-exposed group exhibited an attentional pattern in a non-increasing sequence, with general threat images receiving the highest average fixation duration (27.9%), followed by positive images (26.4%), trauma-relevant threat images (25.6%), and neutral images (20.1%).

Table 2. Percentage of Average Fixation Duration

Image Category	Trauma Exposed	Non-trauma Exposed
Positive	23.5%	26.4%
Neutral	19.5%	20.1%
General Threat	26.1%	27.9%
Trauma Relevant Threat	30.9%	25.6%

4.3 Percentage of Total Fixation Count

The total fixation count refers to the cumulative number of times participants fixated on a specific category of image throughout the duration of the experiment. Table 3 displays the percentage of the average fixation count per screen. The trauma exposed group had a 33.3% fixation count for positive images and 16.7% for general threat images. This suggests that the trauma exposed group tended to revisit positive images more frequently. This helps explain why the average fixation duration for general threat images was higher compared to the average fixation duration for positive images in this group. In the case of the trauma exposed group, the average fixation count for positive images is higher than that for general threat images. As a result, the average fixation duration for positive images is shorter compared to the average fixation duration for general threat images in this group. This could suggest that although the trauma exposed group spent more overall time looking at positive images, *they had a harder time diverting their attention away from general threat images each time they fixated on them.*

Table 3. Percentage of Total Fixation Count in a screen

Image Category	Trauma Exposed	Non-trauma Exposed
Positive	33.3%	16.7%
Neutral	16.7%	33.3%
General Threat	16.7%	16.7%
Trauma Relevant Threat	33.3%	33.3%

5 Analysis

Table 4 shows the proportion of the fixations on smaller time windows of 2 s over the 6 s duration of a slideshow presentation in our experiment.

For the trauma exposed group of individuals; during the first 2 s, there was a significant bias towards general threat images (35.8%), whereas the percentage of attention towards trauma relevant threat images was only 26.3%, even

Table 4. Attention Over Time Span

Image Category	Trauma Exposed			Non-trauma Exposed		
	0–2 s	2–4 s	4–6 s	0–2 s	2–4 s	4–6 s
Positive	27.6%	26.2%	25.3%	26.6%	26.8%	28.4%
Neutral	10.3%	20.1%	14.1%	12.3%	19.4%	15.5%
General Threat	35.8%	29.3%	28.2%	36.4%	28.5%	30.9%
Trauma-relevant threat	26.3%	24.4%	32.4%	24.7%	25.3%	25.2%

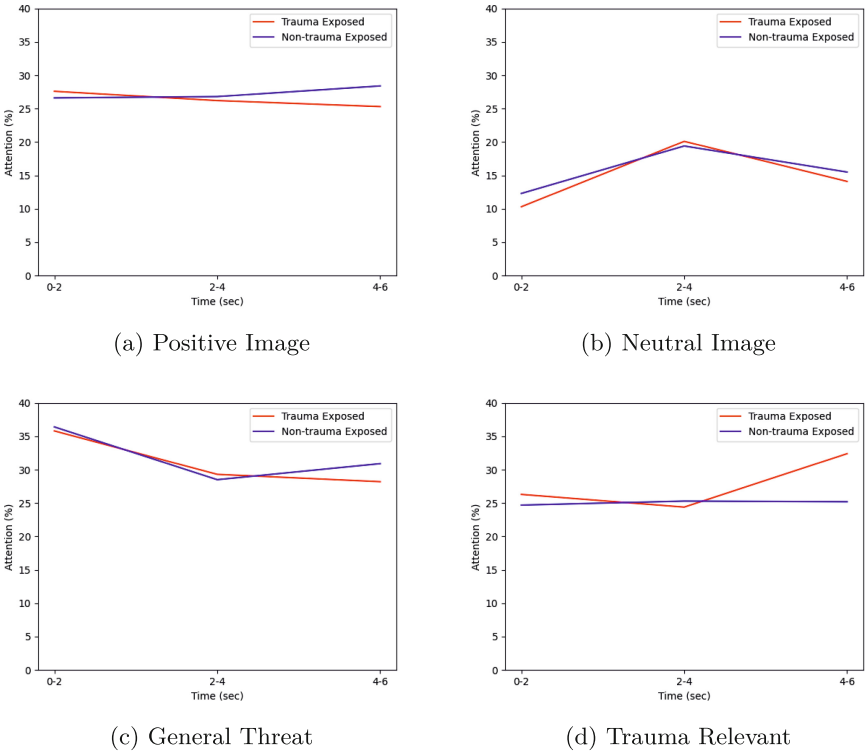
lower than that of positive images (27.6%). In the subsequent 2-second time window, the attention towards all categories decreased, but the attention towards neutral images increased significantly (20.1%). In the final time window of 2 s, the attention towards trauma relevant threat images increased significantly to 32.4%. Notably, the rankings of the categories at this time window align with the rankings of average fixation duration. This pattern might indicate that trauma exposed individuals initially showed avoidance towards trauma relevant threat images and exhibited a stronger fixation on general threat images. However, as time progressed, they looked over all the images on the screen, as reflected by the sudden rise in attention towards neutral images during the middle time window. Eventually, they directed most of their attention towards trauma relevant threat images in the final time window (Fig. 6).

One notable observation is that these individuals generally did not show any significant change in their attentional pattern across the different time windows for any of the categories. For instance, their attention towards positive images remained relatively consistent throughout the three time periods, with values of 26.6%, 26.8%, and 28.4%, which are all close to each other. Similar consistency can be observed for most other image categories, except for neutral and general threat images. The attention towards neutral images increased during the middle time window of 2–4 seconds, rising from 12.3% to 19.4%. The attention towards general threat images decreased during this middle time window, declining from 36.4% to 28.5%.

By comparing the data from the trauma exposed and non-trauma exposed groups side by side, it can be inferred that general threat images were the first to capture attention in both groups. This observation aligns with the general notion of a negativity bias, which suggests that individuals tend to pay more attention to negative or threatening stimuli [14].

6 Models Training and Evaluation

The data were labeled into a positive class (trauma exposed) and negative class(non-trauma exposed). The aim was to train supervised learning models on these labeled datasets to develop a predictive model. This model would be able to take an unlabeled data point from an individual who has not been exposed



(a) Positive Image (b) Neutral Image

(c) General Threat (d) Trauma Relevant

Fig. 6. Attention distribution over time span

to trauma and determine whether their gaze patterns align more closely with the positive class (trauma exposed) or the negative class (non-trauma exposed). If the individual's gaze pattern exhibited similarities to the positive class, we could infer that they potentially possess vulnerability to trauma. This detection of trauma vulnerability constituted the primary focus of our research (Fig. 7).

6.1 Training and Validation

Due to the dataset being small, k-fold cross validation was used to mitigate overfitting of data [3]. The dataset was initially shuffled randomly and divided into k equal partitions. For each partition, it was held out as the validation data while the model was trained on the remaining $k-1$ partitions. The model's performance was then evaluated by measuring the error using the held-out data as validation. This process was repeated for each partition, ensuring that every partition was used as the validation data.

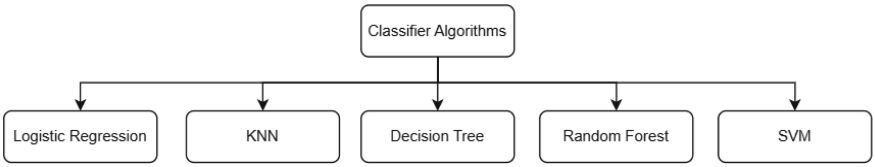


Fig. 7. Classification algorithms used

Table 5. Performance Metrics of the models

Classifier	Accuracy	Precision	Recall
Logistic Regression	.60	.53	.55
KNN	.79	.75	.72
Decision Tree	.68	.65	.60
Random Forest	.80	.77	.78
SVM	.68	.62	.58

6.2 Evaluation

Table 5 presents a summary of the performance metrics for the different models used. As anticipated, logistic regression performed poorly, likely due to its suitability for linear datasets with simpler structures. Since the dataset had a large number of features, logistic regression struggled to yield satisfactory results. The most successful models based on the performance metrics were KNN and Random Forest. Although Random Forest outperformed KNN in all metrics, it was found that KNN performed equally well due to the dataset's inherent characteristics, where similar classes of data tend to cluster together. Even though the models actually overfit the data, the results are suggestive that a predictive model could be used to match the attentional bias of an unsuspecting individual with the profile of a trauma exposed individual; this in turn could potentially be used to *predict whether an individual is vulnerable to certain types of trauma.*

7 Limitations and Future Work

The main limitation of the present study was the scarcity of data. To get a better evaluation of the models, it is necessary that those be trained on a large scale of data, taken from a diverse group of individuals. The data collected in the study was predominantly homogeneous, primarily comprising individuals within the age group of 20–26. This limited diversity may impact the generalization of the findings and their applicability to a broader population. Another limitation was related to the eye tracking method employed in the study. While cost-efficient, the accuracy of the eye tracking measurements was not optimal. This compromised the precision of data analysis. To address this issue, future research could

explore the use of more precise eye tracking instruments or techniques to enhance the accuracy of gaze tracking measurements.

Moving forward, it is crucial to overcome these limitations and address the gaps in the present study. One significant aspect to consider for further development of this work is the collection of a larger dataset. While the current dataset has provided valuable insights, expanding the sample size would offer a broader perspective and enhance the generalizability of the findings. A larger dataset allows for a more comprehensive representation of the target population, capturing a wider range of experiences, backgrounds, and demographics. Collecting data from a diverse set of population also provides the opportunity to explore subgroup analyses, investigating potential variations in trauma vulnerabilities based on factors such as age, gender, socioeconomic status, living conditions and even on culture.

Exploring alternative methodologies and technologies for eye tracking could improve the accuracy and reliability of the gaze data collected. This could involve utilizing more advanced eye tracking equipment or integrating multiple modalities for a comprehensive assessment of attentional responses. Another interesting avenue of research would be to focus on the prediction of specific types of trauma-vulnerability, rather than the general classification of individuals as trauma vulnerable. By focusing on the prediction of specific traumas, researchers and practitioners would gain insights into the prevalence and patterns of different types of traumatic experiences within a population.

A fascinating avenue of research would involve investigating the effectiveness of behavioral therapy in overcoming trauma. By analyzing the eye gaze patterns of individuals before and after undergoing behavioral therapy, it can be investigated whether there are any observable changes in their attentional responses to trauma-related stimuli.

8 Conclusion

The study proposed the utilization of various models aimed at identifying individuals who are vulnerable to trauma, employing a cost-efficient approach that relies solely on a webcam. By leveraging the capabilities of a webcam, the approach in this study offers a practical and cost-effective solution for assessing trauma susceptibility in individuals. Moreover, the findings highlight the neglect of trauma effects and the overall ignorance towards mental health in Bangladesh. The taboo surrounding discussions about mental health has led to many trauma-vulnerable individuals remaining undiagnosed and untreated, living their lives with significant distress. This quantitative analysis underscores the urgent need for trauma care and increased mental health awareness in Bangladesh.

References

1. Benjet, C., et al.: The epidemiology of traumatic event exposure worldwide: results from the world mental health survey consortium. Psychol. Med. **46**(2), 327–343 (2016)

2. Haehner, P., Bleidorn, W., Hopwood, C.: Why do people react differently to negative life events? individual differences in post-event changes in personality traits (2022)
3. Hawkins, D.M.: The problem of overfitting. J. Chem. Inf. Comput. Sci. **44**(1), 1–12 (2004)
4. Hidalgo, R.B., Davidson, J.R., et al.: Posttraumatic stress disorder: epidemiology and health-related considerations. J. Clin. Psychiatry **61**, 5–13 (2000)
5. Kimble, M., Cappello, O., Fleming, K.: Hypervigilance and depression as predictors of eye tracking to ambiguous pictures in trauma survivors. Int. J. Psychophysiol. **187**, 27–33 (2023)
6. Kimble, M.O., Fleming, K., Bandy, C., Kim, J., Zambetti, A.: Eye tracking and visual attention to threating stimuli in veterans of the IRAQ war. J. Anxiety Disord. **24**(3), 293–299 (2010)
7. Kleim, B., Ehring, T., Ehlers, A.: Perceptual processing advantages for trauma-related visual cues in post-traumatic stress disorder. Psychol. Med. **42**(1), 173–181 (2012)
8. Papoutsaki, A., Sangkloy, P., Laskey, J., Daskalova, N., Huang, J., Hays, J.: Webgazer: scalable webcam eye tracking using user interactions. In: Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI), pp. 3839–3845. AAAI (2016)
9. Pennebaker, J.W.: The effects of traumatic disclosure on physical and mental health: the values of writing and talking about upsetting events. Int. J. Emerg. Ment. Health **1**(1), 9–18 (1999)
10. Rosenman, S.: Trauma and posttraumatic stress disorder in Australia: findings in the population sample of the Australian national survey of mental health and wellbeing. Aust. NZ J. Psychiatry **36**(4), 515–520 (2002)
11. Subramanian, K.R.: Myth and mystery of shrinking attention span. Int. J. Trend Res. Dev. **5**(3), 1–6 (2018)
12. Thomas, C.L., Goegan, L.D., Newman, K.R., Arndt, J.E., Sears, C.R.: Attention to threat images in individuals with clinical and subthreshold symptoms of post-traumatic stress disorder. J. Anxiety Disord. **27**(5), 447–455 (2013)
13. Turner, R.J., Lloyd, D.A.: Lifetime traumas and mental health: the significance of cumulative adversity. J.Hhealth Soc. Behav. **36**, 360–376 (1995)
14. Veerapa, E., et al.: Attentional bias towards negative stimuli in healthy individuals and the effects of trait anxiety. Sci. Rep. **10**(1), 11826 (2020)
15. Voges, M.A., Romney, D.M.: Risk and resiliency factors in posttraumatic stress disorder. Ann. Gen Hosp. Psychiatry **2**(1), 1–9 (2003)
16. Wolfe, J.M.: Visual attention. Seeing pp. 335–386 (2000)
17. Wshah, S., Skalka, C., Price, M., et al.: Predicting posttraumatic stress disorder risk: a machine learning approach. JMIR Mental Health **6**(7), e13946 (2019)
18. Youssef, N.A., Lockwood, L., Su, S., Hao, G., Rutten, B.P.: The effects of trauma, with or without PTSD, on the transgenerational DNA methylation alterations in human offsprings. Brain Sci. **8**(5), 83 (2018)
19. Yu, Y., et al.: Mediating and moderating effects of attentional control on the relationship between personality traits and attentional bias among chinese medical students. Psychiatry Res. **246**, 113–118 (2016)

GUI2DSVec: Detecting Visual Design Smells Based on Semantic Embedding of GUI Images and Components

Bo Yang and Shanping Li(✉)

Zhejiang University, Hangzhou, China
{imyb,shan}@zju.edu.cn

Abstract. Visual graphical user interface testing (VGT) can simulate end-user behavior on essentially any device, and data-driven VGT approaches can automatically discern certain visual design imperfections. The visual information representing GUI images and the semantic information of components are crucial for data-driven design defect detection methodologies. The GUI representation techniques in the existing VGT approaches are incapable of unifying multimodal semantic information such as text content, screen images, metadata, et al., but require subsequent manual execution and implementation. To address the dual issues, this paper presents GUI2DSVec (GUI to Design Smell Vector), a novel GUI representation method that automatically extracts semantic information from GUI images and components to embedding vectors without manual exertion. By representing image and component metadata into vector space, this study introduces an automated end-to-end mobile application design smell detection method based on GUI representation. Performance validation on 64,759 real Android UIs demonstrates GUI2DSVec's efficiency in detecting visual design smells (accuracy@20 of 0.77). Additionally, the paper analyzes the impact of discrete representation modules and model variations on the performance of the smell detection method.

Keywords: Mobile GUI testing · GUI representation · Violation detection

1 Introduction

Graphical user interfaces (GUIs) serve as the portal through which end users access information and interact, playing an indispensable role in the mobile internet era. The process of GUI design and development is continuous and poses challenges even for senior designers and developers. Designers must adhere to numerous design guidelines [23], ensuring that the created prototype aligns with standards of consistent, beautiful, and effective [14]. For developers, the advancements in front-end technology [29] have introduced plenty dazzling visual effects to GUI design, such as dense streaming media embedding and dynamic particle animations. While these technologies improve the user experience, they

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 445–457, 2024.
https://doi.org/10.1007/978-3-031-63992-0_29

also present significant challenges for developers. Additionally, the abundance of development frameworks [15] and the existence of contentious development specifications [1] have further elevated the demand for developers' expertise.

These pressures on developers and designers render the occurrence of UI design smells in mobile application GUIs inevitable. A UI design smells[1] is any visual characteristic within the GUI, indicating issues that violate UI design principles and subsequently impact the user experience. Common design smells are shown in Fig. 1 (design smells marked in red boxes). In Fig. 1(a), the text superimposed upon the background image renders indistinct the title in the top bar. In Fig. 1(b), applying tabs and bottom navigation at the same time may bewilder users, for it remains confusing which tab controls the current content. In Fig. 1(c), employing icons to some destinations and not others in a navigation drawer. Icons should be used for all destinations, or none. In Fig. 1(d), within a simple dialog, the action button is redundant for the choice itself is actionable when tapped, and users can tap anywhere outside the dialog to close it. These design smells violate the design guidelines developed by Google Material Design documentation [2].

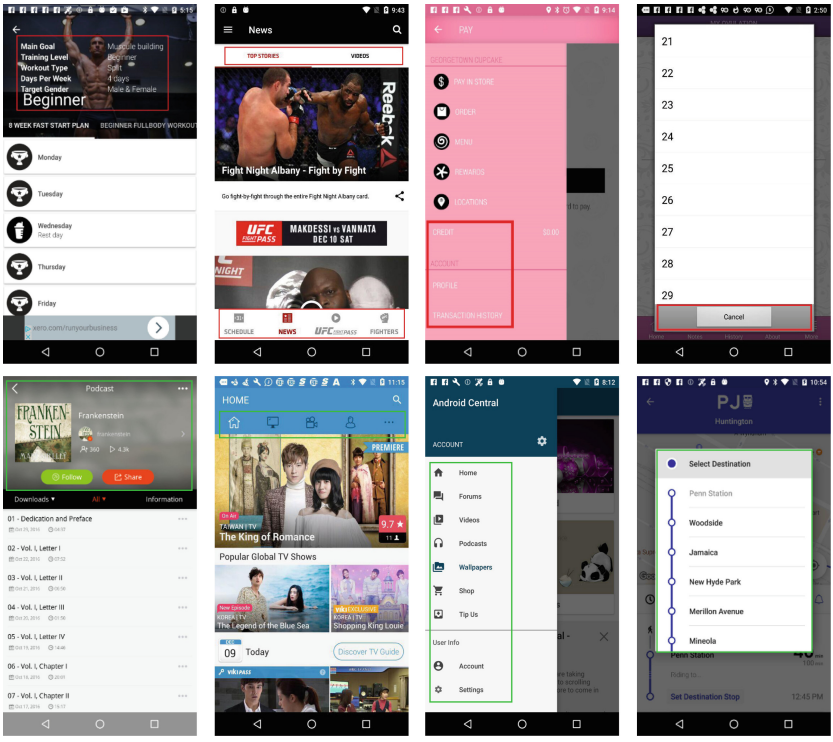
To ensure the correctness of the GUI display and fix design smells, development teams recruit testers or crowdsource GUI testing of applications. Due to the high cost and long duration of manual GUI testing, GUI automation testing technology has become a research focus in academia and industry. Alegroth et al. [4] divided GUI automation testing methods into three generations chronologically. The third-generation testing methods [20,32], namely visual GUI testing (VGT), refer to the utilization of image recognition as the core technology for analyzing and interacting with the front-end of software applications.

However, the current VGT methods are restricted in their ability to identify only a few UI design smells related to text and color [30]. The primary constraints can be summarized as follows: (1) Due to the current lack of design smell data sets, existing machine learning-based visual design smell detection models train predictive models on synthetic data sets [32], often making it difficult to ensure the effectiveness and practicality of data and models; (2) The generalization ability of the model is poor. Different models have different tasks and feature representation structures [22,32], making it difficult to map and learn in the same space; (3) Existing GUI image and component representation methods contain limited semantic information. More details can be found in Sect. 4.

To address the above problems, this paper makes the following contributions:

1. We propose a new GUI representation method GUI2DSVec, that automatically extracts atomic semantic information including text, layout, images, colors, etc. from GUI images and metadata.
2. We implement an automated end-to-end mobile application design smell detection method based on GUI representation.
3. Experiments show that GUI2DSVec achieves 77% accuracy and can automatically detect and classify visual design smells.

[1] The definition of design smell is inspired by code smell [13].



(a) Illegible buttons/text

(b) Tab and bottom navigation together

(c) Icons lost in navigation drawer

(d) No button in simple dialog

Fig. 1. UI design smells (1st row) vs non-smell UIs (2nd row) (issues highlighted in red boxes) (Color figure online)

2 A Semantic Embedding GUI Representation Model

GUI2DSVec is inspired by the word embedding method Word2Vec, that is, predicting the next entity based on the context of the target entity. In Word2Vec, words are used as input, while in GUI2DSVec, GUI component metadata and screenshots are used as input. The generated embedding vectors are used to improve downstream tasks driven by artificial intelligence. Semantic representations of GUIs are meaningful in GUI search [27] and GUI semantic summarization [18]. However, existing GUI representation methods also have limitations. One class of representation methods only focuses on text information on the screen and treats the screen as a bag-of-words model. For example, in SOVITE [16], the exact matching of text labels on the screen is used to summarize the tasks demonstrated by the user. Such representation methods ignore important GUI features such as visual information and layout patterns. Another class of representation methods [28] represents GUIs as sequences of user operations, using Seq2Seq models to map user operation sequences to corresponding

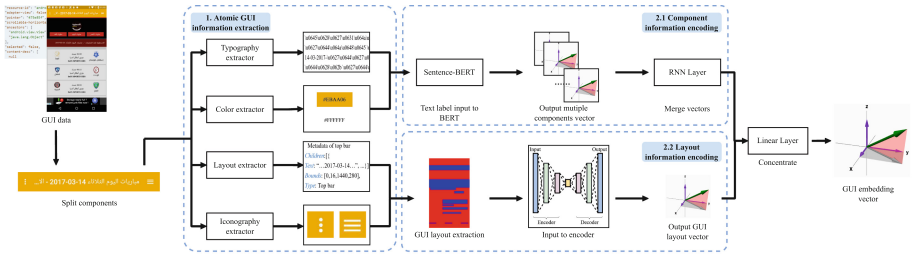


Fig. 2. Flowchart of GUI2DSVec for extracting atomic GUI information and generating GUI embedding vector

GUI components for predicting the user's possible next action or the natural language description of the interaction, etc.

In order to achieve prediction and classification of design smells, GUI2DSVec proposes using GUI hierarchical metadata and GUI images as input to extract component information, layout, and visual information for representation. The feature vectors of this information are extracted through their respective pre-trained models and finally connected as the feature vector of the GUI for subsequent classification tasks.

Figure 2 shows the workflow of GUI2DSVec. GUI2DSVec mainly includes two parts: GUI atomic information extraction and GUI feature information encoding. The goal of GUI atomic information extraction is to glean four crucial atomic GUI information from metadata and input images: text, color, layout, and images. Previous research [30] has demonstrated these four atomic GUI information can be utilized to identify design smells. GUI feature information encoding employs pre-trained models to separately encode the outputs of text and images from the atomic information extractors, separately yielding GUI component feature vectors and GUI layout feature vectors. Specifically, the GUI component feature vector utilizes a pre-trained Sentence-BERT model to calculate, and then merged by a recurrent neural network. The GUI layout feature vector adopts an encoder-decoder architecture for image encoding.

The GUI component and layout feature are then combined using a linear layer to form a 768-dimensional vector, which is GUI embedding vector. The GUI embedding vector can be used as a feature vector for the entire UI in downstream tasks. In this paper, we use visual design smell detection as a case study for demonstration. Our method combines GUI2DSVec and a KNN algorithm-based voter for design smell prediction. The method overview is shown in Fig. 3.

2.1 Atomic GUI Information Extraction

Text Information Extraction. The goal of this step is to crop the area to be processed according to the text area coordinates and screen captures in the metadata. By a scene text detector EAST [33] and text recognition techniques Tesseract [3], the extractor outputs the text information of the area to be

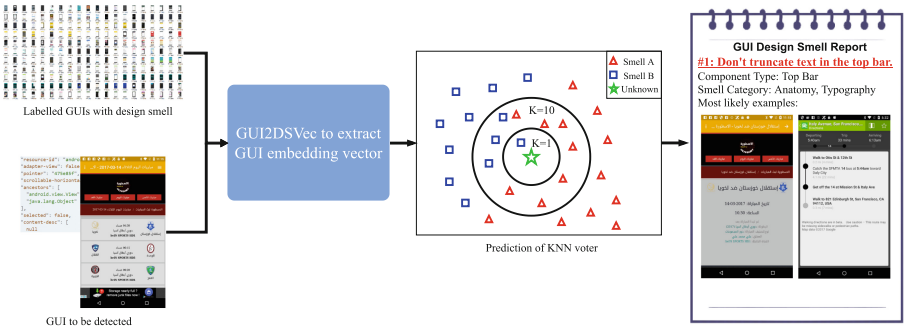


Fig. 3. Overview of design smell prediction

processed, including bounding boxes, number of lines, and strings. Although the area cropped according to the coordinates is basically accurate, it is still necessary to expand the image and perform grayscale processing in the subsequent text detection step to obtain better results. If the input data only contains screen captures, the text extractor has a built-in list for cropping common component areas from the original screen capture.

Color Information Extraction. This step enhances the initial metadata of GUI components by adding attributes to extract the dominant color of GUI components. We introduce the HSV color space for color detection, which is also utilized in recent work [8]. The technique calculates the colored area and corresponding image area percentage of each color. The color with the largest percentage is considered the dominant color of the GUI component.

Image Information Extraction. The image information extractor detects the presence of icons and images in the input UI. Then it attempts to simulate how application users perceive text, components and images in the UI. We uses the UI component detection tool [9] to detect non-text GUI subcomponent areas (such as icons and buttons) and uses EAST to detect text areas. By associating the detectable text and image areas with the component bounding boxes in the metadata, UI design smells related to readability can be determined, such as the two unreadable buttons in Fig. 1(a).

Layout Information Extraction. The layout extractor aims to glean the spatial information of components from the text metadata, such as component composition hierarchy, text content, component bounding box, and component type. Then it organizes these information into a JSON file to represent the GUI's spatial information. To avoid higher-level components in the hierarchy possessing larger bounding boxes causing overlap between components, the extractor

will also extract the edge information presented to the end user for rectification. Specifically, the Canny edge detection [7] is adopted to detect edges of components since the separator lines differ in color from the sides and remain horizontal.

2.2 GUI Components and Images Embedding Vectors

Text Label. The text label of each component is encoded into a word vector by the Sentence-BERT language model [24]. Sentence-BERT is a pre-trained model based on the BERT network, and specifically designed for generating sentence embeddings. Sentence-BERT uses a siamese architecture, which contains 2 BERT architectures that are essentially identical and share the same weights.

GUI Component Vector. In order to merge multiple GUI components into a single fixed-length feature vector, GUI2DSVec uses a recurrent neural network(RNN). The feature vectors of multiple GUI components are fed into the RNN in the preorder traversal order of the hierarchical structure. For each input vector, the hidden state input of the RNN is the output of the previous vector (the hidden state of the initial input vector is 0). This mechanism enables the final output of the RNN to encode all GUI components and can set the GUI feature vectors of different GUI images to a fixed length.

GUI Layout Vector. GUI2DSVec proposes a layout vector extractor for Rico's metadata. First, according to the hierarchical metadata, the bounding boxes of all leaf components are extracted, with different colors to distinguish text and non-text components. Then, an image autoencoder is used to encode each image into a 64-dimensional vector. The autoencoder is trained using a classic encoder-decoder architecture. The input dimension of the encoder is 11200, followed by hidden layers of size 2048 and 256, and the output dimension is 64. That is, $11200 \rightarrow 2048$, $2048 \rightarrow 256$, $256 \rightarrow 64$. These layers apply the ReLU activation function, so the output of each layer is transformed through the activation function to convert any negative input to 0. The decoder has the opposite structure ($64 \rightarrow 256$, $256 \rightarrow 2048$, $2048 \rightarrow 11200$). When the input image passes through the encoder and decoder, the layout vector extractor is trained during the reconstruction process, and the loss function is the mean square error (MSE). The network weights are optimized to produce the best 64-dimensional vector of the original input image to obtain the best reconstruction effect during decoding, that is, the minimum mean square error.

2.3 Concatenating GUI Embedding Vector

Linear layers are used to combine multiple vectors and compress the combined vector into a low-dimensional vector that contains the semantic content of each input. These weights are optimized together with other parameters to minimize the overall loss.

2.4 K-Nearest Neighbor Algorithm Predicts Violation List

The nearest neighbor algorithm [11] is used for searching the most similar patterns in the space. GUI2DSVec adopts cosine similarity to measure the distances between multiple GUI embedding vectors, and employs KNN to find the N screens most similar to a given screen from the dataset.

Specifically, for an input UI, GUI2DSVec first calculate the GUI embedding vector through the steps described above. Then GUI2DSVec find the K closest GUIs to the input GUI embedding vector in the dataset. We use the majority vote of the K nearest GUI cases to determine the type of design smell violated by the input UI. If there is a tie in the majority vote, we compare the average distances and consider the category with the shortest average distance to be the type of design smell violated by the input UI.

3 Experimental Evaluation

To train and evaluate GUI2DSVec in real-world apps, we use the open-source Rico dataset [12] to evaluate the classification effect of design smells. By combining different model variants of information and verifying their performance in classification tasks from different perspectives, this section aims to answer three research questions (RQs):

- RQ 1: What are the performance differences between GUI2DSVec and baselines?
- RQ 2: How do different models perform in smell classification from different perspectives?
- RQ 3: How is the usefulness of GUI2DSVec?

3.1 Experiment Setup

Dataset. The Rico dataset [12] contains 64,759 different GUI images from 9,384 real Android applications. To validate the effectiveness, the first author and another Ph.D. student, who is not the author of the paper, cross-annotated 19,164 UIs with smells and design smell types. Classification methods are based on Bo et al. [30]'s perspective of design smell-related knowledge. Design smells can be divided into anatomy, placement, behavior and usage from the perspective of component knowledge. From the general design knowledge perspective, they can be partitioned into layout, typography, iconography, navigation, communication, and color. Component categories include the 11 most commonly used components: banner, bottom bar, button, bottom navigation, text field, top bar, dialog, list, floating action button (FAB), tab, and navigation drawer.

Baselines. We compare GUI2DSVec to the following baseline models:

1. *Text Embedding Only:* only uses the embedding text as the screen embedding method applied in SOVITE [16]. It can only focus on the text information in the UI.

2. *Layout Embedding Only:* only uses GUI layout information as the screen embedding method applied in RICO [12]. The hierarchy files including type and location information of GUI components are obtained by the autoencoder and represent the entire UI for further prediction.
3. *Screen2Vec* [17]: uses textual content, visual image, and layout patterns to derive the final screen feature. Only the textual content of the GUI components is considered, i.e., the semantic component information contained behind the textual content is not considered.

Training Setup. The model is trained on the cross-entropy loss function using the Adam optimization algorithm. The learning rate is set to 0.001 and the batch size is 256. The model loss function is: $CrossEntropyLoss(t, p) = -\sum_{i \in S} t(i) log(p(i))$, where t denotes the true distribution and p denotes the predicted distribution. In extracting the GUI component information feature vector, the loss is the cross-entropy loss of text prediction. When calculating the cross-entropy loss, each text prediction is compared with each possible text embedding in the vocabulary. In extracting the GUI layout feature vector, the loss is the cross-entropy loss of image reconstruction. However, the layout feature vector does not contain the similarity between the correct prediction and each screen in the dataset. We adopt negative sampling to reduce the overhead of re-calculating each training iteration and avoid overfitting.

Experimental Design. Our experiments adopt the 10-fold cross validation [25] to verify our method. The average of the 10 results is output as the result of one 10-fold cross validation.

For RQ1, since the models of different classification methods are independent, we performed 10-fold cross validation of the model separately for the three classification methods. For RQ2 and RQ3, in addition to the full model that has been verified in RQ1, the remaining four model variants need to perform 10-fold cross validation on the three classification methods respectively.

Evaluation Metrics. In our experiment, we adopt *Accuracy* to measure the performance of our prediction results. Since the smell detection results of the UI to be detected are determined by the majority vote of the K nearest GUI cases, the accuracy of prediction is closely related to the number of k-nearest neighbors searched. Therefore, we use *accuracy*@K to represent the accuracy at different values of K. We use $K = 1, 5, 10, 20$ as developers are unlikely to look through a very long recommendation list.

3.2 RQ1: Performance

Table 1 reports the *accuracy*@20 of design smell prediction for the four methods. The results show that our method has the highest prediciton accuracy in general design and component categories and overall performance. The text-only baseline model has the lowest accuracy, while the layout-only baseline model has

moderate efficiency. This indicates that layout information has a greater impact on performance than text information in the design smell detection task. The Screen2Vec model gets good performance and surpasses our model in component design category, because our model introduces more embedding information and affects the prediction performance in a small number of classification categories.

Table 1. The prediction *accuracy*@20 of design smells for between our model and baselines

Model	Component Design Accuracy (4 types)	General Design Accuracy (6 types)	Component Accuracy (11 types)	Overall Accuracy
Text Only	0.48	0.54	0.59	0.581
Layout Only	0.57	0.60	0.61	0.604
Screen2Vec	**0.72**	0.74	0.74	0.730
GUI2DSVec	0.69	**0.91**	**0.75**	**0.770**

3.3 RQ2: Performance Under Different Classifications

The experiment selects three classification methods to verify the performance of GUI2DSVec. Table 1 reports the *accuracy*@20 of different models within classification methods. The Screen2Vec model (layout + text) performs best in the component knowledge category, even surpassing our model. We analyze that the reason for this is that the GUI embedding vector is too long after full connection with the text embedding vector, resulting in blurred layout embedding vector features and reduced prediction performance. For component category prediction, since similar components have similar layouts and semantic text, and this classification is refined into 11 subcategories, the performance gap of the models under this category is relatively small compared to other categories.

Table 2. Performance of detecting UI design smells for different types in real-app UIs

Typology	*Accuracy*@1	*Accuracy*@5	*Accuracy*@10	*Accuracy*@20
Component Design (4 types)	0.65	0.66	0.67	0.69
General Design (6 types)	0.90	0.90	0.91	0.91
Component (11 types)	0.66	0.69	0.71	0.75
Overall	**0.739**	**0.751**	**0.759**	**0.770**

3.4 RQ3: Usefulness

Table 2 shows the prediction accuracy of GUI2DSVec under different classification methods and different K settings. It can be found that GUI2DSVec achieved

an accuracy of 0.77 in the classification of 21 categories, indicating that for an unknown GUI, GUI2DSVec can efficiently point out problematic components that may have design smells. However, for the component knowledge category and the general design knowledge category, the general design knowledge category with more categories has higher accuracy. We believe the main reason is that the classification and data of general design knowledge are more clear. For example, if it can be pointed out that there are design smells in color, then GUI with similar color layouts are more likely to have design smells; while the too detailed anatomy and usage in the component design knowledge category cannot be clearly distinguished in similar layouts. Discerning comparison of distinct attributes and styles may be requisite for judgment.

4 Related Work

Functional GUI testing [6,26] focuses on GUI display issues that may cause application crashes. This paper focuses more on non-functional GUI testing, i.e., visual smells that will not cause application crashes but have negative impacts on application usability and user experience. The purpose of visual smell detection methods is to enable computers to understand GUIs (i.e., extract GUI features), detect design smells (i.e., identify abnormal visual features from normal GUIs and locate them), and finally provide detection reports.

Zhao et al. [32] first proposed to transform the problem of detecting design smells in GUI animations into a classification problem. They used the encoding part of the GAN as the representation model for GUI animations and used the KNN K-nearest neighbor algorithm to implement the classifier. In the absence of enough labelled data, this method uses GAN and unlabelled real GUI animations to synthesize labelled GUI animation datasets. The result is limited by the classification model, which can only classify the entire UI and cannot locate design defects in a fine-grained manner.

Liu et al. [20] proposed the Owleyes tool for text overlap, image loss, flower screen and other five design smells caused by device compatibility. This method first located design smells on GUI images, but the located areas were too large, and the classification model for these five specific design smells had poor generalization ability. Since the synthesized UIs instead of real UIs were used for training, it could lead to false positives.

The subsequent tools like Nighthawk [21] and Woodpecker [19] are derived from Owleyes. Nighthawk enhances the end-to-end ResNet model, while Woodpecker correlates the target area with the component source code. There are still problems such as poor generalization ability, false positives caused by data enhancement methods and limited smell types.

Alotaibi et al. [5] defined a graph-based component size relationship model for visual defects such as component stacking caused by scaling GUIs. This method requires static analysis of GUIs to extract hierarchical structures and generate size relationship graph models. It is less applicable than methods that only require GUI images as input and detect fewer visual defect categories.

Chen et al. [10] followed the ideas of Zhao et al. [32] to detect smells in image-rich applications (such as games). They not only enhanced the realism of synthesized data by injecting defective code snippets into the code but also extended the types of detectable smells to eight. Similarly, this method requires access to the source code of the application for code injection and has only been verified on game UIs, with generally poor applicability.

Zhang et al. [31] focused on detecting consistency between application programs, GUI policies and legal regulations. Through static analysis and normalization of expert knowledge, they detected four types of design smells that violate the Google Material Design specifications. However, its effectiveness in practice remains to be verified due to the need for application source code and less information involved in these four design smells.

Su et al. [27] proposed dVermin that detect visual defects in GUIs when magnified. This method has achieved good performance on scaling issues but cannot handle other types of design smells.

Previous studies have mainly focused on typesetting (text overlap [19–21] or size anomaly [31]), images (flower screen [10,20]), layout (distortion caused by proportional enlargement [5,27]), and animation (shadow loss after animation [32]) in specific design defects, They have not been able to solve the design defects in text hiding, blurry buttons, interactions and navigation mentioned in Fig. 1.

5 Conclusion and Future Work

This paper proposes a novel GUI representation method called GUI2DSVec to enhance data-driven VGT approaches, implementing an automated end-to-end UI design smell detection methodology. The evaluation of the real-app dataset indicates the utility of our representation modules, and the high detection accuracy of the comprehensive model. In the future, we will extend the method to automatically explore and detect smells throughout the mobile applications at the granularity of source code. Additional studies shall be conducted to validate the usefulness of our methodology and its extensions.

References

1. Html5 specification finalized, squabbling over specs continues (2014). https://tinyurl.com/3ckk6sk2
2. Google material design (2023). https://m2.material.io/components/
3. Tesseract ocr (2023). https://tesseract-ocr.github.io/
4. Alégroth, E., Gao, Z., Oliveira, R., Memon, A.: Conceptualization and evaluation of component-based testing unified with visual GUI testing: an empirical study. In: 2015 IEEE 8th International Conference on Software Testing, Verification and Validation (ICST), pp. 1–10. IEEE (2015)
5. Alotaibi, A.S., Chiou, P.T., Halfond, W.G.: Automated repair of size-based inaccessibility issues in mobile applications. In: 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE), pp. 730–742. IEEE (2021)

6. Amalfitano, D., Fasolino, A.R., Tramontana, P., Ta, B.D., Memon, A.M.: Mobiguitar: automated model-based testing of mobile apps. IEEE Softw. **32**(5), 53–59 (2014)
7. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **6**, 679–698 (1986)
8. Chen, C., Feng, S., Xing, Z., Liu, L., Zhao, S., Wang, J.: Gallery dc: design search and knowledge discovery through auto-created GUI component gallery. Proc. ACM Hum.-Comput. Interact. **3**(CSCW), 1–22 (2019)
9. Chen, J., et al.: Object detection for graphical user interface: old fashioned or deep learning or a combination? In: Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, pp. 1202–1214 (2020)
10. Chen, K., Li, Y., Chen, Y., Fan, C., Hu, Z., Yang, W.: Glib: towards automated test oracle for graphically-rich applications. In: Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, pp. 1093–1104 (2021)
11. Cover, T., Hart, P.: Nearest neighbor pattern classification. IEEE Trans. Inf. Theory **13**(1), 21–27 (1967)
12. Deka, B., et al.: Rico: a mobile app dataset for building data-driven design applications. In: Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology, pp. 845–854 (2017)
13. Fowler, M., Beck, K.: Refactoring: Improving the design of existing code. In: 11th European Conference, Jyväskylä, Finland (1997)
14. Galitz, W.O.: The Essential Guide to user Interface Design: an Introduction to GUI Design Principles and Techniques. Wiley, Hoboken (2007)
15. Kaluža, M., Troskot, K., Vukelić, B.: Comparison of front-end frameworks for web applications development. Zbornik Veleučilišta u Rijeci **6**(1), 261–282 (2018)
16. Li, T.J.J., Chen, J., Xia, H., Mitchell, T.M., Myers, B.A.: Multi-modal repairs of conversational breakdowns in task-oriented dialogs. In: Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology, pp. 1094–1107 (2020)
17. Li, T.J.J., Popowski, L., Mitchell, T., Myers, B.A.: Screen2vec: Semantic embedding of GUI screens and GUI components. In: Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems, pp. 1–15 (2021)
18. Li, T.J.J., Riva, O.: Kite: Building conversational bots from mobile apps. In: Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services, pp. 96–109 (2018)
19. Liu, Z.: Woodpecker: identifying and fixing android UI display issues. In: Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings, pp. 334–336 (2022)
20. Liu, Z., Chen, C., Wang, J., Huang, Y., Hu, J., Wang, Q.: Owl eyes: Spotting UI display issues via visual understanding. In: Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering, pp. 398–409 (2020)
21. Liu, Z., Chen, C., Wang, J., Huang, Y., Hu, J., Wang, Q.: Nighthawk: fully automated localizing UI display issues via visual understanding. IEEE Trans. Software Eng. **49**(1), 403–418 (2022)
22. Moran, K., Li, B., Bernal-Cárdenas, C., Jelf, D., Poshyvanyk, D.: Automated reporting of GUI design violations for mobile apps. In: Proceedings of the 40th International Conference on Software Engineering, pp. 165–175 (2018)

23. Nielsen, J., Molich, R.: Heuristic evaluation of user interfaces. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 1990, pp. 249–256. Association for Computing Machinery, New York, NY, USA (1990)
24. Reimers, N., Gurevych, I.: Sentence-bert: sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084 (2019)
25. Stone, M.: Cross-validatory choice and assessment of statistical predictions. J. Roy. Stat. Soc. Ser. B Methodol. **36**(2), 111–133 (1974)
26. Su, T., et al.: Guided, stochastic model-based GUI testing of android apps. In: Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering, pp. 245–256 (2017)
27. Su, Y., et al.: The metamorphosis: automatic detection of scaling issues for mobile apps. In: 37th IEEE/ACM International Conference on Automated Software Engineering, pp. 1–12 (2022)
28. White, L., Almezen, H.: Generating test cases for GUI responsibilities using complete interaction sequences. In: Proceedings 11th International Symposium on Software Reliability Engineering. ISSRE 2000, pp. 110–121. IEEE (2000)
29. Xing, Y., Huang, J., Lai, Y.: Research and analysis of the front-end frameworks and libraries in e-business development. In: Proceedings of the 2019 11th International Conference on Computer and Automation Engineering, pp. 68–72 (2019)
30. Yang, B., Xing, Z., Xia, X., Chen, C., Ye, D., Li, S.: Don't do that! hunting down visual design smells in complex UIS against design guidelines. In: 2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE), pp. 761–772. IEEE (2021)
31. Zhang, Z., Feng, Y., Ernst, M.D., Porst, S., Dillig, I.: Checking conformance of applications against GUI policies. In: Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, pp. 95–106 (2021)
32. Zhao, D., et al.: Seenomaly: vision-based linting of GUI animation effects against design-don't guidelines. In: Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering, pp. 1286–1297 (2020)
33. Zhou, X., et al.: East: an efficient and accurate scene text detector. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 5551–5560 (2017)

Interaction Technologies

Mobile User Experience from the Lens of Project-Based Learning

Maria Spichkova(✉)

RMIT University, Melbourne, Australia
maria.spichkova@rmit.edu.au

Abstract. This paper presents an overview of mobile application projects conducted at the RMIT University as a part of the Learning and Teaching activities within Bachelor and Master programs, in collaboration with industrial partners. We discuss the lessons learned over eight years of teaching the corresponding courses and compare the results of our student project to the trends summarised in the recently published approached from other universities and countries.

Keywords: usability · user experience · mobile application · education · learning · teaching · project-based learning

1 Introduction

To develop sustainable application that fulfils users' needs, it is crucial to understand the whole range of user experiences and expectations [6,24]. The developers require to understand users' motivations and what they perceive as useful and easy to use [5,27], but this is a complex task which is not easy to master. Even when a developer follows general guidelines and best practices, e.g., the human interface guidelines specified by Apple[1] or the design guidelines specified by Android[2], the needs of target user group should be analysed in addition. One of the approaches to understand needs of particular group of users, is to use so-called "personas" presenting composite user archetypes that covers consist of users' ways of thinking, behaviour, goals, and motivations for a specific context [15,29]. The latest systematic mapping study on the use of personas in Requirements engineering was introduced in [26].

Having theoretical knowledge on the importance of these aspects isn't as effective as learning from real-life scenarios. Real-world problems serve as the catalyst and central point for student engagement, which was confirmed by numerous studies from many areas of software and system development, see for example [8,11,37]. Collaborative active learning allows students to explore how the theories learned in the class apply in real-world context [17]. This has an especially strong effect when the real-life scenarios aren't simulated, i.e. when

[1] https://developer.apple.com/design/human-interface-guidelines.
[2] https://developer.android.com/design/ui/mobile/guides/foundations/accessibility.

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 461–472, 2024.
https://doi.org/10.1007/978-3-031-63992-0_30

the work is conducted with real stakeholders for the currently existing problems that require solutions as soon as possible.

In this paper, we present an overview of projects conducted at the RMIT University as a part of the Learning and Teaching activities in the School of Computing Technologies with the focus on mobile applications and mobile user experience. We introduce how it is embedded into the curriculum, as well as discuss lessons learned over eight years of teaching the corresponding courses and present a number of projects conducted by our students in collaboration with industrial partners in the area of mobile app development.

Outline: The rest of the paper is organised as follows. Section 2 presents an overview of related work. Section 3 presents a number of examples of the student projects as well as a discussion on our observations and lessons learned, comparing them with results of other studies. Finally, Sect. 4 summarises the paper.

2 Related Work

A systematic review of teaching methods in software engineering was presented in [7]. In that study, the authors summarised the most common teaching methods with the aim of a better alignment of Software Engineering (SE) courses with industry-relevant knowledge, tools, and practices.

Project-based learning offers numerous benefits to students. When the study projects are either derived from the industry or mirror genuine industrial challenges, the learning tasks naturally adopt a problem-based approach. By the same reasons, many universities nowadays transform the teaching activities to be *studio-based*, i.e., to focus on learning through hands-on activities and developing as result particular artefact, such as UI mock-ups, Web or mobile app prototypes, project management artefacts, etc. The core difference between studio-based courses and capstone courses is in the level of the study: studio-based leaning in general doesn't have any limitations on the the year of study and can be used for introductory courses as well. In contrast to this, capstone courses are the final year courses that should provide the culminating and integrative experience of a study program. Thus, capstone projects are especially appropriate to share with the students real industry experience [9,10,28,33].

Another notable benefit of project-based learning is offering students experience in team work, helping them develop skills in coordinating and sharing responsibilities within a group [21,34,38].

There are also approaches that aim to introduce students to research in computer science to retain them in academia or to prepare them to further study [16,35]. To encourage curiosity of Bachelor and Master students to the research in Software Engineering, we suggested to include research and analysis components in the projects as a bonus task. Short (one/two-week long) research projects have been sponsored by industrial partners and focused on the topics related to the project to conduct within semester. These have to be conducted after the semester end. The successful results of the proposed *Research embedded in teaching* initiative are presented in [13,14,19,20,43,44,46,48–51].

In our previous work [41,47,52], we presented how the research on eHealth, bio-engineering and autonomous systems is embedded into the study curriculum in the STEM (Science, Technology, Engineering and Mathematics) College at the RMIT University, Melbourne, Australia. We introduced a general structure of our Work Integrated Learning (WIL) modules and the core ideas on how the research and development activities is covered within the final-year courses. In our other work [45], we also discussed the core ideas of our redesign of the Software Engineering (SE) project course to meet such challenges as management of large cohorts of students, providing a real industrial experience, helping students to identify gaps in their skills and close these gaps based on the project experience, etc. In our current work, we went further to cover the research direction of mobile user experience and to compare the results of our student project with the trends summarised in the recently published approached from other universities and countries.

3 Project-Based Learning: Mobile Applications

The project-based capstone courses focus on hands-on practical experience within the related area of study, which in our case was conducted within Bachelor of Software Engineering (BSE) and Master of IT (MIT) study programs at the School of Computing Technologies. The RMIT academics provided their expertise on project management and software engineering, especially focusing on requirements engineering aspects and problem analysis, where some further domain expertise was also provided by our industrial partners.

The project teams typically have from five to six students (in some cases we also accepted a team of four or seven students). As the students already passed the courses on programming, fundamentals of software engineering, and software engineering project management, the capstone project focuses on integrating all the skills and knowledge acquired from the previous courses into a relatively large project. In that capstone course the students apply project management and software development methodologies in real-life settings, assuming that the core concepts have been already studied in the prerequisite courses on Software Engineering Project Management. The structure of the course was evolving over the last eight years, to incorporate the industry shift to Agile/Scrum development as well as to accommodate the recent needs for remote/hybrid working environment.

Generally, the project proposals shared with the students aren't limited to the projects on mobile app development: the students' preferences and interests might be very diverse, therefore the range of proposals should to cover several application areas to increase students' engagement. However, mobile app development and related projects build a large proportion of the proposed capstones.

In the rest of this section we present two examples of the projects that focused on mobile app development.

3.1 Deck Logger App

This project was conducted in Melbourne, Victoria (Australia) in collaboration with Spatial Vision. The aim of the project was to develop a prototype of a mobile app for Australian fishers. The mobile user needs identified in this project are largely different from findings presented in [25] by Kanij et al. based on a case study of fisherfolk communities in Bangladesh. These differences highlight the influence of human aspectas and national culture (see also [1–4,22,23,30,39,40]), and the impacts of low socio-economic background of users.

Some of the usability aspects critical for mobile user experience have been identified in both studies, such as simple (or even simplistic) user interface with large buttons, see Fig. 1. However, these aspects are typical for apps that should be used in environments where a user has limited time to deal with application and might have not very stable (physical) standing position, like on a boat, construction side, etc. However, in our study we haven't identified any specific requirements related to literacy, digital literacy, lack of access to smartphones, etc. For example, one of the complexities with using mobile apps identified by by Kanij et al. was that most of Bangladeshi fishermen could not read the text, which made the feature "text to voice" very important in this context. In contrast to this, Australian fishers didn't have any issues with literacy and digital literacy, have smartphones, and therefore required almost completely different set of functionalities: an application to automate reporting process.

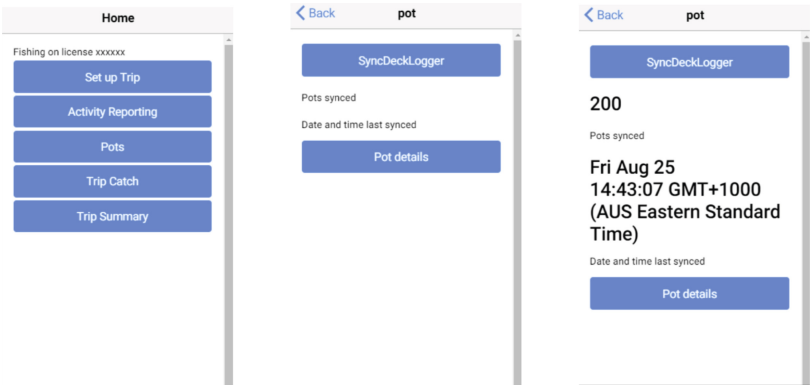


Fig. 1. Samples of the app screens for the Deck Logger App prototype: Home Screen After login, pot details page before syncing with deck logger, and pot details page after syncing with deck logger

Fisheries Victoria requires Rock Lobster fishers to report catch-and-effort data for each fishing trip which they undertake. For each trip, fishers must submit three types of reports: a pre-fishing report, a pre-landing report, and post-landing report. When these reports are manually completed by filling out

an online or paper-based logs depending on the type of report, this becomes a labour intensive and time-consuming activity for fishermen.

At the time of the project, the Victorian Rock Lobster industry applied electronic handheld devices produced by New Zealand based technology company Zebra-Tech to record catch-and-effort data. These handheld devices (so-called "Deck Loggers"[3] are easy-to-use data collection tools that require manual entry of data. However, these devices didn't offer any reporting functionality and thus manual extraction and storage of the data is required. Zebra-Tech also produces a scientific data collection device called a "Wet Tag" that can be attached various pieces of fishing equipment, e.g., a Rock Lobster Pot. These devices record location, temperature, depth and soak-time, which can be automatically transferred via Bluetooth from the Wet Tag to the Deck Logger.

The high-level objectives of this capstone project were to create a working prototype of a mobile application which:

- Enables retrieval and processing of data from existing Zebra-Tech hardware,
- Automates/streamlines the current mandatory reporting process, to decrease the time and effort fishers spend completing the reporting process,
- Enables map-based and statistical summaries of data.

different to the objectives discussed by Kanij et al. [25] in context of the case study of fisherfolk communities in Bangladesh. The reason for this was a large difference in the needs of fishers in Australia and Bangladesh, which is based on the differences in their technical and social background and industrial policies.

The app was developed using Ionic, a hybrid mobile technology that enables the application to be built for both Android and IOS using one code base. Ionic 3 was selected over other available hybrid frameworks as team members were familiar with the associated technologies and languages used within the Ionic framework. Ionic provides native user interface components for allowing for rapid prototyping and development.

3.2 Mobile Meter Reading for Non-Smart Meters

The project was conducted in collaboration with Shine Solutions Group Pty Ltd and Energy Australia, which is an electricity and gas retailing private company in Australia. It supplies electricity and natural gas to more than 2.6 million residential and business customers throughout Australia. At the time of the project, consumers used to update their utility reading manually through using an online portal. Updating the data fully manually might be inconvenient for consumers as they need to enter many details and calculate their utility reading from their meter. Also, the meters are often dusted, which makes it not so easy to read the numbers, especially for elderly people or for people with vision impairment. The cases when some digits are clipped are especially complicated.

The aim of this project was to provide a simple automated accessible alternative: to use a mobile app for capturing readings in a semi-automated way.

[3] https://www.zebra-tech.co.nz/deck-logger/.

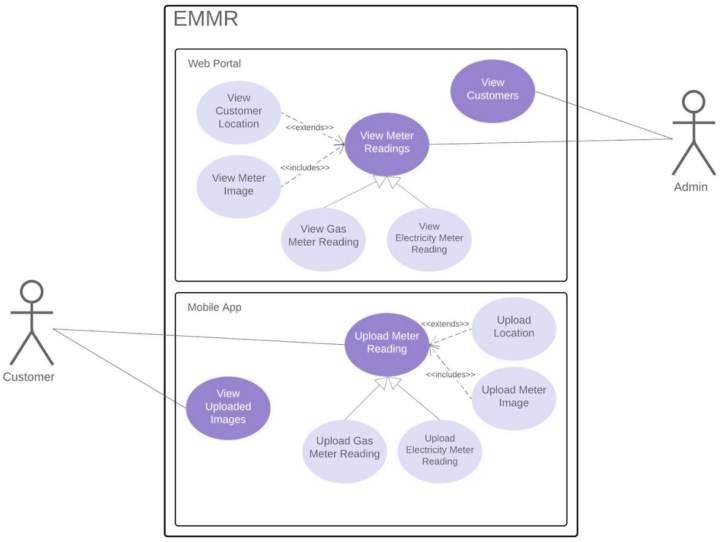


Fig. 2. Mobile Meter Reading: Use case diagram

The results of the corresponding research project, which was focused on analysis of several computer vision approaches to allow efficient automation, have been presented in [44,50]. In this paper we focus on the aspects related to the mobile user experience as well as on the corresponding learning and teaching aspects.

Figure 2 presents a use case diagram for the app. The proposed scenario was simple: a customer using the mobile application clicks an image of the meter, which will automatically start filtering out the meter readings from the image, the unrelated data will be filtered out and the best suited results are presented on the mobile app's screen. If the customer is satisfied with the image recognition results, they submit the meter reading, otherwise they might repeat an attempt. The administrator can use the web application to audit the meter readings at any time.

Figure 3 presents examples of some pages of the app. The app prototype was developed as an Android application built using React Native.

4 Discussion and Conclusions

There are many studies on project-based learning of software development. However, we believe that there are many unexplored / not discussed aspects that need to be analysed to improve learning and teaching process. In the areas like mobile user experience, where the perception of quality and the corresponding expectations are constantly evolving over recent years, the discussion of latest lessons learned is especially important.

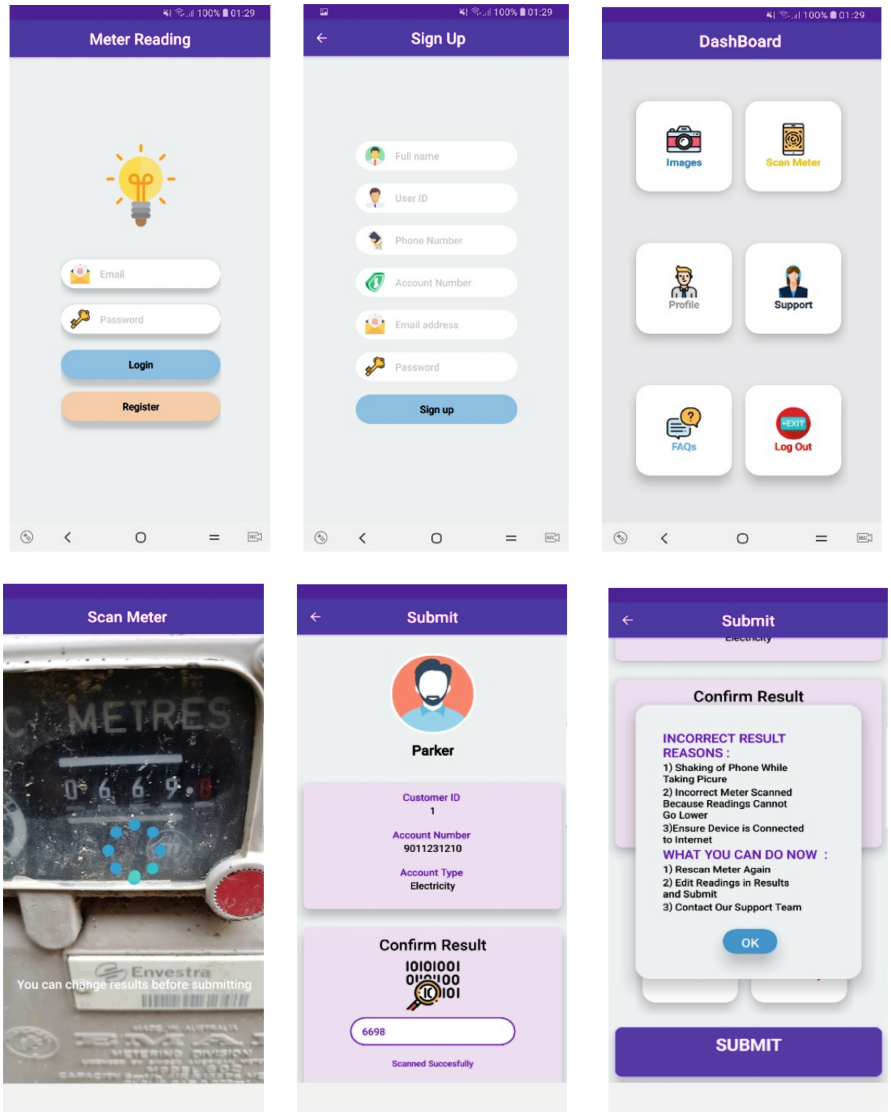


Fig. 3. Samples of the app screens: Login page, Sign Up page, User Dashboard, Capturing of the meter reading, Presenting correct results, Providing hints in the case of incorrectly identified readings

An experience report on using project-based learning in a mobile application development course was presented in [18]. The core lessons learned, identified by authors, aren't specific for mobile app development, and are rather related to general project management aspects. Similar observations have been presented in [31], where a a case study on applying user experience and user-centred design

approaches in teaching of mobile app development was introduced. We agree that these points are critical for project-based learning of software engineering (especially when the project are conducted using Agile/Scrum). Our observations also confirm that many students tend to think about software development in pure waterfall manner, even after completion of the courses on project management that have a strong focus on Agile/ Scrum. It is only once they engage in an actual project with a real industrial partner that they come to acknowledge the possibility of the system's requirements, design, and architecture really evolving throughout the course of the project.

It is generally hard for students (and potentially for for novice software engineers in general) to comprehend the fact that something might be changed over the project, as typically the assessment tasks in school and in the first year course on university level are typically well-formulated and well-structured from the beginning. Teachers aim to provide task descriptions fixed from the beginning and avoid changing them over the submission period (unless a mistake in the task is found, but this is an exceptional case). In real life this is almost never the case, as requirements should be elicited and confirmed, but then can evolve over the duration of the project. This is especially critical for the requirements related to user experience, as in many cases stakeholders might be able to provide detailed feedback only after engaging with an early prototype of the solution. Even when

- the importance of these points is highlighted in the requirements engineering literature, see e.g., [32,42], and
- the students are taught the corresponding material within the software engineering courses,

when it comes to a real project with real stakeholders and real user experience issues, many students are deeply surprised that the adjustments are needed, that the interface might need to be re-done many times, that additional functional and especially non-functional requirements are specified as top-priority in the middle or even towards the end of the project.

Thus, our observations demonstrate that the best way to master these requirements engineering and user experience aspects is to conduct a real project within a capstone course. However, it is also important to mention that completing pre-requisite courses strictly before doing the capstone is crucial, because supervisors can only mentor the students on this stage and if the students missed a large portion on background knowledge it might be to hard to prepare.

As mentioned in [12], the study projects of this type are especially effective if conducted with real industrial clients. From what we observed over the last eight years of supervision / coordination of these projects, it's also critical to provide within the course regular mentoring activities to the students on the aspects they struggle most. Thus, our observations confirmed the points discussed in [36] on enriching the students' software development project with agile coaching. Based on our lessons learned, one of the successful strategies is to enforce having an adjusted Scrum routine for meetings:

- It is important to remind students on having daily stand-ups and to update regularly their project board (Trello or Jira, depending on what the team / industrial partner decided to curate).
- If a team struggles with presentations and explanations in the Sprint review meeting with industrial partners (where the current results should be demonstrated), an additional internal (without the industrial partners) meeting with mentor might be useful, e.g. 0.5-1h before the main meeting.
- An internal Sprint retrospective meeting after the Sprint review meeting is a critical must-have part. It might be as short as 0.5h (or even shorter if the team performs really well) but having a regular catch up and regular opportunity to discuss points to improve is very effective for students to be on track and and also to share their feelings and concerns with the mentor.

In this paper, we have presented an overview of capstone projects conducted at the RMIT University as a part of the Learning and Teaching activities, with the focus on mobile user experience. We introduced two examples of the recent projects conducted in collaboration with industrial partners. We also discussed the core lessons learned from eight years of supervising and coordination the corresponding Bachelor and Master courses, as well as compared the results to the trends summarised in the recently published approached from other universities and countries.

Acknowledgement. We would like to thank Shine Solutions Group Pty Ltd and Spatial Vision for supporting the projects. We also would like to thank students who participated in the discussed projects: Callum Pearse, Baoyun Chen, Shiran Ekanayake, Lijith Win Vijay, Harish Narayanasamy, Ashish Bhardwaj, Johan van Zyl, Lavanya Krishnamurthy, Nirav Desai, and Siddharth Sachdeva.

References

1. Alsanoosy, T., Spichkova, M., Harland, J.: Cultural influence on requirements engineering activities: Australian practitioners' view. In: Siarheyeva, A., Barry, C., Lang, M., Linger, H., Schneider, C. (eds.) Information Systems Development: Information Systems Beyond 2020 (ISD2019 Proceedings) (2019)
2. Alsanoosy, T., Spichkova, M., Harland, J.: A detailed analysis of the influence of Saudi Arabia culture on the requirement engineering process. In: Damiani, E., Spanoudakis, G., Maciaszek, L.A. (eds.) ENASE 2018. CCIS, vol. 1023, pp. 240–260. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-22559-9_11
3. Alsanoosy, T., Spichkova, M., Harland, J.: Does our culture influence requirements engineering activities? Vietnamese practitioners' view. In: 2020 IEEE 28th International Requirements Engineering Conference (RE), pp. 394–395. IEEE (2020)
4. Alsanoosy, T., Spichkova, M., Harland, J.: Identification of cultural influences on requirements engineering activities. In: Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Companion Proceedings, pp. 290–291 (2020)
5. Amin, M., Rezaei, S., Abolghasemi, M.: User satisfaction with mobile websites: the impact of perceived usefulness (PU), perceived ease of use (PEOU) and trust. Nankai Bus. Rev. Int. **5**(3), 258–274 (2014)

6. Anderson, K., Burford, O., Emmerton, L.: Mobile health apps to facilitate self-care: a qualitative study of user experiences. PLoS ONE **11**(5), e0156164 (2016)
7. Aničić, K.P., Stapić, Z.: Teaching methods in software engineering: a systematic review. IEEE Softw. **39**(6), 73–79 (2022)
8. Armarego, J.: Advanced software design: a case in problem-based learning. In: Software Engineering Education and Training, pp. 44–54. CSEET, IEEE (2002)
9. Balaban, M., Sturm, A.: Software engineering lab: an essential component of a software engineering curriculum. In: Software Engineering Education and Training, pp. 21–30. ACM (2018)
10. Bastarrica, M.C., Perovich, D., Samary, M.M.: What can students get from a software engineering capstone course? In: Software Engineering Education and Training, pp. 137–145. IEEE (2017)
11. Boud, D., Feletti, G.: The challenge of problem-based learning. Routledge (2013)
12. Bruegge, B., Krusche, S., Alperowitz, L.: Software engineering project courses with industrial clients. Trans. Comput. Educ. **15**(4), 17 (2015)
13. Christianto, A., et al.: Enhancing the user experience with vertical transportation solutions. Procedia Comput. Sci. **126**, 2075–2084 (2018)
14. Clunne-Kiely, L., et al.: Modelling and implementation of humanoid robot behaviour. In: 21st International Conference on Knowledge-Based and Intelligent Information & Engineering Systems, pp. 2249–2258. Elsevier Science Publishers BV (2017)
15. Cooper, A., Reimann, R., Cronin, D., Noessel, C.: About Face: The Essentials of Interaction Design. Wiley, Hoboken (2014)
16. Cooper, S., Cunningham, S.: Teaching computer science in context. ACM Inroads **1**(1), 5–8 (2010)
17. Dharmaratne, A., Fung Fung, T., Abaei, G.: Implementing a successful collaborative active learning approach in information technology discipline. In: Chang-Tik, C., Kidman, G., Tee, M.Y. (eds.) Collaborative Active Learning: Practical Activity-Based Approaches to Learning, Assessment and Feedback, pp. 237–267. Springer, Singapore (2022). https://doi.org/10.1007/978-981-19-4383-6_10
18. Francese, R., Gravino, C., Risi, M., Scanniello, G., Tortora, G.: Using project-based-learning in a mobile application development course-an experience report. J. Visual Lang. Comput. **31**, 196–205 (2015)
19. Gaikwad, P.K., Jayakumar, C.T., Tilve, E., Bohra, N., Yu, W., Spichkova, M.: Voice-activated solutions for agile retrospective sessions. Procedia Comput. Sci. **159**, 2414–2423 (2019)
20. George, L.C., Guo, Y., Stepanov, D., Peri, V.K.R., Elvitigala, R.L., Spichkova, M.: Usage visualisation for the AWS services. Procedia Comput. Sci. **176**, 3710–3717 (2020)
21. Gol, O., Nafalski, A.: Collaborative learning in engineering education. Ph.D. thesis, Unesco, Internationa Centre for Engineering Education (2007)
22. Grundy, J., Khalajzadeh, H., McIntosh, J., Kanij, T., Mueller, I.: HumaniSE: approaches to achieve more human-centric software engineering. In: Ali, R., Kaindl, H., Maciaszek, L.A. (eds.) ENASE 2020. CCIS, vol. 1375, pp. 444–468. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-70006-5_18
23. Grundy, J., Khalajzadeh, H., McIntosh, J.: Towards human-centric model-driven software engineering. In: International Conference on Evaluation of Novel Approaches to Software Engineering 2020, pp. 299–238. Scitepress (2020)
24. Hoehle, H., Venkatesh, V.: Mobile application usability. MIS Quart. **39**(2), 435–472 (2015)

25. Kanij, T., Anwar, M., Oliver, G., Hossain, M.K.: Developing software for low socio-economic end users: lessons learned from a case study of fisherfolk communities in Bangladesh. In: 2023 IEEE/ACM 45th International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS), pp. 96–107. IEEE (2023)
26. Karolita, D., McIntosh, J., Kanij, T., Grundy, J., Obie, H.O.: Use of personas in requirements engineering: a systematic mapping study. Inf. Softw. Technol. 107264 (2023)
27. Kim, Y.H., Kim, D.J., Wachter, K.: A study of mobile user engagement (MoEN): engagement motivations, perceived value, satisfaction, and continued engagement intention. Decis. Support Syst. **56**, 361–370 (2013)
28. Knudson, D., et al.: Global software engineering experience through international capstone project exchanges. In: Global Software Engineering, pp. 54–58. ACM (2018)
29. Liu, J., et al.: A curated personas and design guidelines tool for better supporting diverse end-users. In: 46th Annual Computers, Software, and Applications Conference (COMPSAC), pp. 1606–1613. IEEE (2022)
30. Liu, X., Blincoe, K., Chhetri, M.B., Grundy, J.: Human-centric software engineering - approaches, technologies, and applications. J. Syst. Softw. **204**, 111791 (2023)
31. López, M.I.C., Cervantes, A.L.E., Martínez, G.D.L.C.: Applying user experience and user-centered design software processes in undergraduate mobile application development teaching. Int. J. Cybernet. Inform. (2023)
32. Nuseibeh, B., Easterbrook, S.: Requirements engineering: a roadmap. In: Future of Software Engineering, pp. 35–46. ACM (2000)
33. Paasivaara, M., Voda, D., Heikkila, V.T., Vanhanen, J., Lassenius, C.: How does participating in a capstone project with industrial customers affect student attitudes? In: Software Engineering Education and Training, pp. 49–57 (2018)
34. Panitz, T.: Collaborative versus cooperative learning: a comparison of the two concepts which will help us understand the underlying nature of interactive learning (1999)
35. Peckham, J., Stephenson, P., Hervé, J.Y., Hutt, R., Encarnação, M.: Increasing student retention in computer science through research programs for undergraduates. In: Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education, pp. 124–128 (2007)
36. Rodríguez, G., Soria, Á., Campo, M.: Measuring the impact of agile coaching on students' performance. IEEE Trans. Educ. **59**(3), 202–209 (2016)
37. Savery, J.: Overview of problem-based learning: definitions and distinctions. Essent. Readings Prob. Based Learn. Explor. Extend. Legacy Howard S. Barrows **9**, 5–15 (2015)
38. Schoor, C., Narciss, S., Körndle, H.: Regulation during cooperative and collaborative learning: a theory-based review of terms and concepts. Educ. Psychol. **50**(2), 97–119 (2015)
39. Shahin, M., et al.: Operationalizing human values in software engineering: a survey. IEEE Access **10**, 75269–75295 (2022)
40. Shams, R.A., Hussain, W., Oliver, G., Nurwidyantoro, A., Perera, H., Whittle, J.: Society-oriented applications development: investigating users' values from Bangladeshi agriculture mobile applications. In: Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Software Engineering in Society, pp. 53–62 (2020)
41. Simic, M., Spichkova, M., Schmidt, H., Peake, I.: Enhancing learning experience by collaborative industrial projects. In: International Conference on Engineering Education and Research (ICEER), pp. 1–8 (2016)

42. Sommerville, I., Sawyer, P.: Requirements Engineering: A Good Practice Guide. Wiley, Hoboken (1997)
43. Spichkova, M., Bartlett, J., Howard, R., Seddon, A., Zhao, X., Jiang, Y.: SMI: stack management interface. In: 23rd International Conference on Engineering of Complex Computer Systems (ICECCS), pp. 156–159 (2018)
44. Spichkova, M., van Zyl, J.: Application of computer vision technologies for automated utility meters reading. In: International Conference on Software Technologies, p. (to appear). SciTePress (2020)
45. Spichkova, M.: Industry-oriented project-based learning of software engineering. In: 24th International Conference on Engineering of Complex Computer Systems (ICECCS), pp. 51–61. IEEE (2019)
46. Spichkova, M., Li, B., Porter, L., Mason, L., Lyu, Y., Weng, Y.: VM2: automated security configuration and testing of virtual machine images. Procedia Comput. Sci. **176**, 3610–3617 (2020)
47. Spichkova, M., Simic, M.: Autonomous systems research embedded in teaching. In: De Pietro, G., Gallo, L., Howlett, R.J., Jain, L.C. (eds.) KES-IIMSS 2017. SIST, vol. 76, pp. 268–277. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-59480-4_27
48. Spichkova, M., Vaish, A., Highet, D.C., Irfan, I., Kesley, K., Kumar, P.D.: Gosecure: securing projects with go. In: ENASE, pp. 587–594 (2020)
49. Spichkova, M., van Zyl, J., Sachdev, S., Bhardwaj, A., Desai, N.: Comparison of computer vision approaches in application to the electricity and gas meter reading. In: Damiani, E., Spanoudakis, G., Maciaszek, L.A. (eds.) ENASE 2019. CCIS, vol. 1172, pp. 303–318. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-40223-5_15
50. Spichkova, M., van Zyl, J., Sachdev, S., Bhardwaj, A., Desai, N.: Easy mobile meter reading for non-smart meters: Comparison of AWS rekognition and google cloud vision approaches. In: Proceedings of the 14th International Conference on Evaluation of Novel Approaches to Software Engineering, pp. 179–188 (2019)
51. Sun, C., et al.: Software development for autonomous and social robotics systems. In: De Pietro, G., Gallo, L., Howlett, R.J., Jain, L.C., Vlacic, L. (eds.) KES-IIMSS-18 2018. SIST, vol. 98, pp. 151–160. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-92231-7_16
52. Young, J., Spichkova, M., Simic, M.: Project-based learning within ehealth, bioengineering and biomedical engineering application areas. Procedia Comput. Sci. **192**, 4952–4961 (2021)

Usability Evaluation of NOW Interactions

Rosaline Barendregt[1,2(✉)], Barbara Wasson[1,2], and Martin Heitmann[1]

[1] Centre for the Science of Learning and Technology, University of Bergen, Bergen, Norway
{rosaline.barendregt,barbara.wasson,martin.heitmann}@uib.no
[2] Department of Information Science and Media Studies, University of Bergen, Bergen, Norway

Abstract. This paper presents the results of two user studies on the suitability of NOW Interactions, a novel, push notification based mobile communication technique designed to lower the threshold for responding to information inquiries by combining a reminder, a request, and a response into one interaction, for future integration into medical diaries and health apps, through evaluating adherence, user experience, and user acceptance. Both studies comprised quantitative usability tests, interviews, and TWEETS engagement evaluations.

Main findings show a 71% adherence rate to initial notifications that increased to 98% for subsequent notifications. Participants reported on particularly valuing the direct response feature of NOW Interactions, generating a positive experience. Furthermore, feedback indicated acceptance of NOW Interactions, especially when envisioned incorporated into essential medical diary apps. Based on the findings from the present user studies, the potential of NOW Interactions for integration into medical diaries and health apps is strengthened.

Keywords: NOW Interactions · Self-report · Push notifications · Quantitative usability test · Adherence · User experience · acceptance

1 Introduction

Effective communication is at the heart of healthcare delivery, affecting diagnosis, treatment, and aftercare. As technology becomes integral to health settings, its role in self-reported data collection, like medical diaries, has grown exponentially. While traditional, paper-based methods continue to be used, digital solutions offer advantages such as real-time data capture, ease of access, and increased user engagement. It is necessary to understand that simply adding a technological layer does not inherently enhance efficacy. An app's success extends beyond its technical functionality, it is heavily influenced by how people experience its use. Even feature-rich apps can under perform if users find them burdensome or challenging. The User Interface (UI) of digital health apps bridges users and technology and has the potential to significantly shape patient experiences. A well-designed UI provides an engaging and intuitive experience, encouraging consistent engagement with digital health apps, leading to adherence to prescribed routines. Such adherence is central to treatment successful [3] with the user experience (UX) of healthcare technology playing a crucial role.

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 473–484, 2024.
https://doi.org/10.1007/978-3-031-63992-0_31

NOW Interactions [2] is a push notification based communication technique designed to simplify self-reported health data collection by merging reminder, request, and response into a single, streamlined interaction. The interactions are designed to make responding to a notification easier than dismissing it. Smooth interactions are crucial in self-reporting, as they minimise the challenges encountered in providing accurate data. Any problems when providing data can compromise assessment and validity, potentially compromising health outcomes. Individuals' ability and willingness to participate in self-reporting is directly related to the effort required, with increased effort leading to decreased task completion [6, 7]. This issue ties into the overarching challenge of *respondent burden* which refers to the difficulty, stress, and time commitment associated with the process of self-reporting. Contributing factors include contact frequency, respondent effort, and the stress while responding [4]. Given its impact on adherence and response rates, addressing respondent burden is crucial in digital health [7]. Optimising UIs and interactions is a practical approach to addressing respondent burden. By reducing the effort required when providing health data, it is possible to enhance ease of use and encourage users to engage in self-reporting. However, even with user-friendly designs, capturing a person's attention consistently remains a challenge. Mobile reminders and push notifications have emerged as powerful tools for capturing attention [13]. In healthcare, they encourage people to engage in health related activities, from adhering to medication schedules [5] to completing questionnaires or make diary entries [11]. However, people nowadays face many distractions, from real-world interruptions to a multitude personal device notifications. These frequent demands on a person's attention, coupled with the potential for cognitive overload, can cause important requests to be overlooked or prematurely dismissed. Poorly timed or designed notifications disrupt and lead to negative UX, even causing users to disable these important alerts [8]. Failure to answer notifications in a healthcare settings can have severe consequences for patient health and well-being.

In this paper, we evaluate the efficacy of NOW Interactions in capturing user's attention and facilitating low threshold data provision. To assess its fit for a self-report diary context, it is essential to gauge users' adherence to this type of interactions and their overall perception of it. We examine if NOW Interactions are perceived as beneficial aids, instead of intrusive alerts, while maintaining a balance between utility and UX. The two user studies evaluate adherence, UX, and user acceptance, employing methods including quantitative usability testing, semi-structured interviews, and engagement evaluations using the TWente Engagement with Ehealth Technologies Scale (TWEETS) [9].

The following sections provide more background on NOW Interactions, outline the studies and their methods, present the results, and discuss the findings.

2 Background

The concept of NOW Interactions was developed to make the collection of selfreported health data through health apps more accessible and less intrusive, especially recognising the unique challenges experienced by individuals with health conditions [2]. Utilising the triggering power of push notifications, NOW Interactions integrates interactable elements directly into the notification, seamlessly blending the reminder, request, and

response options into one cohesive interaction. This persuades users to share brief pieces of health data without opening an app, minimising interruptions to ongoing activities.

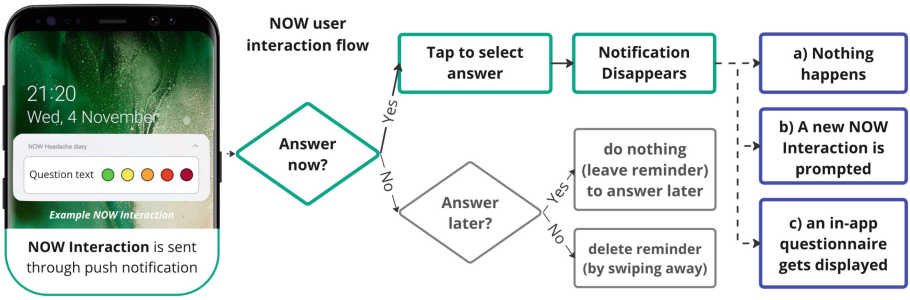


Fig. 1. Reacting to NOW Interactions from a user perspective.

From the user's perspective, a NOW Interaction appears as a push notification that promptly displays the inquiry with relevant answer options. Once an answer is selected, the notification vanishes, registering the chosen response. Subsequently, either no further action is taken, another NOW Interaction is triggered, or the user is redirected to an in-app location, see Figure 1. This design encourages users to engage with the notification rather than dismiss it.

2.1 Design Foundation

NOW Interactions is based on insights from Fogg's Behaviour Model [6], which explains human behaviour. This model identifies three core elements: motivation, ability, and triggers. To successfully engage someone into performing a task, they must have a desire to do it (motivation), the capability to do it (ability), and be reminded to do it (triggers) [6]. NOW Interactions targets the ability element by focusing on the exact moment where the user makes the decision about whether or not to act on a request. Simplifying this decision moment, by allowing data to be provided at the same time as the trigger has drawn the attention, makes the notification more meaningful as it presents a direct, concrete request instead of being just a reminder. By reducing the need to take further action or navigate through potentially complex in-app interfaces, NOW Interactions minimises user effort and simplifies the interaction process. The difference between traditional self-report and NOW Interactions is visualised in Fig. 2.

The design of NOW Interactions considers the cognitive process individuals go through when answering questions. Once their attention is captured by a trigger, respondents usually read, comprehend, retrieve from memory, judge, and then match their answer to available options [1, 15]. By streamlining these steps, NOW Interactions minimises disruptions, enhancing the likelihood of accurate and genuine reporting.

Swift interactions can enhance user engagement. Studies indicate that if a UI's response time remains below 0.1 seconds, a person's flow of thought remains uninterrupted [12]. NOW Interactions reduces the typical delay associated with opening apps, maintaining the user's focus and lowering the likelihood of task abandonment. Building

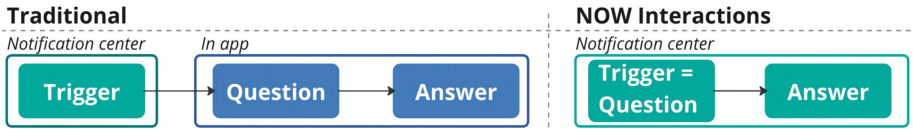


Fig. 2. Comparison of traditional and NOW Interactions responses.

on this efficiency, the design of NOW Interactions is also inspired by microinteractions [14]. These single use-case interactions, break tasks into manageable segments, preventing users from feeling overwhelmed. By presenting a single inquiry at a time, NOW Interactions ensures a smooth continuation of tasks, even if users face interruptions.

In essence, NOW Interactions facilitates a low threshold interaction, streamlining the answering process for minimal disruption and effort.

2.2 Preceding Feasibility Study

Given that existing platforms do not readily accommodate such notifications as envisioned by NOW Interactions, creating the prototype presented a unique set of challenges. To address these, a feasibility study was conducted, as detailed in [2], to identify challenges with the prototype and to gauge the potential of the overarching NOW Interactions concept.

The results of the study led to improving the NOW Interactions prototype for the user studies presented in this paper. Key takeaways on the concept of NOW Interactions included participants' appreciation of direct engagement via notifications without app opening and a positive reception to receiving consecutive questions. While the feasibility of NOW Interactions was established, the study underscored the necessity for more comprehensive research, especially concerning user adherence, acceptance, and experience over extended usage periods.

2.3 The NOW Interactions Prototype

The core functionality of the NOW Interactions prototype evaluated in the user studies presented in this paper is to send interactive notifications at preset times, recording and saving users' responses. The notifications appear in the device's notification centre, where users can respond directly from the notification without needing to open the accompanying app.

The prototype comprises three primary elements: an administrative dashboard, a backend integrated with a database, and a native Android app.

Fig. 3. Different type of icons used in NOW Interactions.

- *Administrative dashboard*: Used for customising notifications and logic, making it a versatile tool capable of adapting to diverse research scenarios.
- *Backend and database*: Provide the infrastructure to manage user interactions, store response data, and facilitate communication between the Android app and the administrative dashboard.
- *Android app*: Delivers NOW Interactions directly to users' notification centre. The app also provides a simple interface that shows a diary view of all answered questions, allowing users to track their responses over time.

The prototype's design prioritises flexibility in data collecting, allowing for custom questions, answer icons, and response evaluation logic. The prototype triggers tailored follow-up questions or actions, ensuring relevant interactions, avoiding unnecessary questioning, and optimising UX through efficient information exchange. For research purposes, the prototype also incorporates an in-app survey feature. This links the data from the NOW Interactions with survey data regarding engagement evaluations while maintaining participant anonymity.

Designing the actual notifications was challenging due to space constraints, with room only for a succinct question and up to five icon-represented answer options. Figure 3 showcases various icon sets used during the current studies.

3 User Studies

To evaluate NOW Interactions, two user studies were carried out. The first study involved a group-defined scenario focusing on post-school sleep and energy levels, while the second simulated a headache diary. Both studies employed quantitative usability evaluation, engagement evaluation, and semi-structured interviews to examine NOW Interactions on:

1. *Adherence*: Gauging consistent user engagement.
2. *User Experience*: Assessing satisfaction and ease of interactions.
3. *User Acceptance*: Evaluating users' perceived value and acceptance.

3.1 The Two Studies

The *first study* was conducted with high school students enrolled in information technology and media production. These students played an active role in shaping the study, contributing to the prototype's content, determining the scenario, adjusting the frequency and timing of notifications, and aiding in the design

of questions and icons. They wanted to focus on tracking post-school energy levels and sleep patterns. Of these students, 11 participated in the quantitative usability study.

The *second study* involved teachers from various primary and secondary schools. They were introduced to NOW Interactions with a predefined medical headache diary scenario. Six teachers participated in the quantitative usability study. For an overview of both studies, see Table 1.

When designing these two user studies, deliberate choices were made regarding incentives, participant selection, and study duration. These were rooted in our aim to

Table 1. Overview of the Studies.

	Students	Teachers
Duration	1 week	2 weeks
Scenario	Group-defined scenario focusing on post-school sleep and energy levels.	Headache diary, tracking occurrences, severity, and whether medicine was taken.
Participants & Method	38 high school students in IT and media were introduced to NOW Interactions during a workshop. 11 students qualified for the digital user test, receiving 14-22 tailored inquiries throughout the week.	During an interaction design workshop, 13 teachers were introduced to NOW Interactions. 6 qualified for the user test, receiving 41-82 inquiries over two weeks.
TWEETS + interview	8 students answered to TWEETS, 6 participated in semi-structured interview	5 teachers answered to TWEETS, 1 participated in semi-structured interview

capture genuine user adherence, cater to the unique characteristics and challenges of each participant group, and obtain meaningful results. The rationale behind these choices is further outlined in Table 2.

3.2 Inclusion and Consent

To be included in the study, participants were required to: 1) express a willingness to participate, including providing written, informed consent, 2) have access to an Android device, and 3) successfully install the prototype.

All participants provided informed consent, permitting the use of their data solely for testing and improving the NOW Interactions prototype. The research obtained approval from the university's system for risk and compliance for processing of personal data in research[1].

Table 2. Study Design Rationale.

Rationale Aspect	Details
Incentives	No incentives offered in order to measure genuine adherence without external motivators. Relevant question content was used to naturally engage participants.
Participant selection	Cohort Teachers were targeted due to their high-stress profession, aligning with the headache diary simulation. Cohort students for their tech-savviness and busy lives. Non-patients chosen to avoid potential stress on medical conditions, and bypass medical ethics committee concerns. *Note that our selection of participants was constrained due to the Covid-19 pandemic.*
Study duration	1 & 2 week durations were selected to observe potential variances in user engagement. Chose 1 & 2 week durations to align with participant comfort while ensuring significant engagement. Distinct durations offer potential to observe differences in usage decline and adherence consistency.

[1] RETTE: https://rette.app.uib.no.

3.3 Usability Testing Methods

Usability testing provides insights into user-technology interactions, helping to highlight areas of improvement. In medical contexts, addressing usability issues before advancing to clinical trials ensures that design-related issues don't overshadow medical assessments. This proactive approach allows patient feedback to focus on medical efficacy and therapeutic value, bypassing basic usability issues. In the two studies the following methods were used:

- *Quantitative usability evaluation*: Objective assessment of user behaviour by collecting specific performance metrics as users engage with the technology.
- *Engagement evaluation*: TWEETS [10], a tailored measure of engagement among users of eHealth technologies defines engagement as a combination of behaviour, cognition, and affect, using nine items. Responses are collected on a 5-point Likert scale, ranging from strong disagreement to strong agreement.
- *Semi-structured Interviews*: Interviews using a mix of open-ended and specific questions, allowing the exploration of set topics while capturing unique participant experiences and perspectives.

3.4 Data Collection and Analysis

The usability testing methods were deployed to gather data on adherence, user experience, and user acceptance.

Participants initially engaged with NOW Interactions within predefined scenarios, with data from these interactions collected in real-time. Analysis of these patterns offered insights into adherence, and engagement levels, and identified instances of dropout, offering a view of engagement over time. TWEETS items were translated to Norwegian and adapted to gauge potential for the future use of NOW Interactions. These adapted items were presented as a conventional questionnaire within the NOW Interactions prototype. The semi-structured interviews involved participants in discussing their experiences and insights on NOW Interactions. The interviews covered usability feedback, design perspectives, and challenges encountered. While certain topics were predefined to ensure consistency, the flexible nature of the interviews allowed for spontaneous insights to gauge participants' likelihood and willingness to use NOW Interactions in future scenarios. Table 3 shows the collected data.

- *Usage metrics analysis*: Inspecting data logs for patterns in interaction frequency, signs of dropouts, and engagement consistency were extracted from data logs, followed by an inter-study comparison.

- *TWEETS analysis*: Each item's mean response was calculated to ascertain participant consensus.
- *Semi-structured interviews analysis*: Transcribed audio recordings underwent thematic coding for comprehensive evaluation.

Table 3. Data Overview.

Core Aspect	Usage Metrics	Interview data	TWEETS data
Adherence	Data includes user responses, time-stamps, insights into interaction frequency, total logs, and drop-outs, etc.	Covers aspects related to user commitment, consistent interaction, and reasons for usage lapses.	Examines user perceptions of integrating such interactions into their daily routines.
User Experience	Usage metrics offer indirect insights into UX.	Discusses functionality, feedback on the notifications, user preferences (likes/dislikes), and challenges.	Focuses on ease of use and overall satisfaction.
User Acceptance	Usage metrics provide indirect insights into user acceptance.	Delves into perceived utility, potential to adopt to this way of reporting, potential integration into daily routines.	Gauges desire for future use of NOW Interactions and perceived benefits.

4 Results

From the digital user test, usage metrics were collected for eleven students and six teachers. Following this, eight students and five teachers provided feedback through the in-app TWEETS survey. Eight students and one teacher further participated in the follow-up semi-structured interview, sharing more insights into their experiences with NOW Interactions.

4.1 Quantitative Usability Test

A total of 522 NOW Interactions were sent across both studies: 330 to teachers and 192 to students. Out of these, 402 notifications (77%) received responses (Table 4 provides a detailed breakdown). It can be noted that there is significant difference between answering rates between the Initial NOW Interactions and the *secondary NOW Interactions* as shown in Table 4. While 71% of the initial notifications received responses (283 out of 400), secondary notifications captured a 98% response rate (119 out of 122). 62% of the secondary interactions was responded to within 5 seconds of the initial notification (Fig. 4c).

Figures 4a and 4b illustrate the students' and teachers' responses over a week and two weeks, respectively. In both figures, each line represents one participant. Among the teachers, we noticed a distinct case of early dropout: one teacher stopped responding on day 6, having been consistent until then (see Fig. 4b). The remaining five teachers displayed consistent engagement, averaging 49 entries each. There was only one day where one of the participants did not log any entry. Among the students, ten maintained consistent engagement until the study's last (half) day. However, one student dropped in at day 5 to respond to two NOW Notifications (Fig. 4a).

Table 4. Overview of NOW notifications sent and answered.

	Initial NOW notification			Secondary NOW notification			Total		
	sent	answered	%	sent	answered	%	sent	answered	%
Teachers (6)	246	181	74 %	84	81	96 %	330	262	78 %
Students (11)	154	102	66 %	38	38	100 %	192	140	72 %
Total	**400**	**283**	**71 %**	**122**	**119**	**98 %**	**522**	**402**	**77 %**

4.2 TWEETS Survey

Based on the TWEETS evaluation scores, both student and teacher groups found the NOW Interactions to be user-friendly, easy to use, and potentially adherable. Every item received an average score between 4 and 5, reflecting a general consensus of agreement to strong agreement. The high scores across both groups suggest that the NOW Interactions were generally well-received, with only minor variances between the two cohorts. The items that received the highest scores were 'If it was important for me to use a medical diary, this type of communication could become part of my daily routine' and 'This type of communication made it easy for me to answer the questions.', both achieving a score of 4.62. The scores on these items suggest participants' readiness to integrate NOW Interactions into their daily routines in a medical context and that they are perceived as a straightforward and effortless mode of communication.

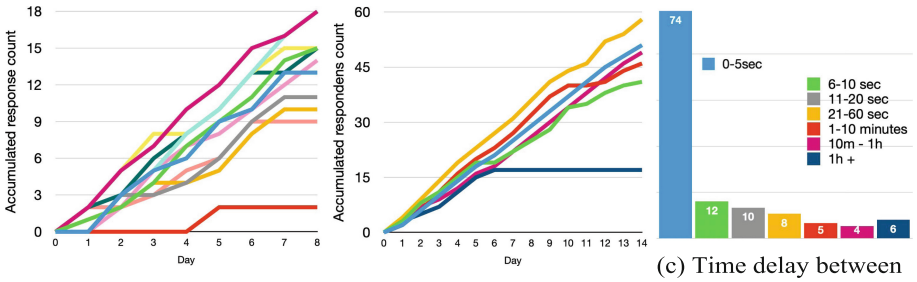
4.3 Semi-structured Interviews

The insights from the semi-structured interviews showed the participants' perceptions, preferences, and concerns regarding NOW Interactions. These insights are grouped and presented by thematic areas:

General Feedback on NOW Interactions: NOW Interactions was received positively and the consensus was it was serving as a timely reminder that encouraged regular participation. Participants praised the system for its efficiency, simplicity, and promptness. Nonetheless, there were some concerns. Among these were notifications sometimes getting lost among other alerts and a wish for more detailed response options for specific questions. The size of the icons was also pointed out by two participants as being somewhat small.

Technical Aspects: Some participants reported challenges with notifications while their phones were in power-saving mode. They hinted at irregularities in notification delivery, with notifications arriving late or seemingly not at all.

Question design: Participants indicated mostly focusing on the icons, instead of the question text. The icons were perceived as self-explanatory and helpful in quickly recognising the question's context, although a few prioritised reading the text. The energy icons and colour differentiation received notably positive feedback. Icons used in the studies are shown in Fig. 3.



(a) Student responses (b) Teacher responses (c) Time delay between initial and 2ndary NOW

Fig. 4. User response frequency and time delays for secondary NOW Interactions.

Flow of Secondary NOW Interactions: All participants reported being comfortable with responding to two consecutive NOW notifications, did not find the experience overwhelming, and felt that the flow was quite manageable. This aligns with the notably high response rates observed earlier, where 98% of the secondary interactions were answered, as seen in 4. Most felt the second notification was delivered timely, however, though some reported occasional delivery delays. Seven participants indicated that managing up to 3 to 5 consecutive interactions could be feasible.

Disruption and Intrusiveness: Participants perceived NOW as minimally disruptive and non-intrusive. Its simplicity compared to other alternatives was emphasised by both the teacher and a student as a key feature. The ability to respond without going through opening an app made the process more pleasant and efficient. Both the teacher and a student highlighted that while the notifications were noticeable, they were not intrusive. Other students reported having their phones on silent mode, thus not finding the notifications bothersome. The participants reported that even though they responded to NOW Interactions over an extended period of time, they did not give much conscious thought to the process. One student specifically mentioned that they have a tendency to discontinue participation in similar studies but chose to stay engaged in this one.

5 Discussion

Determining the potential of NOW Interactions to seamlessly integrate into daily routines, especially within medical diaries, relies heavily on users consistently and easily engaging with them.

In NOW Interactions, *adherence* can be seen from two perspectives: 1) adherence to the initial NOW Interaction, and 2) adherence to the following communication in the form of secondary NOW Interactions. During our studies, initial notifications received a response in 71% of the cases. Once a user responds to the primary notification, they typically engage with the subsequent notifications, as illustrated by a 98% follow-up adherence rate. This trend was supported by interview feedback, where eight of the nine participants indicated they felt they could manage 3-5 sequential notifications. This leads to an interesting consideration, *Could NOW Interactions be used for longer sequences or complete questionnaires*?

While the teacher cohort had a clear case of dropout, the student cohort showed one student (student 4 in Fig. 4a) only having activity in the middle of the study. This might be related to irregularities reported in notification delivery, especially when phones are in power-saving mode, and can impact consistent user engagement. The irregularities might originate from our app installation process which was facilitated through a testing mediator app, bypassing the conventional Play Store listing requirements. Consequently, this decision could result in our app's notifications being down prioritised when in power-saving mode. Addressing this issue could increase response rates to initial NOW Interactions. Both students and teachers provided insights indicating their willingness to incorporate NOW Interactions into their daily routines, especially when linked to a critical medical diary apps, underscoring *user acceptance*. Comments from the interviews also showed a clear preference towards NOW Interactions compared to other methods. Participants particularly praised the direct response feature without the necessity to open an app, indicating an enhanced *UX*. The consistent engagement of users with secondary NOW notifications reflects their acceptance of NOW Interactions as a communication technique.

The feedback was largely positive, yet it is essential to note that students actively contributed to the study's design. Their participation in shaping the scenario's theme, notification timing, and designing elements, could have fostered a sense of ownership, potentially leading to favourable views on NOW Interactions. That said, this active engagement can be seen as reflective of the intrinsic motivation typical of actual medical diary users.

For upcoming tests, we plan to expand our participant pool, previously limited by Covid-19 measures. We will also lengthen the testing phase and further explore the potential of consecutive NOW Interactions. Currently, we are evaluating feedback from health professionals on NOW Interactions' effectiveness and relevance, with many expressing significant interest in its prospective uses.

6 Conclusion

In this paper we presented two usability studies on the efficacy of NOW Interactions, a novel communication method designed to lower the threshold for responding to information inquiries by combining a reminder, a request, and a response into one interaction. Results were promising with a 98% response rate to secondary interactions, highlighting user engagement.

Participants reported on particularly valuing the direct response feature and indicated willingness to incorporate NOW Interactions into their daily routines when linked to medical diary apps. While the study did spotlight areas for improvement, notably concerns with notification delivery in power-saving mode, the overarching sentiment was positive.

The usability studies presented in this paper are the first to assess NOW Interactions and suggest it holds considerable potential for improving user adherence and experience for medical diaries. Future studies will focus on developing a more stable prototype, delving deeper into its capabilities, and exploring its applicability in broader sectors, including education.

Acknowledgements. Thanks to all study participants for their time and insights. Special thanks to the High School teacher who facilitated our engagement with the students.

References

1. Aday, L.A., Cornelius, L.J.: Designing and Conducting Health Surveys: A Comprehensive Guide. Wiley, Hoboken (2006)
2. Barendregt, R., Wasson, B.: Persuasive mobile now interactions. In: Kurosu, M. (eds.) Human-Computer Interaction. User Experience and Behavior, HCII 2022. Lecture Notes in Computer Science, vol. 13304, pp. 313–326. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-05412-9_22
3. Baumeister, H., Reichler, L., Munzinger, M., Lin, J.: The impact of guidance on internet-based mental health interventions—a systematic review. Internet Interventions **1**(4), 205–215 (2014)
4. Bradburn, N.: Respondent burden. In: Proceedings of the Survey Research Methods Section of the American Statistical Association, vol. 35, p. 40. American Statistical Association Alexandria, VA, USA (1978)
5. Fenerty, S.D., West, C., Davis, S.A., Kaplan, S.G., Feldman, S.R.: The effect of reminder systems on patients' adherence to treatment. Patient Prefer. Adherence **6**, 127–135 (2012)
6. Fogg, B.J.: A Behavior Model for Persuasive Design, pp. 1–7 (2009)
7. Graf, I.: Respondent burden. In: Lavrakas, P.J. (ed.) Encyclopedia of survey research methods, vol. 1, pp. 739–740. SAGE publications, Inc. (2008)
8. Hansson, R., Ljungstrand, P., Redström, J.: Subtle and public notification cues for mobile devices. In: Abowd, G.D., Brumitt, B., Shafer, S. (eds.) UbiComp 2001. LNCS, vol. 2201, pp. 240–246. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45427-6_20
9. Kelders, S.M., Kip, H.: Development and initial validation of a scale to measure engagement with ehealth technologies. In: Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems, pp. 1–6 (2019)
10. Kelders, S.M., Kip, H., Greeff, J.: Psychometric evaluation of the twente engagement with ehealth technologies scale (tweets): evaluation study. J. Med. Internet Res. **22**(10), e17757 (2020)
11. Koitsalu, M., Eklund, M., Adolfsson, J., Gronberg, H., Brandberg, Y.: Effects of pre-notification, invitation length, questionnaire length and reminder on participation rate: a quasi-randomised controlled trial. BMC Med. Res. Methodol. **18**(1), 1–5 (2018)
12. Nielsen, J.: Usability Engineering. Morgan Kaufmann, Burlington (1994)
13. Pielot, M., Church, K., De Oliveira, R.: An in-situ study of mobile phone notifications. In: Proceedings of the 16th International Conference on Human-Computer Interaction with Mobile Devices & Services, pp. 233–242 (2014)
14. Saffer, D.: Microinteractions: Designing with Details. O'Reilly Media Inc, Sebastopol (2013)
15. Tourangeau, R., Rips, L.J., Rasinski, K.: The Psychology of Survey Response. Cambridge University Press, Cambridge (2000)

QA Reasoning Enhancement Model Based on the Fusion of Dictionary and Hierarchical Directed Graph

Yuhang Bie[1], Meiling Liu[1(✉)], and Jiyun Zhou[2]

[1] Northeast Forestry University, Harbin, China
mlliu@nefu.edu.cn
[2] Johns Hopkins University, Baltimore, USA

Abstract. In the study of question and answer system, using pre-training and pre-training language model and knowledge graph for joint reasoning still faces two challenges: one is how to effectively solve the problem of knowledge lack in the pre-training stage; the other is how to capture more local evidence after external knowledge enhancement. To address these two pain points, This paper presents a new quiz inference enhancement model——Inference Enhancement Model (InferEM), From the perspective of knowledge enhancement and the interpretability of the reasoning process, First, the dictionary information is integrated into the pre-training through synonym replacement, translation enhancement, Improve the model's ability to predict low-frequency words; Then, we propose the hierarchical digraph method, Using the hierarchical directed graph (HD-GNN) extracted from the knowledge graph, Query related neighbors' attention selection strongly related edges capture local information, Enhance the reliability of the evidence chain. In this paper, we evaluate the newly proposed model InferEM model on two dataset benchmarks in the field of common sense quiz, which outperforms the existing single quiz inference model and the existing pre-trained language and knowledge graph joint model.

Keywords: Q&A Knowledge reasoning · Hierarchical directed graph · Knowledge Graph

1 Introduction

In the big system of instant communication on the internet, for those seeking solutions to problems, the faster the problem is solved, the better. Their needs are timely and accurate. Sometimes, solving question-and-answer type problems requires daily experience, while others require specialized knowledge. This is difficult for most people to achieve, so more and more self-help QA systems have emerged. Recently, many QA tasks have gradually become more difficult, requiring not only machine understanding of the content of the question, but also relationship reasoning of entities and their relationships through the use of external knowledge [1–4], posing great challenges to QA models. For example,

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 485–499, 2024.
https://doi.org/10.1007/978-3-031-63992-0_32

the question in Fig. 1 requires a model to reason about the entities mentioned, inferring implicit relationships between concepts. Background knowledge like "Where can I stand on a river to see water falling without getting wet?" may be difficult for machines to understand, but it is common sense for humans, and using various knowledge to help understand the meaning of pre-training is a key human ability.

Common sense QA evaluates whether machines can understand pre-training like humans by asking questions that rely on common sense knowledge behind the answers. Initially, models relied on pre-trained language models for prediction and transferred learning based on a large amount of initial knowledge, showing excellent performance for lower difficulty QA. However, as difficulty increases, the shortcomings of pre-trained language models gradually become apparent, as the vocabulary of the model is greatly challenged by the increase in difficulty, resulting in unsatisfactory QA performance. Later, the emergence of knowledge graphs solved the problem of knowledge shortage, and the appearance of a large amount of structured data elevated the accuracy of reasoning to a new level. Of course, knowledge graphs are not without their drawbacks. The construction of knowledge graphs is too large, and the process of internal retrieval can be extremely time-consuming, leading to problems with unidentifiable neighbor nodes with too few neighbors, resulting in incomplete local information affecting the final results. Recently, some researchers have proposed a solution that combines knowledge graphs and pre-trained language models to overcome the shortcomings of low accuracy of a single model's reasoning. Although the fusion of pre-trained language models and knowledge graphs has improved the accuracy of reasoning to a certain extent, the difficulties of each still remain unresolved.

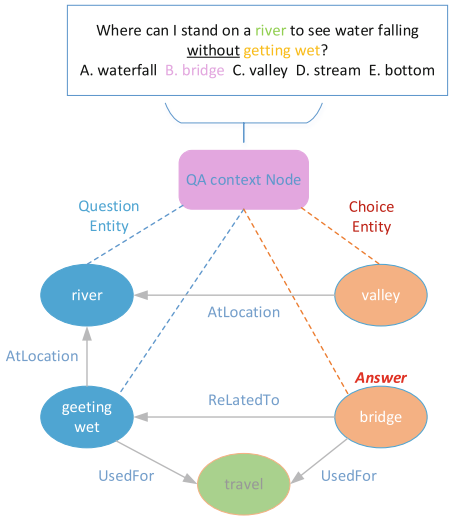


Fig. 1. Given the QA context, our goal is to derive answers through joint inference of language and knowledge maps.

In this paper, improvements were made to address the shortcomings of the two models mentioned above. Specifically, the weaknesses of GNN were addressed, and its pre-training was enhanced. The Inference Enhancement Model (Inference Enhancement Model, InferEM) was proposed, which improves on the fusion model in two ways: first, by improving the handling of low-frequency words and implicit knowledge prediction in the pre-training language model; second, by recognizing that there are limitations to capturing local evidence and that directed subgraphs are more advantageous than knowledge graphs and provide interpretability. The main contributions of this paper are as follows:

Dt-RoBERTa using the description information of the dictionary, three ways are adopted to improve the prediction ability of low-frequency words. At the same time, the self-attention mechanism is improved. One is the pool output connection, the second is the external attention, and the third is the layered external attention, which greatly improves the model efficiency and generalization ability.

We propose a hierarchical directed graph method, through the encoding and reasoning mechanism design of local information, it adopts multi-dimensional features and adopts different neighbor sampling strategies for each dimension, thus encoding different local information of nodes. The two model uses the reversible module and multi-head attention mechanism to capture and model each inference route and obtain the best inference results by choosing different inference routes. Through the choice of inference route, the model can make full use of local information, as well as global information, improving the reasoning ability.

2 Relation Work

Usually, knowledge in question answering can be implicitly encoded in large pre-trained language models on unstructured text [5, 6] or explicitly represented in structured knowledge graphs, such as Freebase [7] and ConceptNet [8], where entities are represented as nodes and their relationships are represented as edges. Recently, pre-trained language models have achieved significant success in many question answering tasks [9, 10]. However, while pre-trained language models cover a wide range of knowledge and have achieved great success, knowledge at the commonsense level is still insufficient, resulting in significant gaps between the inferred results and reality, with the most significant being the deviation in joint reasoning. In addition, pre-trained language models perform well in text-based unstructured prediction but perform poorly in structured reasoning (e.g., handling negation) [11]. On the other hand, knowledge graphs are suitable for structured reasoning [12, 13] and can make interpretable predictions, such as by providing reasoning paths [16], but may lack coverage and fail to capture implicit information effectively [15, 16]. Previous work [14, 17, 18] retrieved subgraphs from the knowledge graph by obtaining the topic entity (the knowledge graph entity mentioned in the question-answer pair) and its few-hop neighbors. However, this introduces many entity nodes that are semantically irrelevant to the question-answer pair, especially when the number of topic entities or hops increases.

Existing fusion inference methods [14, 19–21] of pre-trained language models and knowledge graphs treat them as two independent modalities. They separately apply pre-trained language models to question-answer pairs, but pre-trained language models lack

sufficient external knowledge for training and structured knowledge is not applicable to training pre-trained language models. In addition, the success in modeling graph-structured data introduces GNNs to capture subgraph structures in knowledge graphs, which solves the shortcomings of directly retrieving from the knowledge graph, while the path structure of subgraphs is conducive to the interpretability of reasoning. R-GCN [22], CompGCN [23], KE-GCN [24], and QA-GNN [25] propose updating entity representations by aggregating all neighbors of each layer through paths. However, they cannot distinguish the structural roles of different neighbors, and they are also not interpretable.

Although these methods have solved the challenges to some extent in question answering tasks, there are still two aspects to be addressed regarding the insufficient knowledge of pre-trained language models and the inability to effectively capture local information in subgraphs. Combining pre-trained language models and knowledge graphs for inference presents two challenges: (1) how to effectively solve the knowledge deficiency problem with pre-trained language models and perform joint reasoning given a question-answer pair; (2) how to strengthen the capture of local evidence behind rich external knowledge. To address these two challenges, this paper proposes the InferEM model and conducts extensive experiments on two question answering datasets, showing significant improvements over previous models.

3 Approach

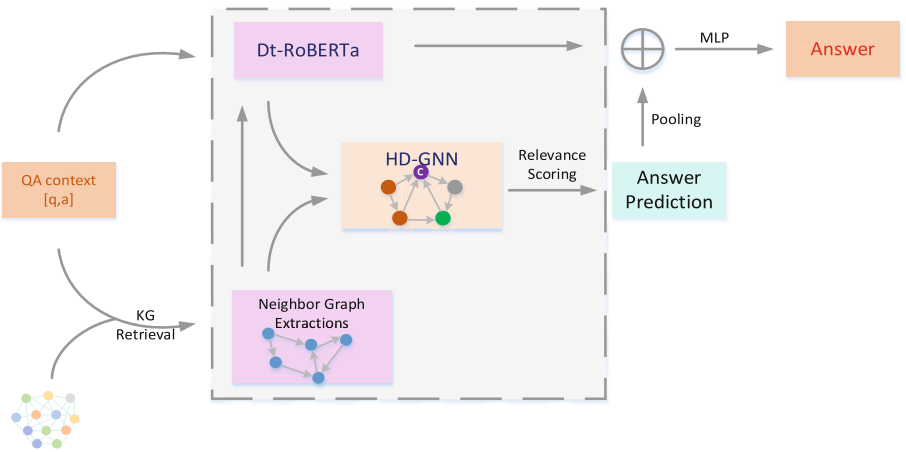


Fig. 2. Q&A inference enhancement model diagram

As shown in Fig. 2, given a question q and an answer option a, they are concatenated to form a question pair (question and answer choice) [q;a]. To perform reasoning on the given question pair using the knowledge from both the pre-trained language model and knowledge graph, InferEM works as follows. First, a node $\mathbf{C}$ representing the question pair is introduced and connected to the topic entities $\tilde{\mathrm{V}}_{\mathrm{q,a}}$, resulting in a pre-fusion graph

$\mathcal{G}_w$ with two knowledge sources (Sect. 3.1).Then, the pre-trained language model is enhanced with knowledge for the question pair using a dictionary (Sect. 3.2) to obtain its representation, and a Initial Graph $\mathcal{G}_w$ is retrieved from the knowledge graph. To adaptively capture the relationship between the question pair node and each other node, a correlation score is computed for each pair using the pre-trained language model and added as an additional feature to each node. Then, a hierarchical directed graph module based on attention is proposed, which performs layered $\mathcal{G}_w$ message passing on the pre-fusion graph (Sect. 3.3). Finally, the pre-trained language model, hierarchical directed graph, and pooled work graph representation are used for final prediction.

3.1 Initial Graph

To design a joint inference space for these two knowledge sources, we need to connect them in a common graph structure. This paper introduces a question-to-pair context node $\complement$ to represent the question-to-pair context, and uses two new relationship types γ_q^c and γ_a^c will be connected to the subgraph $\mathcal{G}_s$ on $\varphi_{q,a}$, each subject entity in $\complement$. These relationship types capture the relationship between relevant entities in the question-pair context and the knowledge graph, depending on whether the entity is found in the question or answer section of the question-to-answer context. Since this subgraph is only a preliminary construction of the connections between entities, this paper calls the pre-joint graph $\mathcal{G}_w = (\beta_w, \varphi_w)$.

$$\varphi_w = \varphi_s \cup \{\complement\} \tag{1}$$

$$\beta_w = \beta_s \cup \left(\complement, \gamma_q^c, e\right) e \in \varphi_q \cup \left(\complement, \gamma_a^c, e\right) e \in \varphi_q \tag{2}$$

Each node in $\mathcal{G}_w$ is associated with one of the four types: $\Gamma = \{\complement, Q, A, O\}$, each type represents the context node $\complement$, node in φ_q, node in φ_a and other nodes. This article represents the text of the context node and the knowledge graph node $e \in \varphi_s$ (entity name) as ($\complement$) and $text(e)$.

In this paper, we initialize the node embedding by answering a pre-trained language model representation of the context

$$\complement^{plm} = f_{enc}((\complement)) \tag{3}$$

Perfect each node on the $\mathcal{G}_s$ through its entity embedding.

3.2 Pre-trained Language Model Based on Dictionary Knowledge Enhancement

The advantage of Dt-RoBERTa lies in using the lexicon description information [36] as an external knowledge to enhance the pre-training language model, in order to improve the reasoning ability of the model and the ability to deal with problems without background knowledge. Solved the model of low frequency words in the corpus, through the low-frequency word dictionary and the corresponding low frequency word definition to enhance training pre-training language model, and introduced for the word level and sentence level of two special tasks, these information can help the model to learn the

relationship between the entity and semantics, so as to improve the performance of the model on the common sense reasoning task.

This method is mainly improved from two aspects: one is to use the dictionary knowledge to enhance the semantic information and improve the prediction ability of the model to the low frequency words; the other is to adopt a new self-attention mechanism to improve the efficiency and generalization ability of the model.

Several aspects of improving the ability to predict low-frequency words by using dictionary knowledge:

(1) **Synonym replacement**: Use synonyms in the dictionary to expand the semantic radiation range of low-frequency words. During the pre-training process, Dt-RoBERTa will randomly replace some words in the original text. If these replaced words have similar words in the dictionary, the synonym in the dictionary will be used to replace the original words, so that the model can better learn the semantic information of low-frequency vocabulary.

(2) **Translation enhancement**: A multilingual dictionary is used to translate some low-frequency words from the source language to the target language to use the broader representation of the target language. This approach can both expand the semantic radiation range of the model to low-frequency words and increase the effect on translanguage tasks.

(3) **Word sense disambiguation:** In the pre-training process, word sense disambiguation will be taken into account, that is, a common word can have multiple semantics. It will select the appropriate descriptions in multiple dictionaries to assist in the analysis of the semantic relations of words. This way can enhance the model to understand the polysemy in the low-frequency vocabulary and improve the generalization ability.

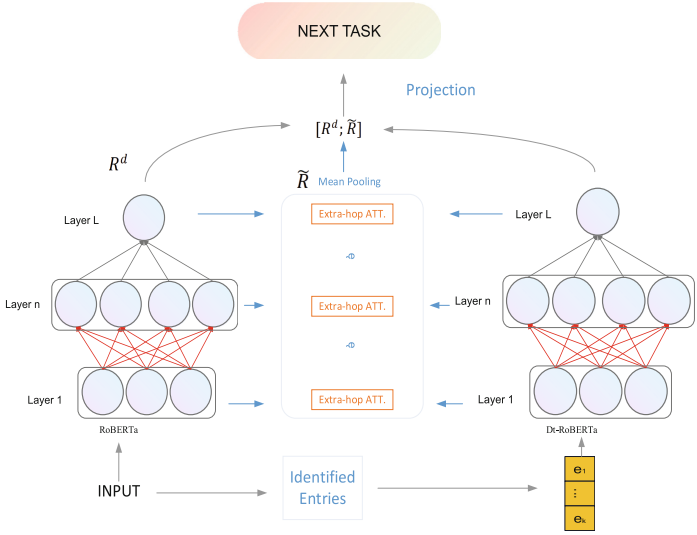


Fig. 3. Layered external hop attention

In fine-tuning, this paper uses the pre-trained-Dt-RoBERTa as a plug-in knowledge base to retrieve the implicit knowledge of identified entries in the input sequence and inject the retrieved knowledge into the input, enhancing its representation through a novel external hop attention mechanism. Specifically, the dictionary entries are first identified from a given input, then the corresponding entry information is retrieved using Dt-RoBERTa as a knowledge base, and finally the retrieved entry information is injected into the original input to obtain an enhanced representation of the downstream tasks. To better utilize the retrieved tacit knowledge in downstream tasks, three different knowledge injection mechanisms are introduced: Layered external hop attention, as shown in Fig. 3.

Layered external hop attention. To further improve the performance, the extra-hop attention of the last layer is extended to each inner layer, making it a more sensible $\tilde{R}$ layer. Attention scores were calculated for each layer, and finally with their average for implicit input knowledge injection. Specifically, the layered external jump attention can be described as follows:

$$\tilde{R}_{\mathrm{l}} = \sum_{i=1}^{\mathrm{M}} ATT\left(R_l, e_i^l\right) \tag{4}$$

$$\tilde{R} = \frac{1}{L} \sum_{l=1}^{L} \tilde{R}_{\mathrm{l}} \tag{5}$$

where, $\tilde{R}$ represents the weighted sum of the layer l output of Dt-RoBERTa. For the implicit final $\tilde{R}$ obtained by Eq. (5), a similar approach is provided for the downstream tasks using $[R^d; \tilde{R}]$.

3.3 A Hierarchical Directed Graph Based on the Graph Neural Network

In order to improve the pre-joint graph, this paper changes the GNN model and improve the directed graph. Relational paths show strong transferable and interpretable inference power on the knowledge graph [27–29]. However, because the nodes in the pathway are only connected in order, they are limited in capturing the more complex dependencies in the knowledge graph. The GNN-based methods can learn different subgraph structures. However, none of the existing methods can effectively learn subgraph structures that are both interpretable and inductive as rules. Therefore, this paper proposes a new structure to capture important local evidence, using the following two methods: local information coding and inference mechanism design.

Local information coding: In the feature representation of nodes, not only the global information between nodes, but also the local information of nodes themselves, is taken into account. Specifically, the model employs multidimensional features and different neighbor sampling strategies for each dimension, thus encoding different local information of the nodes. In this way, the feature vectors of nodes can contain more rich local information, thus improving the ability of the model to capture local information.

Reasoning mechanism design: a mechanism based on the selection of reasoning route is adopted. From the original language information to the reasoning result, each

reasoning route will be modelled. Each node from linguistic information to inference results can be regarded as a tiny inference route. The model uses the reversible module and multi-head attention mechanism to capture and model each inference route, and obtain the best inference results by choosing different inference routes. Through the choice of inference route, the model can make full use of local information, as well as global information, improving the reasoning ability.

Inspired by RED-GNN [39], in this paper, we introduce a special type of directed graph-hierarchical graph. The hierarchical graph is a directed graph with only one source node (s) and one convergence node (t). All edges are directed, with nodes between successive layers and pointing from layer m to layer $m+1$. A general approach is used to augment the triplet [23, 28] with inverse and identity relations. Then, all relational paths of length less than or equal to L between e_q and e_a can be represented as relational paths of length $e_q \rightarrow_{r^1} \cdot \rightarrow_{r^2} \cdot \cdots \rightarrow_{r^L} e_a$. In this way, they can be formed as paths in hierarchical graphs with a single source entity e_q and a converging entity e_a. This structure preserves all the relational pathways between e_q and e_a and maintains the subgraph structure. Based on this result, a hierarchical graph is introduced.

The hierarchical graph $\mathcal{G}_{\mathrm{e_q,e_a|L}}$ is a hierarchical graph with the source entity e_q and the sink entity e_a The entities in the same layer are different from each other. Any path from e_q to e_a in the hierarchy graph is a relational path L of length $e_q \rightarrow_{r^1} \cdot \rightarrow_{r^2} \cdot \cdots \rightarrow_{r^L} e_a$, where r^m connects the entities in the $m-1$ layer with the entities in the m layer. If there is no relational path connecting e_q and e_a, define $\mathcal{G}_{\mathrm{e_q,e_a|L}} = 0$.

Inspired by the inference ability of relational pathways, the goal of this paper is to perform knowledge graph inference using directed graphs. However, unlike learning structurally simple relational paths with the sequential model [27, 30], how to efficiently construct and efficiently learn from directed graphs is challenging.

In this paper, $\beta^m_{e_q,e_a|L}$ is the edge, $\varphi^m_{e_q,e_a|L} = e_o|(e_s, r, e_o) \in \beta^m_{e_q,e_a|L}$ is an entity in the r layer of the directed graph $\mathcal{G}_{\mathrm{e_q,e_a|L}} = \beta^1_{e_q,e_a|L} \otimes \cdots \otimes \beta^L_{e_q,e_a|L}$, where hierarchical connectivity. Deine the joint operator as

$$\mathcal{G}_{e_{q1},e_{a1}|L} \cup \mathcal{G}_{e_{q2},e_{a2}|L} = \beta^L_{e_{q1},e_{a1}|L} \cup \beta^L_{e_{q2},e_{a2}|L} \otimes \ldots \otimes \beta^L_{e_{q1},e_{a1}|L} \cup \beta^L_{e_{q2},e_{a2}|L} \tag{6}$$

Given an entity e, represent $\hat{\beta}^m_e$, $\overset{\smile}{\beta}{}^m_e$, and φ^m_e as sets of outer, inner, and entities, from which e is seen walking through the r jump path.

Here, we show how GNN can be improved to learn efficiently and efficiently from directed graphs. Extracting the subgraph structure and then learning the subgraph representation is a common practice for subgraph encoding in the literature, such as GraphSage [31] and GraIL [32]. Given a query triad (e_q, r_q, e_a), subgraph coding generally includes three processes:

(1) Exthe neighbors of e_q and e_a;
(2) Take the intersection to construct the subgraph;
(3) Message messaging and use graph level representation as subgraph encoding;

When working on the directed graph $\mathcal{G}_{e_q,e_a|L}$, first, the neighborhoods of e_q and e_a are obtained in the knowledge graph. Second, take the intersection of the neighborhoods of e_q and e_a to derive the directed graph $\mathcal{G}_{e_q,e_a|L}$, layer by layer. Third, if the digraph is

empty, set $\mathcal{G}_{e_q,e_a|L}$ to 0. Otherwise, the messaging is delivered layer by layer. Since e_a is a single junction entity, the final layer $h^L_{e_a}(e_q, r_q)$ is used as a subgraph representation to encode the directed graph $\mathcal{G}_{e_q,e_a|L}$. This paper names this solution as Hierarchical diagrams.

When using a different $e_a \in \varphi$, but the same query $(e_q, r_q, ?)$ To evaluate (e_q, r_q, e_a), the e_q of the adjacent edge $\widehat{\beta}^m_{e_q}$, $m = 1...L$ is shared. The following observations were made with:

The edge set visible from e_q by m step $\widehat{\beta}^m_{e_q}$ equals $\cup_{e_a \in \varphi} \beta^m_{e_q,e_a|L}$, namely $\beta^m_{e_q}$ is the union of layer r edges in the directed graph between e_q and all entities $e_a \in \varphi$. Due to the existence of sharing the same edge set, the overlap problem can affect the computational cost, and the common method to solve this problem is dynamic programming. This has been used to aggregate node representations on large-scale graph [31], or in the representation of the knowledge graph propagation problem. Inspired by the efficient low books of dynamic programming, this paper recursively constructs the directed graph between e_q and all entities e_o as

$$\mathcal{G}_{e_q,e_0|m} = U_{(e_s,r,e_o)\in\widehat{\beta}^m_{e_q}} \mathcal{G}_{e_q,e_s|m-1} \otimes \left\{(e_s, r, e_0) \in \widehat{\beta}^m_{e_q}\right\} \tag{7}$$

Once the $m-1$ of all entities $e_s \in \varphi^{m-1}_{e_q}$ in the $\mathcal{G}_{e_q,e_s|m-1}$ layer, $\mathcal{G}_{e_q,e_s|m-1}$ indicates updated to encode by combining $\mathcal{G}_{e_q,e_0|m}$ with shared edges $(e_s, r, e_o) \in \widehat{\beta}^r_{e_q}$ in layer m. Based on the above statement and formula (7), multiple directed diagrams can be effectively encoded recursively in $\widehat{\beta}^r_{e_q}$.

3.4 Accuracy and Interpretability Design of QA Reasoning

Dt-RoBERTa has used a large amount of lexicon knowledge as pre-training data that can help the model understand semantic relationships and thereby improve the model performance on common sense inference tasks. Adding hierarchical directed graphs to pre-training can provide a richer and organized knowledge representation, thus further improving model performance and interpretability.

A hierarchial directed graph is a graphical structure used to represent a conceptual hierarchy. In the common sense reasoning task, concept hierarchy is very important for reasoning because it can help the model understand the relationships and hierarchy between concepts and thus reasoning.

Given the different triples sharing the same hierarchical graph, the local evidence used for inference in this paper is different. In order to obtain knowledge of query-related from the hierarchical graph and find interpretable local evidence, this paper uses the attention mechanism [33], encoding r_q into attention weights to control the importance of different edges in $\mathcal{G}_{e_q,e_0|m}$. The messaging function is specified as

$$h^m_{e_o}(e_q, r_q) = \delta\left\{W^m \cdot \sum_{(e_s,r,e_o)\in\widehat{\mathcal{E}}^m_{e_q}} \alpha^m_{e_s,r,e_o|r_q}\left[h^{m-1}_{e_s}(e_q, r_q) + h^m_r\right]\right\} \tag{8}$$

The attention weights $\alpha^m_{e_s,r,e_o|r_q}$ are at the edge (e_q, r, e_a) are

$$\alpha^m_{e_s,r,e_q|r_q} = \sigma\left(\left(w^m_\alpha\right)^\top \mathrm{ReLU}\left(W^m_\alpha \cdot \left(h^{m-1}_{e_s}\left(e_q, r_q\right) \oplus h^m_r \oplus h^m_{r_q}\right)\right)\right) \tag{9}$$

where $w_\alpha^m \in \mathbb{R}^{d_\alpha}$, $W_\alpha^m \in \mathbb{R}^{d_\alpha \times 3d}$ and $\oplus$ are the connection operator. Use the Sigmod function σ to ensure that multiple edges can be selected in the same neighborhood.

After converging the L layer through (8), the representation $h_{e_a}^m(e_q, r_q)$ can encode the basic information for scoring (e_q, r_q, e_a). Therefore, a simple scoring function was used

$$f(e_q, r_q, e_a) = w^\top h_{e_a}^L(e_q, r_q) \tag{10}$$

where the $w^\top \in \mathbb{R}^d$. Link the multiclass log loss [34] to each triplet (e_q, r_q, e_a), i. e

$$\sum\nolimits_{(e_q, r_q, e_a) \in \mathcal{T}_{\text{tra}}} \left\{ -f\left(e_q, r_q, e_a\right) + \log\left(\sum\nolimits_{\forall e \in \varphi} e^{f(e_q, r_q, e)} \right) \right\} \tag{11}$$

The first part in formula (11) is the score of the positive triad $\left(e_q, r_q, e_a\right)$ in $\mathcal{T}_{\text{tra}}$, which is the set of training queries, the second part contains the same query $\left(e_q, r_q, ?\right)$ The fraction of all the triples of the vs. The model parameters $\theta = \left\{\{W^m\}, \{w_\alpha^m\}, \{W_\alpha^m\}, \{h_r^m\}, w\right\}$ are randomly initialized and optimized by random gradient descent [35] minimization (12). If a set of relational pathways is strongly correlated with the query triplet, they can be interpretable by identifying the attention weights in the Hierarchical diagrams.

4 The Experiment

All experiments were written using the PyTorch framework and run on an NVIDIA A100 GPU with 80GB of memory. The test values of this model are referenced using the common measure of common knowledge and answers, namely accuracy (Acc).

4.1 Data Set

This paper evaluates InferEM on three question answer datasets: CommonsenseQA [1] and OpenBookQA [4].

CommonsenseQA is a five-choice question and answer task that requires common sense reasoning and contains 12,102 questions. The test set of CommonsenseQA is not publicly available, and the model prediction evaluation can only be published every two weeks through the official leaderboard. Therefore, the main experiment on the in-house (IH) data segmentation used in [14] by Lin et al.

OpenBookQA is a four-choice question and answer task that requires reasoning using basic scientific knowledge and contains 5,957 questions. This paper uses the official data segmentation from the [4] of Mihaylov et al.

4.2 Knowledge Graph

For CommonsenseQA and OpenBookQA, this paper uses ConceptNet, a general domain knowledge graph, as a structured knowledge source $\mathcal{G}$ for this paper. It has a total of 799,273 nodes and 2,487,810 edges. The node embeddings are initialized using the entity embedding prepared by Feng et al. [20], which applies the pre-trained trained language

model to all triples in ConneceptNet, and then obtains the ensemble representation of each entity.

Given each question and answer pair, this paper retrieved subgraph $\mathcal{G}_s$ from $\mathcal{G}$ following the pre-processing step described in Feng et al. Then, the $\mathcal{G}_s$ was trimmed to maintain the first 200 nodes from the model of Sect. 3.1.

4.3 Baseline

For fine-tuned pre-trained language models, pre-trained language models without the knowledge graph are used for comparison. Testing on the CommonsenseQA dataset using RoBERTa-larga [9] and Dt-RoBERTa and on the OpenBookQA dataset using RoBERTa-larga, AristoRoBERTa [37] and Dt-RoBERTa.

For knowledge graph and pre-trained language models, compare existing knowledge graph and pre-trained language models, sharing the same knowledge graph architecture, but using different graph neural network models. Comparing the following state-of-the-art baseline models: (1) Relation Network [38], (2) RGCN [22], (3) GconAttn [19], (4) KagNet [14], (5) MHGRN [20], and (6) QA-GNN [25]. (1), (2), (3) the GNN of the perceptual relationship of KGs, and (4) and (5) use the path in the knowledge graph for reasoning. QA-GNN uses question and answer pairs to score correlations and update them jointly. QA-GNN is the existing top performance model in the framework of this knowledge graph and pre-trained language model. For a fair comparison, the paper uses the same pre-trained language model across all baselines and the models in this paper. The key difference between InferEM and these models is that (1) none of them focuses on the importance of external knowledge of the pre-trained language model, resulting in the failure to accurately predict implicit information; (2) previous models are not good at capturing local evidence, which causes less local information on relationship edges.

4.4 Main Results

Tables 1 and 2 show the results for CommonsenseQA and OpenBookQA, respectively. On both datasets, this paper observed consistent improvement over fine-tuned pre-trained and existing knowledge map and pre-trained language model models, for example, on CommonsenseQA, 5.7% over RoBERTa and 0.82% over the previous best-knowledge map and pre-trained language model QA-GNN. Performance over the QA-GNN model shows that InferEM better uses KGs to perform joint inference than existing knowledge graph and pre-trained language model methods.

CommonsenseQA Accuracy comparison of the internal segmentation. Since the official test is hidden, this paper reports here the accuracy of internal Dev (IHdev) and Test (IHtest) after the data segmentation of Lin et al. The experimental results show that InferEM* (no pre-trained language model using RoBERTa-large) is better than the other models. The InferEM was more significantly.

The test accuracy of the OpenBookQA was compared. The first row of the table uses RoBERTa-lag, AristoRoBERTa, and Dt-RoBERTa as additional input for the quiz pair. The results showed that Dt-RoBERTa performed the best. The first column of the table is a comparison of the baseline of the current graph structure model. The experimental results show that the same pre-trained language model InferEM* effect outperforms the

Table 1. Performance comparison on Commonsense QA in-house split.

Methods	IHdev-Acc.(%)	IHtest-Acc.(%)
RoBERTa-large (w/o KG)	73.07	68.69
+ RGCN	72.69	68.41
+ GconAttn	72.61	68.59
+ KagNet	73.47	69.01
+ RN	74.57	69.08
+ MHGRN	74.45	71.11
+ QA-GNN	76.54	73.41
+ InferEM*	76.87	73.74
InferEM(Ours)	**77.13**	**74.56**

Table 2. Comparing the testing accuracy of OpenBook QA with pre training

Methods	RoBERTa-large	AristoRoBERTa	Dt-RoBERTa
RGCN	62.45	74.60	78.11
GconAtten	64.75	71.80	78.34
RN	65.20	75.35	79.86
MHGRN	66.85	80.60	81.17
QA-GNN	67.80	82.77	83.29
InferEM*(Ours)	**68.43**	**83.31**	**84.17**

previous methods and outperforms the latest QA-GNN model. Moreover, InferEM was also higher than the existing models, with the best performance.

4.5 Ablation Experiments

As shown in the ablation experiment in Fig. 4, it is concluded that the Dt-RoBERTa pre-trained language model with lexicon knowledge enhancement can improve the lack of pre-training knowledge, and the performance of Dt-RoBERTa is stronger than other models in prediction.

Depth of the InferEM model. In Fig. 5, this paper shows the use of different layer L on the x-axis to test the effect on accuracy and the coverage of test triples (e_q, r_q, e_a). e_a is visible in e_q. Intuitively, when L is increased, more triples will be overlaid and the path or subgraph between e_q and e_a will contain more information but will be more difficult to learn. As shown in the figure, when L is too small, for as $L \leq 2$, the performance of InferEM is relatively poor, mainly due to the limited information encoded in the hierarchical graph in such a small range. When $L \geq 3$, InferEM achieves the best

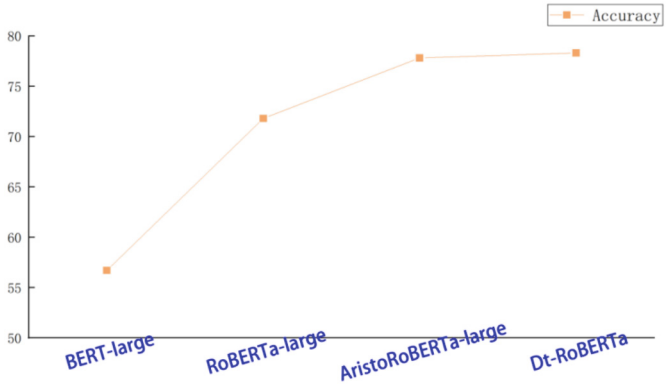


Fig. 4. Accuracy corresponding to different pre trained models

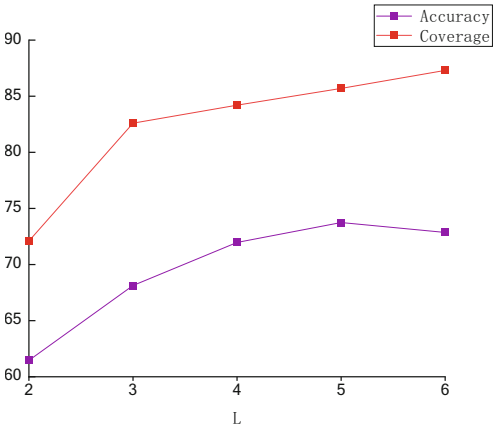


Fig. 5. The impact of hierarchy on accuracy and coverage

performance, where hierarchical graphs can contain richer information, and important information of inference can be learned efficiently.

5 Conclusion

This paper presents InferEM, an end-to-end question answering model utilizing knowledge graph and pre-trained language models. The key innovations of this paper include two points: first, to improve the vocabulary knowledge of pre-trained language model, especially the case of insufficient low frequency words, using Dt-RoBERTa to improve the predictive reasoning ability of the model; and second, to solve the problem of capturing insufficient internal evidence, the D-GNN model is used to improve GNN and interpretable. Through quantitative and qualitative analysis, this paper shows that InferEM has significant improvement over the existing pre-trained language model, knowledge map and pre-trained language model, and has the ability to explain and structural reasoning.

The following work, on the one hand, is to train the Chinese corpus, to compensate for the field and improve the robustness of the model; on the other hand, to develop more interpretable models to transform the reasoning process into a natural language form to better explain the decision of the model. In addition, the knowledge graph can also be extended and updated, because the knowledge graph is the basis of the common sense reasoning model and plays a key role.

Acknowledgment. This work was supported by National Natural Science Foundation of China 61702091 and Heilongjiang Natural Science Foundation Project LH2022F002.

References

1. TaLMor, A., Herzig, J., Lourie, N., et al.: Commonsenseqa: a question answering challenge targeting commonsense knowledge. arXiv preprint arXiv:1811.00937 (2018)
2. Sap, M., Rashkin, H., Chen, D., et al.: Socialiqa: commonsense reasoning about social interactions. arXiv preprint arXiv:1904.09728 (2019)
3. Clark, P., Cowhey, I., Etzioni, O., et al.: Think you have solved question answering? try arc, the ai2 reasoning challenge. arXiv preprint arXiv:1803.05457 (2018)
4. Mihaylov, T., Clark, P., Khot, T., et al.: Can a suit of armor conduct electricity? a new dataset for open book question answering. arXiv preprint arXiv:1809.02789 (2018)
5. Petroni, F., Rocktäschel, T., Lewis, P., et al.: Language models as knowledge bases?. arXiv preprint arXiv:1909.01066 (2019)
6. Bosselut, A., Rashkin, H., Sap, M., et al.: COMET: Commonsense transformers for automatic knowledge graph construction. arXiv preprint arXiv:1906.05317 (2019)
7. Bollacker, K., Evans, C., Paritosh, P., et al.: Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 1247–1250 (2008)
8. Speer, R., Chin, J., Havasi, C.: C. N. 5.5: an open multilingual graph of general knowledge. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, pp. 4444–4451, December 2016
9. Liu, Y., Ott, M., Goyal, N., et al.: Roberta: a robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
10. Raffel, C., Shazeer, N., Roberts, A., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res. **21**(140), 1–67 (2020)
11. Kassner, N., Schütze, H.: Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. arXiv preprint arXiv:1911.03343 (2019)
12. Ren, H., Hu, W., Leskovec, J.: Query2box: reasoning over knowledge graphs in vector space using box embeddings. arXiv preprint arXiv:2002.05969 (2020)
13. Ren, H., Leskovec, J.: Beta embeddings for multi-hop logical reasoning in knowledge graphs. Adv. Neural. Inf. Process. Syst. **33**, 19716–19726 (2020)
14. Lin, B.Y., Chen, X., Chen, J., et al.: Kagnet: knowledge-aware graph networks for commonsense reasoning. arXiv preprint arXiv:1909.02151 (2019)
15. Bordes, A., Usunier, N., Garcia-Duran, A., et al.: Translating embeddings for modeling multi-relational data. Adv. Neural Inf. Process. Syst. **26** (2013)
16. Guu, K., Miller, J., Liang, P.: Traversing knowledge graphs in vector space. arXiv preprint arXiv:1506.01094 (2015)
17. Bao, J., Duan, N., Yan, Z., et al.: Constraint-based question answering with knowledge graph. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pp. 2503–2514 (2016)

18. Sun, H., Dhingra, B., Zaheer, M., et al.: Open domain question answering using early fusion of knowledge bases and text. arXiv preprint arXiv:1809.00782 (2018)
19. Wang, X., Kapanipathi, P., Musa, R., et al.: Improving natural language inference using external knowledge in the science questions domain. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 01, pp. 7208–7215 (2019)
20. Feng, Y., Chen, X., Lin, B.Y., et al.: Scalable multi-hop relational reasoning for knowledge-aware question answering. arXiv preprint arXiv:2005.00646 (2020)
21. Lv, S., Guo, D., Xu, J., et al.: Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 05 pp. 8449–8456 (2020)
22. Schlichtkrull, M., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: Gangemi, A., et al. The Semantic Web, ESWC 2018, LNCS, vol. 10843, pp. 593–607. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93417-4_38
23. Vashishth, S., Sanyal, S., Nitin, V., et al.: Composition-based multi-relational graph convolutional networks. arXiv preprint arXiv:1911.03082 (2019)
24. Yu, D., Yang, Y., Zhang, R., et al.: Knowledge embedding based graph convolutional network. In: Proceedings of the Web Conference, vol. 2021, pp. 1619–1628 (2021)
25. Yasunaga, M., Ren, H., Bosselut, A., et al.: QA-GNN: Reasoning with language models and knowledge graphs for question answering. arXiv preprint arXiv:2104.06378 (2021)
26. Zhao, C., Xiong, C., Rosset, C., et al.: Transformer-xh: Multi-evidence reasoning with extra hop attention (2020)
27. Das, R., Dhuliawala, S., Zaheer, M., et al.: Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. arXiv preprint arXiv:1711.05851 (2017)
28. Sadeghian, A., Armandpour, M., Ding, P., et al.: Drum: end-to-end differentiable rule mining on knowledge graphs. Adv. Neural Inf. Process. Syst. **32** (2019)
29. Yang, F., Yang, Z., Cohen, W.W.: Differentiable learning of logical rules for knowledge base reasoning. Adv. Neural Inf. Process. Syst. **30** (2017)
30. Xiong, W., Hoang, T., Wang, W.Y.: Deeppath: a reinforcement learning method for knowledge graph reasoning. arXiv preprint arXiv:1707.06690 (2017)
31. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. Adv. Neural Inf. Process. Syst. **30** (2017)
32. Teru, K., Denis, E., Hamilton, W.: Inductive relation prediction by subgraph reasoning. In: International Conference on Machine Learning, pp. 9448–9457. PMLR (2020)
33. Veličković, P., Cucurull, G., Casanova, A., et al.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
34. Lacroix, T., Usunier, N., Obozinski, G.: Canonical tensor decomposition for knowledge base completion. In: International Conference on Machine Learning, pp. 2863–2872. PMLR (2018)
35. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
36. Chen, Q., Li, F.L., Xu, G., et al.: DictBERT: Dictionary description knowledge enhanced language model pre-training via contrastive learning. arXiv preprint arXiv:2208.00635 (2022)
37. Clark, P., Etzioni, O., Khot, T., et al.: From 'F' to 'A'on the NY regents science exams: an overview of the aristo project. AI Mag. **41**(4), 39–53 (2020)
38. Santoro, A., Raposo, D., Barrett, D.G., et al.: A simple neural network module for relational reasoning. Adv. Neural Inf. Process. Syst. **30** (2017)
39. Zhang, Y., Yao, Q.: Knowledge graph reasoning with relational digraph. In: Proceedings of the ACM Web Conference, vol. 2022, pp. 912–924 (2022)

AutoQuitous Workshop Session 1

Towards Distributed Control Under Deficient Communication with Multi-agent Reinforcement Learning

Arne Kummerow(✉) and Torben Weis

University of Duisburg-Essen, Bismarckstraße 90, 47057 Duisburg, Germany
{arne.kummerow,torben.weis}@uni-due.de

Abstract. Multi-agent reinforcement learning solves optimization problems in sequential decision-making and enables controlling spatially distributed actuators. We consider a cooperative setting where agents can exchange information via a central controller, e.g. a cloud-based service. In real world applications however, communication channels are often error-prone and agents may become disconnected and can neither send its observation nor receive observations from other agents. We formalize this problem as a subclass of decentralized Markov decision processes and discuss the complexity of the problem. We then propose several solution concepts that involve breaking down the complexity by considering only a subset of failure scenarios, learning independent policies for each failure scenario, reconstructing missing information and learning policies that incorporate the state uncertainty in the training process.

Keywords: MARL · Dec-MDP · Dec-POMDP · Communication Failure

1 Introduction

Many real world control problems involve operating spatially distributed actuators by use of a potentially faulty communication network, e.g. a power grid, a water drainage network or a traffic light control system [1,2]. Reinforcement learning has been used extensively in recent years to solve optimization problems in sequential decision-making [3] by learning a policy that maps the system state to control actions, but using these algorithms in a distributed scenario requires perfect communication channels. This is caused by the fact that they learn policies that do not account for being incapable of action per se, as it becomes the case when a central controller running the policy loses connection to one or more of the actuators. Modeling our task as a multi-agent system overcomes this issue by separating available state information and accessible control actions to enable

Funded by the Federal Ministry of Economic Affairs and Climate Protection of Germany (BMWK) within the project RIWWER.

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 503–510, 2024.
https://doi.org/10.1007/978-3-031-63992-0_33

generation of multiple policies that can be deployed locally where actuators are located, thus mitigating the need of a continuous communication stream.

In this paper we will focus on the cooperative multi-agent reinforcement learning (MARL) setting [5], where we want to optimize all agents against some common goal, since we are specifically interested in the problem that is posed by coordinating these agents across distant locations to solve a global objective. A lot of work has been done that incorporate the exchange of information as part of the learned policy, i.e. agents can decide what information they want to send to other agents and also to which of these. This concept is known as learning to communicate and is typically motivated by a limited communication bandwidth, latency or a restricted network topology (e.g. agents can only talk to adjacent agents). This might be the case in sensor networks or autonomous cars, for example. In the following we want to focus on scenarios where a central controller is present and no limits in communication bandwidth or latency exist. Agents may become disconnected to the central controller, however. We think that this type of information structure applies to a lot of applications, since we often encounter settings where agents have access to the internet via a cellular network and can communicate to a cloud-based service. The central controller in this case is very resilient against failure, but the communication channels to the agents that drive the actuators are not.

2 Preliminaries

A Markov decision process (MDP) is a discrete time decision framework for the single agent setting and is defined as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, R, \gamma)$, where $\mathcal{S}$ is the set of all states, $\mathcal{A}$ is the set of all action that the agent can take. Being in state $s \in \mathcal{S}$, the agent selects an action $a \in \mathcal{A}$ and transitions to next state s' with probability $\mathcal{T}(s'|s,a)$. $R\colon \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the reward function that acts as feedback for the agent, providing the optimization quantity for a given transition (s,a,s'). $\gamma \in [0,1)$ is the discount factor. The goal of the agent is to maximize the expected discounted reward $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1})]$ over encountered trajectories by learning a policy $\pi\colon \mathcal{S} \to \mathcal{A}$ that maps from states to actions. Depending on the environment, the optimal policy might however be a stochastic one, i.e. π is more generally defined as $\pi\colon \mathcal{S} \to \Delta(\mathcal{A})$, and the agent may select its action by $a \sim \pi(.|s)$.

Stochastic games are an extension to MDPs that capture the interaction of multiple agents with the environment in that they extend MDPs with a set of n agents denoted by $\mathcal{N}$. Every agent $i \in \mathcal{N}$ has its own action space $\mathcal{A}_i$ and gets reward R_i. We will from now on denote the joint action space as $\mathcal{A} = \times_{i \in \mathcal{N}} \mathcal{A}_i$. In the cooperative setting considered in this work, $R_i = R_j$ holds for all $i, j \in \mathcal{N}$.

To model the uncertainty of an agent's limited view of the environment, we define the set of joint observations as $\Omega = \times_{i \in \mathcal{N}} \Omega_i$, as well as an observation function $O\colon \mathcal{A} \times \mathcal{S} \to \Delta(\Omega)$ that indicates the probability $O(o|a,s')$ of the agents observing a joint observation o after taking joint action a and transition to state s'. Each agent can only see its corresponding part of the joint

observation. This leads us to the well known framework of decentralized partially observable Markov decision processes (Dec-POMDPs) [6], which capture the aforementioned structures as a tuple $(\mathcal{N}, \mathcal{S}, \mathcal{A}, \mathcal{T}, R, \Omega, O, \gamma)$. Regarding the observation function, different information structures have been identified [7] that describe the amount of information that an agent gets from its observation. If there exists a mapping from each observation that uniquely identifies the full state, then we can reduce the problem to a single agent decision problem by learning a joint policy that maps to the joint action space. During execution, each agent will select its action according to that policy. Another important class of observation functions are those where the composition of all observations can uniquely identify the system state. This class is referred to as jointly fully observable and the corresponding framework is denoted as a decentralized Markov decision process (Dec-MDP), which are a subclass of Dec-POMDPs. For $n > 3$ they are NEXP-complete in the worst case however, as it is the case for Dec-POMDPs for $n > 2$ [8]. A slightly stronger assumption than the joint full observability is a decomposable state $\mathcal{S} = (\mathcal{S}_1, ..., \mathcal{S}_n)$ whereat each agent observes its local state $\mathcal{S}_i$ [10,11]. Even though an agent may not observe the full state information during execution time, we will assume that the full state, observations and actions of all agents are known during training, known as the centralized-training-decentralized-execution scheme [5].

Note that during execution, an observation itself does not provide enough information to decide on an action without losing optimality [6], since the Markov property is no longer valid for partial observations. Therefore, policies have to be build on action-observation histories, which in practice is often realized by use of a recurrent architecture [5,9].

3 Problem Formulation

To address the issue of deficient communication channels, we formalize the problem as a subclass of Dec-MDPs, represented by a tuple $(\mathcal{N}, \mathcal{S}_E, \mathcal{M}, \mathcal{A}, \mathcal{T}_E, \mathcal{T}_M, R, \Omega, \mathcal{O}, \gamma)$. $\mathcal{N}$ is the set of n agents, $\mathcal{S} = \mathcal{S}_E \times \mathcal{M}$ is the set of states with $\mathcal{S}_E = \times_{i \in \mathcal{N}} \mathcal{S}_i$ being the set of environment states consisting of local states for each agent $i \in \mathcal{N}$. $\mathcal{M} = \{0, 1\}^n$ is the set of mask states that specify for each agent if it is connected to the central controller, with *1* representing that the agent is connected and can send its local state $\mathcal{S}_i$ and receive states of other agents that are connected and *0* representing that the agent is disconnected and can only observe its local state. We denote $0 = (0, \ldots, 0)$ for the case that every agent is disconnected and $1 = (1, \ldots, 1)$ if all agents are connected. Moreover, we define the relation $\trianglelefteq$ on $\mathcal{M}$ with $(m_1, \ldots, m_i, \ldots, m_n) \trianglelefteq (m'_1, \ldots, m'_i, \ldots, m'_n)$ if $m_i = 1 \implies m'_i = 1$ for all $i \in \mathcal{N}$ meaning that a mask state m is less or equally available than m' if all agents that are connected in m are also connected in m' (Fig. 1).

$\mathcal{A} = \times_{i \in \mathcal{N}} \mathcal{A}_i$ is the set of joint actions and $R \colon \mathcal{S}_E \times \mathcal{A} \times \mathcal{S}_E \to \mathbb{R}$ is the reward function for a common goal for all agents. $\mathcal{T} \colon \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition probability function, which can be factorized into two components $\mathcal{T}_E \colon \mathcal{S}_E \times \mathcal{A} \to \Delta(\mathcal{S}_E)$ and $\mathcal{T}_M \colon \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{M})$ so that

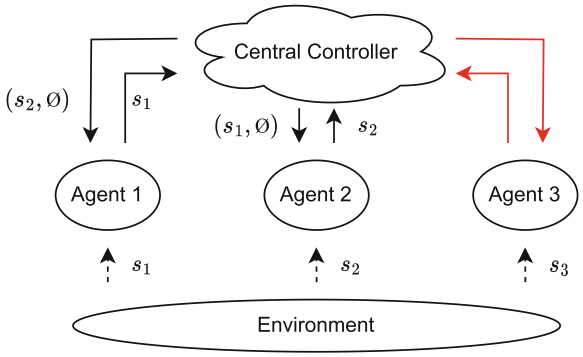


Fig. 1. Example of the proposed information structure. Three agents can exchange local states via a central controller. In the displayed situation, agent 1 and 2 are connected and see the states of each other, agent 3 is disconnected and observes local information only. Mask state is $(1, 1, 0)$.

$$\mathcal{T}((s'_E, m')|(s_E, m), a) = \mathcal{T}_E(s'_E|s_E, a)\mathcal{T}_M(m'|(s_E, m), a). \tag{1}$$

In other words, we assume that the environment state is not affected by the availability mask, but the mask state may be affected by the environment state. The process that determines whether an agent is connected to the central controller therefore does not change the actual state of system, but the communication channels may be influenced by the environment state.

$\Omega = \times_{i\in\mathcal{N}}\Omega_i$ is the set of joint observations. Each observation consists of the local state component, as well as the local state components of all other agents that are connected to the central controller, i.e.

$$\Omega_i = \mathcal{S}_i \cup (\mathcal{S}_i \times (\times_{j\in\mathcal{N}\setminus\{i\}}(\mathcal{S}_j \cup \{\emptyset\}))). \tag{2}$$

$\emptyset$ acts as a placeholder for the local state of an agent that is disconnected from the central controller. Note that we need the left union in Eq. 2 to distinguish between the case of an agent being disconnected and the case in which all other but that one agent is disconnected. Other than in the general case of a Dec-MDP, we assume the observation function to be deterministic and only dependent on the state. We thus write $\mathcal{O}\colon \mathcal{S} \to \Omega$ and define $\mathcal{O}(s) = (\mathcal{O}_1(s), \ldots, \mathcal{O}_n(s))$ with $\mathcal{O}_i\colon \mathcal{S} \to \Omega_i$ being defined as

$$\mathcal{O}_i(((s_1, \ldots, s_i, \ldots, s_n), (m_1, \ldots, m_i, \ldots, m_n))) = \begin{cases} s_i & \text{if } m_i = 0 \\ (s_i, (s_1 \otimes m_1, \ldots, s_{i-1} \otimes m_{i-1}, s_{i+1} \otimes m_{i+1}, \ldots, s_n \otimes m_n)) & \text{else.} \end{cases} \tag{3}$$

$\otimes\colon M \times \{0, 1\} \to M \cup \{\emptyset\}$ is the masking operator defined as $m \otimes 0 = \emptyset$ and $m \otimes 1 = m$ for any set M and $m \in M$.

3.1 Generalizations

In the previous section we assumed the probability function for the availability mask $\mathcal{T}_M$ to be dependent on the system state which is jointly fully observable. It might however be the case that the transition probabilities for the mask is not solely dependent on the system state, but also on some additional information that is not available to the agents. This is the case when probabilities change e.g. due to network congestion which is independent of the dynamics we want to control. Additionally, the definition of the mask transition as in Eq. 1 might violate the Markov assumption, since given the history of masks the probabilities might be different than given only the current mask (e.g. when an agent stays unavailable for at least two time steps). Alternatively, we can define the state as $\mathcal{S} = \mathcal{S}_E \times \mathcal{S}_M$, with $\mathcal{S}_M$ representing the communication state that is not observable by the agents. In that case the problem becomes a subclass of Dec-POMDPs. The same is true if the central controller has some additional information, since agents can't observe such in case of the mask state being 0 or when the observation function does not just reflect the local state, but is truly partially observable and has the general stochastic form known from Dec-POMDPs.

3.2 On the Complexity

When augmenting the state vector with additional $|\mathcal{M}| = 2^n$ failure scenarios, it can easily be seen that this drastically increases the complexity of the problem compared to the standard single agent decision problem that acts on complete information but also compared to the general case of a Dec-MDP in which agents only observe their local states. The size of the state space is now exponential in the number of agents, which is also true for their observation spaces. This implies that excessive amounts of collected data is needed during training because the number of samples required to get a good estimate of the expected reward also grows exponentially. This becomes even worse when the probability of disconnection is low, since the agent will only experience a failure state very rarely. Sparse exploration leads to a high variance in the reward signal and therefore to poor decisions. The action space is however not affected, due to the fact that the agents can always take the same actions, regardless of their connection status.

4 Solution Concepts

In this section we will discuss approaches to overcome issues with the proposed information structure, namely exponential state and observation space sizes, high variance for unlikely failure scenarios and varying input sizes for policies.

4.1 Subset of Failure Scenarios

In order to tackle the problem of exponential state and observation space sizes, a simple solution is to only account for a subset of failure scenarios $G_{\mathcal{M}} \subset \mathcal{M}$.

In case of a failure m that is not in that set, we choose some $m' \in G_{\mathcal{M}}$ for which $m' \trianglelefteq m$ holds and ignore some available information. Even though this leads to a suboptimal solution, we can reduce the number of training data needed, since we assumed the environment transition probabilities $\mathcal{T}_E$ not be affected by the communication state, i.e. we can control $\mathcal{S}_E$ regardless of $\mathcal{M}$. If more than one element of $G_{\mathcal{M}}$ is possible for m', we want to choose the element that is "closest" to m. For $m'_1 \trianglelefteq m'_2$ we choose m'_2, since we can make potentially better decisions the more information we include in the process. For $m'_1 \not\trianglelefteq m'_2$ however, we might want to select the mask that has more agents available, or randomly select one if the number of connected agents is equal. The minimal set $G_{\mathcal{M}}$ is $\{0\}$, since we can reduce every failure to that. When failures are very unlikely, it might for example make sense to train on $\{0, 1\}$, so we can act optimal most of the time and have some emergency strategy that applies to all the cases in which at least one agent has no connection to the central controller. When the probability for different agents to be unavailable is known a priori and does not depend on the environment state, optimization regarding $G_{\mathcal{M}}$ are possible, in that we include failure scenarios that are encountered most often. Otherwise, we can adapt $G_{\mathcal{M}}$ during training based on the frequency of experienced mask states. This might however prevent convergence of policies and further theoretical investigation is needed.

4.2 Independent Policies

When learning on an environment that exhibits low probabilities for certain failure scenarios or even for all other settings than 1, decision under such mask states might be poor due to the lack of experience. Instead, we can explicitly train these scenarios by restricting access to state information despite the actual connection state. While this reduces variance, it will lead to a pessimistic policy that overvalues communication failures. Reinforcement learning algorithms estimate the expected reward by drawing samples of state trajectories by interacting with the environment. Unfortunately, this process becomes biased when occurrences of mask states are altered. Alternatively, we can train multiple independent policies for each of the considered mask states and apply the action of the corresponding policy during execution. This however presumes a low rate in change of the mask state since the policies assume that the information structure they were trained on does not change at all. Frequent changes in the connection state will therefore degrade the overall performance. Nonetheless, it allows us to sample as much training data for a given mask state as needed to converge without affecting other policies. This idea has been implemented in [2], though restricted to the case of $G_{\mathcal{M}} = \{0, 1\}$, and showed being a robust backup for communication failures.

Note that an agent that is disconnected does not know the actual mask state and therefore can't decide which policy to apply. It hence must use the same policy whenever it is disconnected, so every agent needs to have only one policy for that case which will lead to a bias for these policies anyway, though it is assumed to be much smaller than using the same policy for all mask states

given that the probability for mask states during training does not reflect the occurrences of mask states during execution. Disconnected agents may also use some fixed rule based policy if they are disconnected. Another advantage of this approach is that policies have fixed input sizes which simplifies the use of artificial neural networks, which are commonly used in reinforcement learning to realize the policy [3].

4.3 Reconstruction

Another approach is to learn policies that act on complete information only. After convergence, fixing the policies enables the reduction of the underlying decision process to a Markov chain, since action probabilities do not change and can be seen as a part of transition function. This allows us to learn one or more prediction models by feeding back the reconstruction loss of the whole state given the partial state and the history of previous state information, e.g. by use of a recurrent neural network. During execution, we impute missing values by use of the prediction model and feed the complete state to the learned policy. Combinations with learning only on a subset of mask states are also possible in that we can fall back to one of the closest elements $m' \in G_{\mathcal{M}}$ with $m \trianglelefteq m'$ by imputing missing parts of m', given actual mask state m. Fallback in both directions are also possible, e.g. we can ignore information if the encountered mask state has only little more information than some mask state we trained on and impute values if only little information is missing.

A disadvantage of this approach is the fact that the longer a connection is broken, the more the performance is likely to degrade based on the assumption that the divergence between the predicted state and the actual state becomes greater the longer a communication failure persists. Policies cannot distinguish between missing information that has been imputed and a state component that was actually received by the central controller, since the prediction model has been trained after convergence of the policy. This state aliasing then may lead bad decisions. To overcome this issue, we can instead include the uncertainty induced by missing observations in the training process. Agents may for example maintain a belief over other agents' states, e.g. they learn the probability distribution of the state of other agents given the available information. This can then be used as an input to the policy, similar to work that has been done in [1].

5 Conclusion

In this paper we gave a formal definition of a communication structure that is posed by a central controller exchanging information with agents using error-prone communication channels in a spatially distributed control problem. We identified the problem to be a subclass of Dec-MDPs or Dec-POMDPs, dependent on whether the central controller has additional information that agents cannot observe locally and whether communication failures depend on some non-observable process. We discussed the complexity of the problem and argued

that the problem is even harder to solve than a multi-agent system with fixed observation spaces due to the fact that observation space sizes in the proposed formalism are exponential in the number of agents. In addition to that we identified further problems that may arise when using reinforcement learning algorithms to solve the problem, namely the high variance in the reward signal when the probabilities for communication failures are low and the varying input sizes for policies. Several solution concepts have been proposed that involve breaking down the complexity by considering only a subset of failure scenarios, learning independent policies for each failure scenario, reconstructing missing information and learning policies that incorporate the state uncertainty in the training process. Nevertheless, investigation of other information structures than those incorporating a central controller is needed. There is a lack of theoretical foundation for multi-agent systems that include the possibility of communication failures. We hope that this paper will motivate further research in this direction.

References

1. Zhou, H., et al.: Multiagent Bayesian deep reinforcement learning for microgrid energy management under communication failures. IEEE Internet Things J. **9**(14), 11685–11698 (2021)
2. Zhang, Z., Tian, W., Liao, Z.: Towards coordinated and robust real-time control: a decentralized approach for combined sewer overflow and urban flooding reduction based on multi-agent reinforcement learning. Water Res. **229**, 119498 (2023)
3. Li, Y.: Deep reinforcement learning: an overview. arXiv preprint arXiv:1701.07274 (2017)
4. Zhang, K., Yang, Z., Başar, T.: Multi-agent reinforcement learning: a selective overview of theories and algorithms. In: Vamvoudakis, K.G., Wan, Y., Lewis, F.L., Cansever, D. (eds.) Handbook of Reinforcement Learning and Control. SSDC, vol. 325, pp. 321–384. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-60990-0_12
5. Oroojlooy, A., Hajinezhad, D.: A review of cooperative multi-agent deep reinforcement learning. Appl. Intell. **53**(11), 13677–13722 (2023)
6. Oliehoek, F.A., Amato, C.: A Concise Introduction to Decentralized POMDPs, vol. 1. Springer International Publishing, Cham, Switzerland (2016). https://doi.org/10.1007/978-3-319-28929-8
7. Goldman, C.V., Zilberstein, S.: Decentralized control of cooperative systems: categorization and complexity analysis. J. Artif. Intell. Res. **22**, 143–174 (2004)
8. Bernstein, D.S., Givan, R., Immerman, N., Zilberstein, S.: The complexity of decentralized control of Markov decision processes. Math. Oper. Res. **27**(4), 819–840 (2002)
9. Zhong, Y., et al.: Heterogeneous-agent reinforcement learning. arXiv preprint arXiv:2304.09870 (2023)
10. Becker, R., Zilberstein, S., Lesser, V., Goldman, C.V.: Transition-independent decentralized Markov decision processes. In: Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 41–48 (2003)
11. Nair, R., Varakantham, P., Tambe, M., Yokoo, M.: Networked distributed POMDPs: a synthesis of distributed constraint optimization and POMDPs. In: AAAI, vol. 5, pp. 138–139 (2005)

Deep Learning Based Radio Frequency Fingerprint Identification by Exploiting Spatial Stereoscopic Features

Shunliang Zhang[1,2], Jing Li[1,2], and Xiaolei Guo[1,2(✉)]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{zhangshunliang,lijing,guoxiaolei}@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

Abstract. The unique radio frequency fingerprint (RFF) resulting from hardware imperfections can be used to identify wireless devices to resist impersonating or spoofing attacks. The existing methods for RFF identification typically rely on the transient or steady-state features of RF signals. However, the limited number of extracted features affects the performance of radio device identification, which is not satisfactory. This paper proposes a spatial stereoscopic feature extraction method that transforms the three statistical features based on radio signal frequency envelopes into three-dimensional images to address this issue. By increasing the dimensionality, it can improve the quantity of features. Additionally, we propose a multi-channel parallel convolutional neural network (MPCNN) for classifying devices based on the spatial stereoscopic features. Experimental results demonstrate that our proposed method outperforms benchmark approaches in terms of identification accuracy. Specifically, it achieves mobile device identification with an accuracy higher than 99% even with a small sample size.

Keywords: Deep learning · Device identification · RFF · MPCNN

1 Introduction

The widespread deployment of Internet of Things (IoT) devices greatly enhances people's daily lives. However, the broadcast nature of wireless communications introduces security and privacy concerns. Most existing wireless communication systems achieve security through upper-layer mechanisms, which may not be suitable for emerging wireless communication scenarios like IoT due to the computational and power limitations of the devices [1]. As a Physical layer security (PLS) technique, physical layer authentication (PLA) enables a receiver to verify the identity of a transmitter using physical layer attributes of received signals, such as channel and radio frequency (RF) characteristics. The unique fingerprint derived from the features of electromagnetic waves emitted by the transmitter can be used to identify wireless devices. Due to inherent hardware imperfections

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 511–519, 2024.
https://doi.org/10.1007/978-3-031-63992-0_34

caused by manufacturing and environmental factors, device-dependent deviations from ideal hardware can result in unique features of transmitted electromagnetic waves. Tampering with the RF fingerprint (RFF) is very difficult, making it suitable for device identification to defend against attacks.

The RFF-based PLA is applicable for device classification and identification in the IoT environment since it doesn't rely on complex cryptography operations or third-party authentication. Machine learning (ML) methods have been extensively used for RFF identification (RFFI), but their performance depends on manual feature engineering and is limited in different radio environments. Traditional ML methods also struggle with a large number of targets [2], making them unsuitable for the 6G system with massive IoT/IoV devices. Deep learning (DL) methods can extract more distinguishing RF fingerprints and show promise in improving RFF-based identification [3]. Combining transient-state and steady-state signal features in RFFI is a potential approach since both types have their strengths and weaknesses. However, conventional statistical learning or ML methods struggle to automatically select influential features by removing irrelevant or redundant features from different RF signals.

Motivated by this situation, this paper proposes a DL-based method to identify mobile devices based on spatial stereoscopic images constructed using statistical features of three-dimensional (3D) RF signals from both the transient state and steady state. The construction of spatial stereoscopic feature images improves the quantity of sample features and reduces the reliance on sample size. The main contributions of this paper are as follows:

- We developed a novel method for the extraction and representation of RF fingerprint features. Specifically, 3D features are extracted from RF signals in both the transient state and the steady state, and these features are constructed as spatial stereoscopic images.
- A multi-channel parallel CNN algorithm is developed to train models and classify RF fingerprints based on their spatial characteristics.
- We verified the performance of the proposed method through extensive simulations, and compared it with benchmark methods to highlight its advantages in terms of identification accuracy and sample size.

2 Related Work

IoT devices with limited computing and power make it difficult for traditional ML algorithms and upper-layer mechanisms to meet the security and reliability requirements of future B5G/6G systems. DL-based RFF recognition ensures the security of upcoming 6G systems [4]. Comparative analysis shows that DL-based methods outperform traditional ML methods in identification accuracy [5].

2.1 RFFI Based on Constellation Maps

A constellation map is an image representation of the statistical characteristics of modulation signals, which has been utilized by many studies as RFF fingerprint features for RFF identification. Peng et al. [6] employed the differential

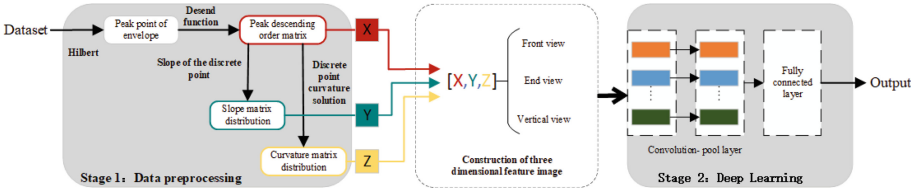


Fig. 1. Experimental framework

constellation map as the recognition feature for CNN, successfully classifying 99 Zigbee devices and evaluating RF fingerprint recognition performance under different signal-to-noise ratio (SNR) conditions. Yang et al. [7] utilized the differential constellation trajectory diagram algorithm to extract RF fingerprints. While the RFFI method based on constellation maps achieves high recognition accuracy to some extent, it is highly dependent on the SNR. When the SNR falls below a certain threshold, statistical features based on constellations degrade, consequently impacting the recognition performance.

2.2 RFFI Based on Spectral Graph

Spectral maps exhibit fine-grained time-frequency characteristics and have shown good classification performance in the RFFI scheme based on spectral graphs. Shen [8] successfully classified and recognized 20 LoRa devices using spectrum diagrams and CNNs. The experiment revealed that the RF fingerprint based on spectrum diagrams has a certain level of uniqueness and outperforms schemes based on I/Q data and FFT [9]. By converting the one-dimensional time-domain format of wireless signals into 2D spectral images, the characteristics of the spectral map are derived as RF fingerprints. VGG-16 is employed to identify RF fingerprints, resulting in the recognition of 20 different wireless transmitters [10]. With the advancement of DL, researchers have proposed new RF fingerprint feature extraction methods tailored to specific scenarios. For instance, [11] developed an SCCNN-based RFFI for civil aircraft ADS-B equipment.

3 RFF Identification Framework

3.1 The Proposed Framework

Figure 1 outlines the proposed DL-based framework for RFFI. The framework consists of two stages: RF signal preprocessing and RFFI. The 3D feature images generated from the first state are used as input for device identification in the second stage.

Phase 1: Data preprocessing and 3D image construction. The spatial stereoscopic features are constructed based on the complete envelope of radio frequency signals, including transient and steady-state features. Firstly, the envelope distribution points of the RF signal are extracted. Subsequently, the descending

arrangement of envelope distribution, slope feature distribution, and curvature feature distribution are obtained. Finally, the spatial image is constructed by using the obtained 3D features.

Phase 2: Space stereoscopic feature-based device identification. The 3D feature images obtained in stage 1 are used as input into the multi-channel parallel convolutional neural network model (MPCNN) for device identification and classification. Specifically, the three views of the 3D images are used as input to the three channels respectively for device identification.

3.2 Data Preprocessing

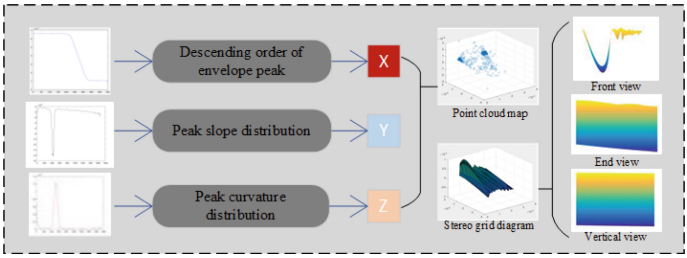


Fig. 2. Spatial stereo feature image construction

The data used in this experiment is from public datasets [12]. As shown in Fig. 2, data preprocessing consists of four steps as follows.

Step 1: The peak point of signal envelope extraction. The typical methods to extract the peak point of the signal envelope include the Hilbert, square energy, and Shannon energy. In this experiment, Hilbert's principle is used, and the specific solution is based on Formula 1. The input signal is denoted by $x(t)$. The Formulas 2 and 3 represent the peak points and the descending arrangement respectively.

$$\hat{x(t)} = H[x(t)] = X(t) * \frac{1}{\pi t} \tag{1}$$

$$P_n = peaks\left\{x(t) + j\hat{x(t)}\right\}, (n = 1....N) \tag{2}$$

$$D_n = Descend\{P_n\} = [x_n, y_n] \tag{3}$$

Step 2: Slope feature extraction. Since the envelope obtained in Step 1 is distributed as discrete points, the slope feature of the envelope distribution can be derived using the slope of these points, as shown in Formula 4.

$$\frac{y_n - y_{n-1}}{x_n - x_{n-1}} = k_i, (i = 1, 2...n) \tag{4}$$

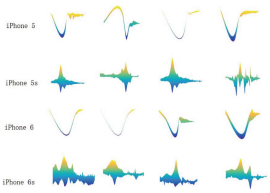


Fig. 3. Front view of 3D feature images

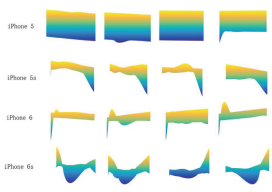


Fig. 4. End view of 3D feature images

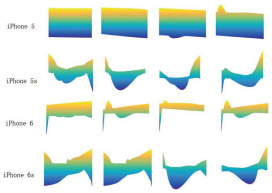


Fig. 5. Vertical view of 3D feature images

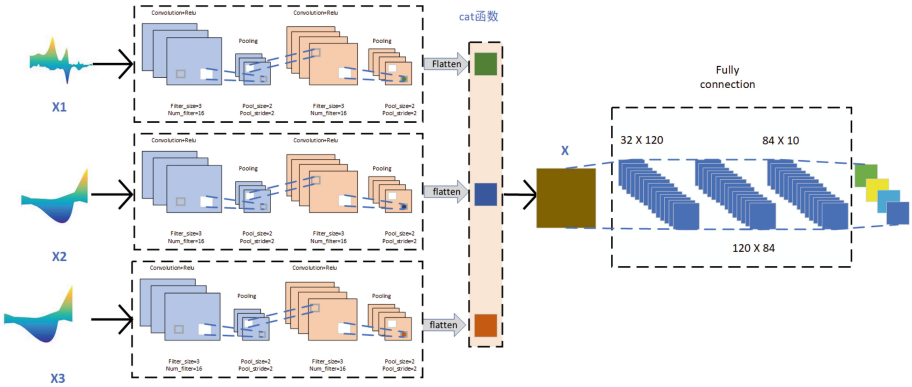


Fig. 6. Multi-channel parallel CNN model

Step 3: Curvature feature extraction. Similar to the slope feature extraction, the curvature feature extraction is also based on discrete point curvature. Two algorithms are proposed to derive the curvature features from discrete points.

Theorem 1. *The curvature of the curve derived from three non-collinear points can be determined by approximating it to a circle using the tangent line of the three-point vector distance. The Formula 5 demonstrates the calculation of curvature. Assuming the vector distances between these points are denoted as a and b. Based on the vector distance, Formula 6 can be used to calculate the curvature of discrete points and obtain the curvature distribution.*

$$k = \frac{\left| x'' y' - x' y'' \right|}{(x')^2 + (y')^2} \tag{5}$$

$$k = \frac{2(a_3 b_2 - a_2 b_3)}{(a_2^2 + b_2^2)^{\frac{3}{2}}} \tag{6}$$

Theorem 2. *The curvature circle can be constructed using the three non-collinear points, and the radius of the circle can be obtained. According to the relationship between the radius and the curvature, the curvature can be derived as* $k = 1/R$.

Step 4: 3D image construction and three views extraction. The three envelope peak feature points from previous steps are inputted as X, Y, and Z coordinates. The scatter function processes the 3D scatter images, which are then used for grid-based 3D image construction. Three views of the 3D image are extracted using the view function, as depicted in Figs. 3, 4, and 5.

3.3 Multi-channel Parallel CNN Model

The RF signal features are represented as spatial stereoscopic images, and three views (front, side, and top) are used to extract spatial features. These three views serve as inputs to the multi-channel parallel CNN model. In Fig. 6, X_1, X_2, and X_3 undergo two convolution and pooling layers in each channel. The flatten function normalizes the obtained feature vectors, and the cat function concatenates the normalized feature vectors. The comprehensive feature, denoted as X, is obtained from these three-channel feature vectors. Three fully connected layers are used to connect the comprehensive feature X. Finally, the model outputs the features of the three input images from the three channels.

4 Experiment Evaluation

4.1 Evaluation Metrics

The identification accuracy in the test stage is defined as Formula 7.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

There are four categories of classification, "TP" stands for true positive of positive class, "TN" stands for true positive of negative class, "FP" stands for false positive of positive class, and "FN" stands for false positive of negative class.

4.2 Experiment Design

Exp. I: The identification accuracy of mobile phones. Mobile devices from two famous manufacturers including Samsung and Apple are used in the experiment. Specifically, 7 different models of iPhone devices and 4 different models of Samsung devices are selected for identification. The stereo feature images extracted from these devices are classified and identified. The effectiveness of the proposed method is evaluated according to the identification accuracy.

Exp. II: The impact of sample size. By comparing several previous RFFI methods based on the CNN model for image construction with different RF characteristics, the experiment evaluates the performance of the proposed method compared with existing benchmark solutions.

4.3 The Performance of MPCNN on Device Identification

Algorithm 1 is used to calculate the curvature of discrete points with three-point parameters, and algorithm 2 is used to solve the curvature of discrete points with the three-point fixed circle.

Table 1. Device identification accuracy based on algorithm 1

Device of type	Training acc.	Test accuracy	Average acc.
iPhone5	99.84%	99.73%	99.63%
iPhone5s	97.32%	98.77%	97.61%
iPhone6	99.36%	99.39%	99.37%
iPhone6s	100%	99.80%	99.96%
iPhone7	99.77%	99.64%	99.74%
iPhone7plus	99.85%	99.62%	99.80%
Note3	99.22%	99.00%	99.18%
S3	99.57%	99.87%	99.63%
S4	99.15%	99.71%	99.26%
Average Acc.	99.44%	99.46%	99.42%

Table 2. Device identification accuracy based on Algorithm 2

Device of type	Training acc.	Test accuracy	Average acc.
iPhone5	97.99%	96.31%	97.65%
iPhone5s	96.21%	95.54%	96.08%
iPhone6	96.39%	95.46%	96.20%
iPhone6s	98.24%	97.58%	98.11%
iPhone7	96.17%	96.15%	96.11%
iPhone7plus	95.53%	94.11%	95.25%
Note3	99.57%	98.84%	99.42%
S3	100%	99.80%	99.96%
S4	96.97%	91.14%	95.80%
Average Acc.	97.47%	96.43%	97.21%

Tables 1 and 2 display the device classification accuracy of the proposed MPCNN method. The input for this method includes spatial stereoscopic feature images obtained from algorithm 1 and algorithm 2 respectively. The performance of the proposed method utilizing these two curvature calculation methods achieves an accuracy of over 95%. Additionally, the MPCNN's classification accuracy using input from algorithm 1 reaches 99%, surpassing the accuracy of input from algorithm 2. This is because algorithm 2 forcibly determines curvature circles, resulting in a loss of curvature features and a decrease in the representativeness of the constructed spatial features.

Table 3. Identification performance of different CNN+RFF methods

Num.	Target	Feature	Dataset	ACC.
1	Smartphone (Blutooth)	Spatial stereoscopic feature image	1650	99.42%
2	Smartphone	PRACH preamble	878	96.33%
3	LoRa	Spectral diagram	1000	97.61%
4	Zigbee	constellation diagram	24300	99.1%
5	USRP	I/Q	399360	99.86%
6	WiFi	I/Q waveform	402000	95%

4.4 The Influence of Sample Size

Table 3 shows the performance of existing CNN-based RFFI methods using various RF signal features. CNN-based RFFI methods utilizing spatial stereoscopic feature images achieve the second-best performance, showcasing the advantages of the proposed RF signal feature extraction and representation method. Figure 7 compares the performance of existing CNN-based RFFI methods and the proposed method with different sample sizes. RFFI methods based on differential constellation maps and I/Q sequences require a sample size of 10^6 for target performance. In contrast, the RFFI method based on the spectrum diagram and PRACH leading code requires a similar sample size as the proposed method for achieving the claimed performance. The latter three methods require considerably smaller sample sizes compared to the former three methods. Furthermore, as depicted in Fig. 8, the proposed method outperforms the other two existing methods (spectrum diagram and PRACH leading) in identification accuracy with a similar sample size. Through comprehensive analysis, it is evident that the proposed method can achieve nearly ideal identification accuracy with a very small sample size, making it more suitable for scenarios with limited practical radio environment samples.

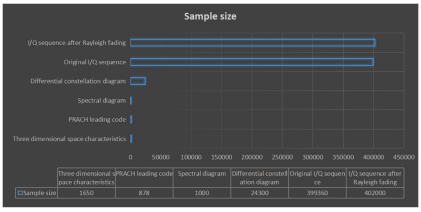


Fig. 7. Sample size comparison

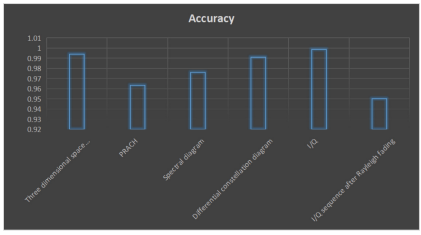


Fig. 8. Accuracy comparison

5 Conclusion

This paper proposes a novel MPCNN-based RFFI method and a new method to construct spatial 3D features from the radio-frequency signal envelope. This method enhances the feature dimension and quantity by constructing multiple features in a three-dimensional space. Simulation results indicate that this approach solves the practical issue of improving recognition performance with limited data samples, while also reducing the complexity of RF signal data preprocessing.

Acknowledgment. This work is funded by the National Key Research and Development Program of China under Grant No. 2021YFB2910103.

References

1. Xie, N., Li, Z., Tan, H.: A survey of physical-layer authentication in wireless communications. IEEE Commun. Surv. Tutorials **23**(1), 282–310 (2021)
2. Peng, L., Zhang, J., Liu, M., Hu, A.: DL based RFFI using differential constellation trace figure. IEEE Trans. Veh. Technol. **69**(1), 1091–1095 (2020)
3. Yu, J., Hu, A., Li, G., Peng, L.: A robust RF fingerprinting approach using multisampling convolutional neural network. IEEE Internet Things J. **6**(4), 6786–6799 (2019)
4. Pan, Y., Yang, S., Peng, H., et al.: Specific emitter identification based on deep residual networks. IEEE Access **7**, 54425–54434 (2019)
5. Youssef, K., Bouchard, L., Haigh, K., et al.: Machine learning approach to RF transmitter identification. IEEE J. Radio Freq. Identif. **2**(4), 197–205 (2018)
6. Peng, L., Zhang, J., Liu, M., Hu, A.: Deep learning based RFFI using differential constellation trace figure. IEEE Trans. Veh. Technol. **69**(1), 1091–1095 (2020)
7. Yang, T., Zhao, J., Wang, X., Xu, F.: Deep learning based RFF recognition with differential constellation trace figure towards closed and open set. IEEE/CIC International Conference on Communications in China (ICCC), pp. 908–913 (2022)
8. Shen, G., Zhang, J., Marshall, A., Peng, L., Wang, X.: Radio frequency fingerprint identification for LoRa using spectrogram and CNN. In: IEEE Conference on Computer Communications (INFOCOM), pp. 1–10 (2021)
9. Shen, G., Zhang, J., Marshall, A., et al.: Radio frequency fingerprint identification for LoRa using deep learning. IEEE J. Sel. Areas Commun. **39**(8), 2604–2616 (2021)
10. Zong, L., Xu, C., Yuan, H.: A RF fingerprint recognition method based on deeply convolutional neural network. In: IEEE Information Technology and Mechatronics Engineering Conference (ITOEC), pp. 1778–1781 (2020)
11. Li, J., Ying, Y., Wang, S., et al.: Slice combination convolutional neural network based radio frequency fingerprint identification for Internet of Things. Wireless Netw. **29**, 1–14 (2023). https://doi.org/10.1007/s11276-023-03241-8
12. Hall, E.S., Vawdrey, D.K., Knutson, C.D., et al.: Enabling remote access to personal electronic medical records. IEEE Eng. Med. Biol. Mag. **22**(3), 133–139 (2003)

Modeling Emergent Behaviour for Enhanced Autonomy in Cyber-Physical Systems

Matthias Schaffeld(✉) and Torben Weis

University of Duisburg-Essen, 47057 Duisburg, Germany
{matthias.schaffeld,torben.weis}@uni-due.de

Abstract. The convergence of Cyber-Physical Systems (CPS) and autonomous systems presents numerous application areas and challenges. Many CPS use cases require autonomy, which necessitates robust monitoring, maintenance, and longevity of these systems. However, handling unmodeled data, environmental uncertainties, and emergent behaviour poses significant challenges. This paper proposes the use of a novel formalism, Extended Hidden Markov Models with ϵ-emissions (ϵ-HMMT), to estimate the correctness of an autonomous system's emergent behaviour in complex and changing environments. We discuss the advantages and disadvantages of different modeling approaches, emphasizing the need for a more adaptable and robust modeling technique.

Keywords: behavioural modeling · monitoring · cyber-physical systems

1 Introduction

As cyber-physical systems (CPS) become more ubiquitous and pervasive, the number of applications that also require them to be more autonomous increases. To support this autonomy, we want to focus on enhancing the monitoring, analysis, verification, and longevity of these autonomous CPS. We argue, that all of these aspects can be enhanced by using robust modeling. There are however many challenges to model CPS. First, when creating and modeling complex CPS models, one has to deal with, incomplete technical data, possibly wrong system assumptions, hidden (physical) processes and interactions, or physical processes that are too intricate to model accurately. Therefore, in modeling, we must rely on simplifying assumptions and develop coarse models that only partially represent the real system's behaviour [7]. Furthermore, these models replicating the real-world engineered CPS, have to account for uncertainties within the physical environment, interactions, and communication systems. The magnitude of uncertainty's influence on a system often hinges on the degree of control attainable over its environment. In the case of autonomous systems, such as CPS interacting with the physical world, controllability typically remains quite constrained.

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 520–528, 2024.
https://doi.org/10.1007/978-3-031-63992-0_35

Consequently, these systems must possess the capability to accommodate data that was not accessible during their initial modeling, development, and testing phases. Additionally, it's essential to acknowledge that these systems are susceptible to parameter drift as a result of alterations and wear and tear. A model that has once been suitable does not necessarily have to remain so.

Secondly, CPS inherently require a combination of models and methods of various engineering disciplines with those of computer engineering, which do not combine well [5]. Even if they did, the system components are inherently interdependent and the emergent behaviour can not easily be determined by the heterogeneous models the individual system components are based on.

In the field of CPS, Model Driven Design is a widely used approach in which different categories of models are used, each of which has its own advantages and disadvantages. Evaluating system behaviour using these models is an important objective. However, due to the intricate nature of CPS, relying solely on any of these approaches or their combinations falls short of accurately predicting the emergent system behaviour during deployment. Hence, our proposed modeling formalism seeks to fill this gap by facilitating the learning and prediction of the emergent behaviour of the comprehensive system. Its purpose is not to encapsulate the entire system with all its components, dependencies, and physical laws. Instead, it aims to seamlessly integrate into existing systems as a supplementary tool, functioning as a heuristic to assess the correctness of emergent system behaviour by effectively capturing it.

2 Key Concepts and Desired Properties

This section presents key concepts that will be addressed in our research, as well as an overview of the desired properties of our formalism.

First, we share the definition of autonomous systems from Müller et al. [6] according to which an autonomous system is a defined technical system capable of autonomously and systematically achieving its predetermined objectives in uncertain environmental conditions. It exhibits self-governance, adapts to its environment, executes processes systematically, and is self-contained while allowing variation in instrumental goals to achieve its primary set objectives.

A CPS is an integration of physical entities and associated processes equipped with computational capabilities that enable a seamless connection between the physical world in which sensors and actuators operate and the virtual world of information processing. It combines elements from different domains, such as embedded computing, physical systems, and communication technologies, to monitor, control, and coordinate operations in a variety of applications, focusing on the interaction between computational components and physical processes. Given that CPS are required to make decisions, adapt, and execute actions to achieve their designated objectives, all while contending with incomplete sensory information, approximate modeling, uncertainty within the physical system, and limitations in environmental control, it becomes evident that there is a significant overlap between autonomous systems and CPS.

Within this system, the cyber elements exhibit discrete dynamics, while the physical aspects demonstrate continuous behaviour. This necessitates a pivotal challenge: discretizing the continuum [7] effectively to facilitate accurate interpretation, prediction, and interaction between the cyber and physical components. However, our focus lies not in exhaustive detailing of every individual component but rather in comprehending the emergent behaviour of the entire system and its constituent parts. Consequently, we adopt a phenomenological perspective, viewing the behaviour of the CPS as a sequence of discrete events.

In the context of CPS design and deployment, we distinguish three types of instances of the same system. The first is the *conceptual instance*, which encapsulates the developers' mental model of the CPS. This model encompasses all knowledge and assumptions related to the desired system behaviour, addressing questions such as the environment's characteristics, the system's objectives, the requisite parameters for cyber components, decision-making processes, action execution methods, and the impact of actions on the environment. Given the inherent complexity and uncertainties of CPS, conceptual instances inevitably incorporate assumptions, approximations, and the potential for errors. Unfortunately, since these are only mental models, their comprehension remains elusive.

Nonetheless, the subsequent form of instance we refer to as *specification instance*, is constructed upon these mental constructs. Specification instances formally articulate the entirety of system components, encompassing the environment, physical processes, and physical and cyber components, as well as their interdependencies, behavioural patterns, constraints, and objectives. Ideally, these specification instances encompass all the information found in the conceptual models providing detailed representations, such as differential equations for physical processes. However, this means that the inherent approximations, assumptions, and errors found in the conceptual instance are also adopted. Furthermore, while these concretize and enhance the tangibility of conceptual instances, they often also sacrifice a degree of precision inherent in their conceptual counterparts. Notably, not every conceptual instance readily translates into a suitable specification instance as a fitting modeling formalism may be missing, and not all types of models seamlessly combine, for example, to represent their interactions. Moreover, existing implicit assumptions remain implicit in the transition to specification instances, and dynamics that are too complex or intricate to capture in modeling require further simplifications, resulting in further loss of information. After all components of the system considered in the conceptual instance have been defined, it becomes possible to conduct a thorough verification process in alignment with best practices. At this point, the intended system behaviour can be verified through the use of simulations and test data. Nevertheless, the uncertainty persists regarding the extent to which the test data accurately mirrors the real-world environment and whether the simulations adequately capture the genuine behaviour of the implemented system.

The third type of instance, which we want to focus on, is the *implementation instance*. These are the realization of their specification counterparts, both in the physical and cyber domains, thereby inheriting the same set of assump-

tions, errors, and the potential for information loss. However, this transition from specification to implementation introduces its own set of challenges. Issues may arise due to resource constraints, misinterpretations of the specifications, implicit assumptions, and makeshift solutions, which often stem from the system's inherent uncertainties and the inherent incompleteness of specification instances. This transformation from conceptual to implementation instance can, therefore, be likened to a game of "Telephone", where information is passed along and becomes distorted in the process. In light of these complexities, we contend that our primary focus should be on capturing, observing, and verifying the emergent behaviour of the implementation instance. This perspective is further reinforced by the fact that even when models and software undergo verification, manufacturers are compelled to validate the actual component behaviour in conjunction with their hardware, recognizing that behaviour may diverge depending on the specific combination of software and hardware elements [5].

Based on this reflection on autonomous systems, CPS, and their instances, we want to develop a modeling formalism that satisfies the following properties:

Capture of Emergent System Behaviour: Emergent system behaviour refers to the system's behaviour that arises from the interactions of its components, which are often not explicitly specified. We aim to develop a modeling formalism that has the ability to capture and represent such emergent behaviours. This is crucial for understanding complex systems where global behaviour cannot be easily deduced from the properties of individual components alone.

Learnability from Data: Our focus is the emergent behaviour of CPS during deployment, which involves handling uncertainties and previously unseen data. However, we lack precise models that fully capture this behaviour. Consequently, a modeling formalism must be equipped with the capacity to derive crucial model parameters and relationships from observed data.

Determination of Deviations: The ability to detect and quantify deviations in system behaviour is an essential requirement for our formalism. It is especially relevant in contexts like emergency management, as highlighted by the example of aging infrastructures increasing the risk of system unreliability [2].

Smoothing of Observed System Behaviour: In many practical applications, observed system behaviour may contain noise or erratic fluctuations. A modeling formalism should incorporate algorithms or mechanics for smoothing this behaviour to enhance the accuracy and analysis of system dynamics.

Confidence Measure in Prediction: To instill trust and reliability in the predictions generated by a suitable modeling formalism, it has to incorporate some form of a confidence measure. This measure provides an indication of the degree of certainty associated with the model's predictions.

Adaptability: As systems evolve a modeling formalism must be able to accommodate changes in system behaviour. This ensures that our modeling approach remains relevant and effective as the system behaviour changes.

Non-disruptive Modeling: The formalism has to be non-invasive, meaning that the system under consideration is not compelled to undergo substantial modifications to conform to the modeling formalism. This principle aligns with the concept of non-disruptive modeling, where the modeling process should not disrupt the normal operation of the system. This also enables the capability of retrofitting, allowing the formalism to be applied to existing systems without necessitating extensive changes.

3 Approach and Challenges

In the realm of modeling emergent behaviour in Cyber-Physical Systems (CPS), a range of modeling techniques presents itself as viable options. The survey conducted by Rai et al. [7] has categorized these techniques into four distinct categories, namely physics-based equations, state machine-based modeling, rule and agent-based modeling, and data-driven modeling.

Physics-based equations excel in capturing the physical dynamics of CPS, while state machines are particularly adept at modeling discrete system dynamics. Rule and agent-based modeling, on the other hand, excel at representing intricate dependencies between cyber and physical components, effectively capturing system semantics. All these categories offer the advantage of traceability in modeling but face challenges when dealing with the complexity of CPS. Moreover, they share a common limitation-they can only represent known system dynamics, leaving out the nuances of real-world system behaviour during deployment.

To address the need for capturing actual system behaviour in deployment scenarios, these models must integrate data-driven learning methodologies. However, the fourth category, data-driven modeling, stands out by virtue of its capability to encompass both continuous and discrete dynamics. Nevertheless, it comes with the drawback of reduced explainability and the requirement for substantial data to achieve accuracy [7].

Furthermore, when modeling CPS behaviour, the choice between deterministic and non-deterministic modeling is a key consideration. In his survey [5], Lee discusses the pros and cons of both approaches, which we briefly outline below.

Deterministic models offer the advantage of clarity and precision, making them valuable for analysis and the development of CPS. They establish a definitive system behaviour providing an unambiguous definition of what "correct" behaviour is. However, deterministic models have inherent limitations, such as potential incompleteness in capturing uncertainties and complexities, particularly in intricate CPS scenarios, where they can grow unwieldy.

In contrast, non-deterministic models excel in accommodating the inherent uncertainties of the real world, offering the ability to represent entire families

of possible behaviours. This quality allows them to deal with sources of non-determinism, including physical noise, component failures, imperfect actuation, and unpredictable delays. Furthermore, non-deterministic models are well-suited for situations where unknown or unknowable properties play a central role. Building on this foundation, Lee asserts that deterministic modeling, when coupled with faithful implementations, remains relevant in the context of CPS and leverages the established advantages of deterministic models. However, our main focus is on capturing the emergent behaviour of CPS that serves as a basis for analysis, verification, and maintenance. Given the inherent complexity and non-deterministic nature of CPS, we argue for the extension of a non-deterministic probabilistic formalism to represent behaviour - Hidden Markov Models (HMM).

Given that state machines are inherent to the computational components of CPS that determine the executed system behaviour, we aim to address existing limitations, such as the lack of robustness and adaptability, associated with traditional approaches. To achieve this, we propose leveraging the ϵ-Hidden Markov Model with Transition Emissions (ϵ-HMMT) which we introduced in previous work [9]. It extends HMMs by incorporating ϵ-emissions, that allow for unobservable transitions, which enable the handling of missing or erroneous observations. It also moves the emissions from states to transitions, facilitating more fine-grained modeling of emissions, leading to an alignment to the more accustomed formalism of the discrete automaton. Furthermore, in our previous work (Bernemann et al. [1]) we extended and verified the Baum-Welch and Viterbi algorithms for ϵ-HMMT. Consequently, the Baum-Welch algorithm can learn system behaviour by observing event sequences while accounting for ϵ-emissions. The Viterbi algorithm on the other hand considers both past and future emissions when determining the best explanation for a sequence of observations.

However, the use of HMM comes with inherent limitations. This approach assumes that the system's behaviour is observable through discrete events. Furthermore, it assumes the Markov property, meaning that the next event depends solely on the previous one. Deviations from this assumption necessitate an exponential increase in the number of states. Currently, the model also does not account for temporal dependencies. Nonetheless, we plan to extend the formalism with techniques such as Good Turing and Back-off Estimation.

In the survey done by Gunes et al. [3], which describes 21 challenges in the field of CPS, we identified seven challenges that we perceive as pertinent to our research objectives focused on enhancing CPS autonomy through the modeling of emergent behaviour. These challenges are succinctly summarized below and elaborated on within the context of our proposed methodology:

Predictability: Predictability denotes the extent to which one can predict the system's state, behaviour, or functionality, through qualitative or quantitative means. Problems such as the inherent uncertainty of the environment and non-deterministic processes must be dealt with to achieve predictability.

Considering the inherent utilization of probabilities and the incorporation of ϵ-emissions, the ϵ-HMMT represents a significant step towards addressing the

challenges posed by non-determinism and uncertainty in CPS. By virtue of the Viterbi algorithm's ability to consider past and future observations, including potentially erroneous ones, when determining the optimal explanation for an observed sequence, we can harness this capability to introduce confidence measures for various explanations, current states, and predictions.

Accuracy denotes the extent to which the observed/measured behaviour of the system matches its calculated/actual behaviour. Accuracy assessment using the ϵ-HMMT presents a straightforward process. It involves a comparison between the observed system behaviour and the behaviour computed using the Viterbi algorithm. This assessment can be done through various approaches, ranging from a comprehensive method employing a window-based strategy in conjunction with system-specific knowledge to simpler techniques such as measuring the edit distance between the calculated and observed event sequences.

Dependability denotes the system's ability to consistently deliver required functionalities, maintain performance, and produce expected outcomes without significant degradation throughout its operation. It represents the extent of trust that can be placed in the entire system. Dynamic interdependencies between cyber and physical components as well as issues like timing uncertainties in sensor data and actuation make achieving dependability a challenging task.

After applying the extended Baum-Welch algorithm to learn the emergent system behaviour, one can subsequently assess the model's accuracy and employ it as a benchmark for monitoring the system. If there is a decline in the accuracy of the system or its components, targeted maintenance can be scheduled to ensure dependability. By incorporating a confidence metric rooted in the likelihood of observation explanations as determined by the Viterbi algorithm, we can establish a quantifiable confidence indicator of the system behaviour's correctness.

Maintainability denotes the system's ability to perform repairs and recoveries in the event of failures. This includes continuous monitoring and testing of the infrastructure, which enables the identification and remediation of components that are prone to recurring failures. While monitoring the system against a learned benchmark, one can schedule targeted maintenance if the accuracy of the system or specific components drops. The extent to which erroneous behaviours and components can be identified and isolated remains to be evaluated.

Robustness denotes the system's capacity to maintain stability and functionality, even in the presence of failures, disturbances, and unforeseen events. A robust CPS effectively handles failures and disruptions, including sensor noise, actuator inaccuracies, communication issues, and hardware errors. Robustness is an essential CPS challenge due to the incomplete knowledge about the system and its environment during design, leading to unavoidable disturbances [8].

Given that the ϵ-HMMT offers ϵ-emissions and introduces more intricate modeling through transition emissions, it holds the potential to increase robustness. This enhancement is achieved by enabling established HMM algorithms to proficiently manage imperfect or erroneous data while supporting intricate modeling.

Adaptability denotes the system's ability to (autonomously) modify its configuration to deal with the continuous change of environmental conditions.

The behavioural model embedded within a trained ϵ-HMMT aims to validate the anticipated system state, refine potentially inaccurate observation data, and assess system accuracy, enabling crucial adjustments. Reiteratively training the model with data from a modified CPS while leveraging the prior model as input offers valuable insights into the extent of CPS modifications. Additionally, this approach introduces adaptability to a state machine-based modeling paradigm.

Reliability denotes the system's ability to consistently perform its intended function in uncertain, evolving environments. The system has to be able to manage uncertainties such as incomplete knowledge and parameter variations. The limits of modeling uncertainty, component accuracy, network connections, and software errors are all factors that decrease reliability [4].

Selecting a modeling formalism that embraces uncertainty as a CPS tool and enhancing its robustness holds significant potential for evaluating and bolstering CPS reliability. By striving to address at least some, if not all, of the aforementioned challenges, we can anticipate an improvement in reliability as well.

4 Conclusion

In conclusion, we emphasized the need to model emergent behaviour in CPS to promote their autonomy and recognize the inherent challenges and uncertainties associated with modeling these systems. We defined key concepts, addressed challenges in transitioning from conceptual models to real-world implementation, and outlined the desired properties for our proposed modeling formalism. We also explored a range of modeling techniques and discussed the choice between deterministic and non-deterministic modeling, ultimately advocating for the use of an extended HMM formalism (ϵ-HMMs) to capture the dynamic emergent behaviour of CPS. In parallel, we identified CPS challenges including dependability, predictability, accuracy, reliability, robustness, adaptability, and maintainability, and discussed the application of our methodology. Our forthcoming work will focus on evaluating the limitations of our approach as a basis for further useful and needed extensions to the formalism.

References

1. Bernemann, R., König, B., Schaffeld, M., Weis, T.: Probabilistic systems with hidden state and unobservable transitions. arXiv:2205.13871 (2022)
2. F.E.M.A.: Crisis response and disaster resilience 2030: forging strategic action in an age of uncertainty. Tech. rep., The Strategic Foresight Initiativ (SFI) (2012)
3. Gunes, V., Peter, S., Givargis, T., Vahid, F.: A survey on concepts, applications, and challenges in cyber-physical systems. KSII Trans. Internet Inf. Syst. **8**, 4242–4268 (2014). https://doi.org/10.3837/tiis.2014.12.001

4. Sztipanovits, J., Ying, S., I.C.D.C.J.D.H.K.P.J.M.V.P.L.S.: Strategic r&d opportunities for 21st century cyber-physical systems. Tech. rep., Steering Committee for Foundations in Innovation for Cyber-Physical Systems. (2013)
5. Lee, E.A.: The past, present and future of cyber-physical systems: a focus on models. Sensors **15**(3), 4837–4869 (2015). https://doi.org/10.3390/s150304837
6. Müller, M., Müller, T., Talkhestani, B.A., Marks, P., Jazdi, N., Weyrich, M.: Industrial autonomous systems: a survey on definitions, characteristics and abilities. at - Automatisierungstechnik **69**(1), 3–13 (2021). https://doi.org/10.1515/auto-2020-0131
7. Rai, R., Sahu, C.K.: Driven by data or derived through physics? A review of hybrid physics guided machine learning techniques with cyber-physical system (CPS) focus. IEEE Access **8**, 71050–71073 (2020)
8. Rungger, M., Tabuada, P.: A notion of robustness for cyber-physical systems. IEEE Trans. Autom. Control **61**(8), 2108–2123 (2016)
9. Schaffeld, M., Bernemann, R., Weis, T., König, B., Matkovic, V.: Lifecycle-based view on cyber-physical system models using extended hidden Markov models. In: 2022 20th ACM-IEEE International Conference on Formal Methods and Models for System Design (MEMOCODE) (2022)

On-Device Soft Sensors: Real-Time Fluid Flow Estimation from Level Sensor Data

Tianheng Ling(✉), Chao Qian, and Gregor Schiele

Intelligent Embedded Systems Lab, University of Duisburg-Essen,
47057 Duisburg, Germany
{tianheng.ling,chao.qian,gregor.schiele}@uni-due.de

Abstract. Soft sensors are crucial in bridging autonomous systems' physical and digital realms, enhancing sensor fusion and perception. Instead of deploying soft sensors on the Cloud, this study shift towards employing on-device soft sensors, promising heightened efficiency and bolstering data security. Our approach substantially improves energy efficiency by deploying Artificial Intelligence (AI) directly on devices within a wireless sensor network. Furthermore, the synergistic integration of the Microcontroller Unit and Field-Programmable Gate Array (FPGA) leverages the rapid AI inference capabilities of the latter. Empirical evidence from our real-world use case demonstrates that FPGA-based soft sensors achieve inference times ranging remarkably from 1.04 to 12.04 μs. These compelling results highlight the considerable potential of our innovative approach for executing real-time inference tasks efficiently, thereby presenting a feasible alternative that effectively addresses the latency challenges intrinsic to Cloud-based deployments.

Keywords: On-Device AI · Soft Sensors · Real-Time Inference · FPGA-Based Inference

1 Introduction

Soft sensors, envisioned as virtual constructs, utilize algorithms and computational models to infer impalpable values by processing data derived from physical sensors [3]. They have gained traction in ubiquitous computing, offering invaluable assistance where obtaining direct measurements is either impractical or economically burdensome, enriching data-driven insights in various systems [5]. Integrating Deep Learning (DL) techniques into soft sensors has spurred a movement toward Cloud-based deployments [5]. Since these soft sensors function as services on the Cloud, they inherit undeniable advantages: scalable computational resources, vast storage for training data, enhanced data analytic capabilities for model refinement, and flexible access to data and insights [11].

However, Cloud service effectiveness is closely linked to the stability and reliability of network conditions. Even slight instability in network connections can significantly delay the transmission and processing of sensor data, adversely

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 529–537, 2024.
https://doi.org/10.1007/978-3-031-63992-0_36

affecting the real-time applications that rely on swift and accurate sensor data for immediate decision-making tasks. Furthermore, the low transmission rates of many wireless sensor networks present substantial challenges for the continuous, real-time transmission of extensive sensor data to the Cloud. For example, LoRaWAN achieves transmission rates of 0.3 kbps to 50 kbps [7].

In response to these pressing challenges, this paper promotes a shift towards implementing on-device soft sensors. We introduce a novel approach that seamlessly integrates low-power Microcontroller Units (MCUs) with embedded Field-Programmable Gate Arrays (FPGAs). This integration allows for the execution of AI functionalities directly on devices, thereby reducing latency, enhancing reliability, and bolstering real-time system performance. These improvements collectively offer a robust and efficient alternative to the prevalent issues associated with Cloud-based approaches. To substantiate the viability and efficacy of this approach, this study delves into a practical application: estimating fluid flow using level sensor data. Furthermore, we evaluate model performance, energy efficiency, and timing on both MCU and FPGA platforms, highlighting the tangible benefits of our innovative strategy in real-world scenarios.

The remainder of this paper is structured as follows: Sect. 2 reviews the literature relevant to our study. Section 3 outlines the proposed system architecture, while Sect. 4 presents a case study demonstrating the effectiveness of our approach. Finally, Sect. 5 concludes the paper and discusses future research.

2 Related Work

The burgeoning soft sensors have integrated into various domains [10]. Despite their invaluable capabilities, deploying these sensors in Cloud-based environments invites challenges, predominantly network latency and data security, which often diminish their benefits, necessitating effective alternatives.

Recent research signals a shift towards on-device soft sensors. Aguasvivas Manzano et al. [1] exemplified this by deploying Neural Network models on ARM Cortex-M4 MCUs, realizing soft sensors with 7ms inference time and 45.4 mW power consumption. Flores et al. [4] applied Tiny ML-based soft sensors in vehicles, implementing a Multi-layer Perceptron (MLP) model on a 32-bit MCU (ESP32) for efficient CO2 emissions estimation, showcasing a decreased model inference time to 37 μs by tweaking the model size. Balaji et al. [2] introduced a novel approach, integrating reconfigurable hardware accelerators for fast on-device AI inference in a wearable interface, with initial prototypes using FPGAs for emulation, marking significant improvements in speed and efficiency.

Drawing inspiration from these works, we introduce an innovative approach for implementing on-device soft sensors by integrating MCUs with embedded FPGAs. To validate this, we focused on estimating fluid flow using level sensor data, a critical task in our research project, as directly using flow sensors in underground sewer systems introduces challenges due to their high cost, maintenance demands, and vulnerability to harsh conditions [12]. While traditional physical and statistical models have been extensively utilized, the complexity

of real-world phenomena, amplified by factors like human activities, poses challenges for traditional linear models to capture non-linear and intricate patterns. There is a discernible gap in the literature concerning DL-based research, with a need for dedicated public data sets exacerbating the issue. Notably, even fewer studies explore on-device AI inference to address this demand.

We commenced our research with the data set released by Noori et al. [6]. Although limited in size, it offers a starting point for our investigation. Using an MLP model with a non-contact radar sensor to estimate non-Newtonian fluid flow in open Venturi channels, their work provides a foundational baseline. Our study further explores the potential of on-device soft sensors for real-time data processing tasks, responding to the challenges presented by Cloud-based IoT deployments, and aims to contribute to the ongoing discourse in the field.

3 System Architecture

This section presents the architecture of our on-device soft sensor system. We begin with defining and elucidating soft sensors, then delve into the process of localizing these sensors on target hardware. This examination will highlight our proposed system architecture's distinctive features and advantages.

3.1 Architecture Overview

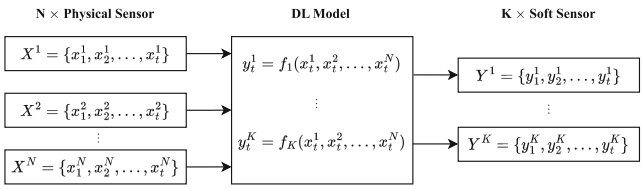


Fig. 1. General System Architecture of Soft Sensors

Figure 1 presents our proposed general system architecture, designed to generate K independent soft sensors from data collected by N sensors. In this architecture, X^1 to X^N represent discrete time series obtained by sampling N sensors at consistent intervals of T. We introduce K independent fusion functions, denoted as f_1 through f_K, each designed to integrate the N inputs at any given time t. These functions yield new features serving as the output for specific soft sensors at each time point. Maintaining the initial sampling frequency, the architecture illustrated in Fig. 1 effectively transforms an N-dimensional time series into a K-dimensional one. Crucially, when K is less than N, this results in a reduction of the data volume required for wireless transmission, diminishing it to a K/N fraction of the original volume. These fusion functions must be completed within period T to meet the real-time requirement.

In our approach, these fusion functions are actualized within the proposed system by implementing DL models. The training phase of these models occurs on the Cloud during the initial development stage of our soft sensor system. After attaining satisfactory training outcomes, the models are deployed onto IoT devices, thereby operating as efficient on-device soft sensors.

3.2 Elastic Node AI Hardware Platform

Figure 2 illustrates the fifth version of the *Elastic Node*, a hardware platform meticulously designed for on-device AI development [9]. This platform primarily employs the Cortex-M0 MCU (RP2040), efficiently controlling integrated resources like sensors, a wireless module, and the Spartan-7 S15 FPGA. The FPGA plays a pivotal role, serving as configurable hardware for DL accelerators to significantly enhance the speed and energy efficiency of DL inference tasks. For detailed hardware insights, refer to Fig. 3, which provides a schematic of the soft sensor's hardware implementation. The sensors, labeled S_1 to S_N, interface with the MCU via a digital data bus (e.g., I2C or SPI) or analog channels connected to the MCU's internal ADC.

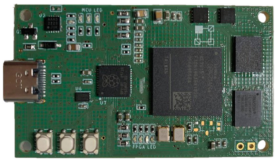


Fig. 2. Elastic Node V5

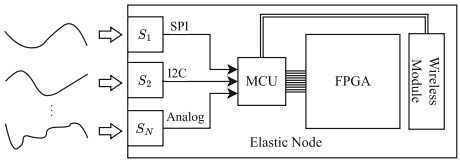


Fig. 3. Elastic Node V5 with Soft Sensors

Regarding operational dynamics, the FPGA remains inactive until the MCU activates it upon accumulating sufficient raw data. Depending on the computational needs, the MCU either directly proceeds with model inference or delegates this task to the accelerator, necessitating the FPGA's activation and configuration. After processing, the soft sensor's output can be wirelessly transmitted to the Cloud. Furthermore, due to its reconfiguration feature, the FPGA can seamlessly switch between various DL models.

3.3 Model Deployment on Elastic Node

Leveraging Elastic Node V5's heterogeneous architecture, which includes MCUs and FPGAs, our soft sensors can execute on either platform, providing deployment flexibility. Figure 4 illustrates the simplified training and deployment process of DL models onto MCUs using TensorFlow (TF) Lite, facilitating a seamless transition from development to implementation.

We initiate the process by designing and training a DL model for a specific application and performance criterion. For instance, the Mean Square Error

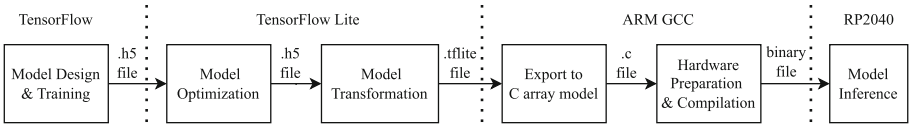


Fig. 4. Workflow of DL Deployment on MCU

(MSE) between the model output and test label must be within a set threshold. Using TF Lite's APIs, we can optimize the model using techniques like quantization and pruning for MCU deployment. Due to the absence of file system support on MCU, the model should be converted from the *.tflite* format to a C array. The C array is then incorporated into a project with runtime programs and dependencies, facilitating the generation of an executable binary compatible with the MCU. This process culminates in the binary's firmware to the MCU for on-site computations.

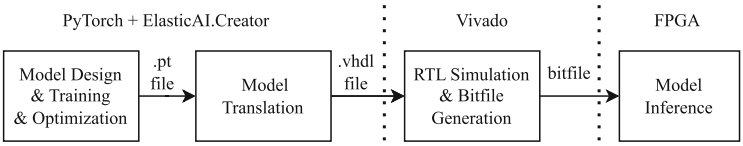


Fig. 5. Workflow of DL Deployment on FPGA

Crafting efficient FPGA hardware accelerators is intricate. To simplify this challenge, we employed the *ElasticAI.Creator* toolchain [8], tailored for the PyTorch (PT) framework, as illustrated in Fig. 5. This tool facilitates the intuitive design of models using its supported layer components. The toolchain manages the model's training and optimization to ensure it is suited for the embedded FPGA specifications. As a result, it outputs VHDL files representing the hardware accelerator at the Register-Transfer Level (RTL). These RTL designs are subsequently synthesized by the Vivado design suite into a bitfile. With the capacity to store multiple bitfiles on one flash, the MCU can direct the FPGA to switch between them as needed, thereby emulating different soft sensors. In this configuration, the MCU predominantly handles physical sensor data transmission to the FPGA and triggers the inference process, while the FPGA undertakes the computational-heavy tasks, signaling the MCU once finished.

4 Case Study: Fluid Flow Estimation

This section delves into a case study centered on real-time fluid flow estimation via level sensor data. We begin by describing the data set used, then outline our experimental setup, and conclude by presenting and evaluating the results garnered from our approach.

4.1 Data Set and Experiments Settings

Noori et al. [6] simulated drilling conditions using an advanced flow loop apparatus comprising a mud tank, blender, centrifugal pump, and an open Venturi channel. Three level sensors were utilized: an ultrasonic level meter, a VEGAPULS 64 radar, and a KROHNE Optiwave radar. The data from these sensors served as input data, while a precise Coriolis mass flow meter provided the target data. Data was collected at a 10kHz sampling rate. We used 1,800 samples from their study, allocating 80% for training and 20% for testing.

We employed a simple MLP model analogous to the one used in the study by Noori et al. [6], comprising one hidden layer and an output layer with a single neuron. Experiments were conducted with hidden layers of 10 (as initially used), 30, 60, and 120 neurons to investigate the impact of model size on performance, to deploy on-device soft sensors on both MCU and FPGA.

For MCU-based soft sensors, training was conducted under the TF framework, utilizing the *Adam* optimizer with parameters $\beta_1 = 0.9$, $\beta_2 = 0.98$, and $\epsilon = 10^{-9}$. An initial learning rate of 0.001, step size of 3, and gamma of 0.5 were set for learning rate scheduling. We used the MSE loss function and also employed MSE as the evaluation metric. After executing 100 experiments across 500 epochs with early stopping, the optimal full-precision (TF_{FP32}) model was selected. This model was subsequently quantized to 8-bit integer formats (TF_{quant}) using post-training quantization (PTQ). For FPGA-based soft sensors, the best full-precision (PT_{FP32}) model was obtained under identical experimental conditions but within PT framework. Notably, the quantized (PT_{quant}) model was generated using fixed-point (6,8) quantization-aware training (QAT), where 8 represents the total bit width and 4 is allocated for fractional bits.

4.2 Results and Evaluation

Figure 6 illustrates the MSE of our MLP models with various configurations of hidden neurons, trained under both TF and PT frameworks. The MSEs of our models are competitive compared to the study by Noori et al. [6], which achieved a test MSE as low as 78.13 under a similar setup. Within the TF framework, both TF_{FP32} and TF_{quant} models exhibit minor fluctuations in MSE across different numbers of hidden neurons, indicating consistent performance. In contrast, within the PT framework, PT_{FP32} models show more substantial variance in MSE. Notably, except for the model with 10 hidden neurons, PT_{FP32} models perform better compared to their counterparts. However, a significant increase in MSE is observed among the PT_{quant} models, suggesting accuracy degradation due to the applied quantization strategy. Furthermore, both PT_{FP32} and PT_{quant} models exhibit considerable MSE variations with different hidden neuron counts, underscoring their sensitivity to model size adjustments.

These insights emphasize the crucial role of selecting the appropriate framework, quantization strategy, and model size in enhancing on-device soft sensors. Building on these findings, Table 1 presents the performance metrics of soft sensors deployed on MCUs, as well as the estimated performance of FPGA-based

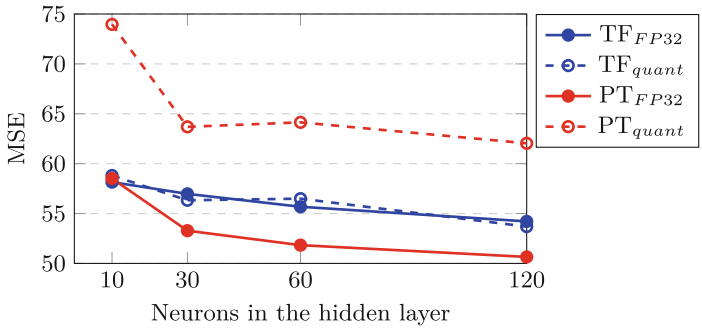


Fig. 6. MSE Values Under TF and PT Frameworks Across Various Hidden Neurons

Table 1. Performance Comparison with Various Hidden Neurons

Hardware	Hidden Neurons	Inference time (μs)	Power (mW)	Energy (μJ/inference)
MCU	10	29.58	119	3.52
	30	56.35	119	6.71
	60	96.48	119	11.48
	120	176.89	119	21.05
FPGA	10	1.04	28	0.03
	30	3.04	29	0.09
	60	6.04	29	0.18
	120	12.04	29	0.35

soft sensors derived from Vivado simulations. As the number of hidden neurons escalates, the inference time on MCUs increases. Specifically, with 60 neurons, the inference time approaches 96.48 μs, closely aligning with the real-time requirement of 100 μs at a 10 kHz sampling rate. In stark contrast, the inference on FPGA is expedient, being up to 28.44× faster than its MCU counterpart depending on the neuron count, thereby satisfying real-time requirements even with 120 hidden neurons. While the power consumption of the inference on MCU remains consistently at 119 mW regardless of neuron number, the consumption of the inference on FPGA peaks at a mere 29 mW. This combination of rapid inference speed and low power consumption endows the FPGA with superior energy efficiency per inference relative to MCU.

5 Conclusion and Future Work

This study highlights the viability and efficiency of on-device soft sensors for real-time fluid flow estimation. Through rigorous experimentation with various MLP models on MCUs and FPGAs, we conclusively demonstrate the superior

inference speed and energy efficiency afforded by FPGA deployments. Nonetheless, the PT_{quant} models necessitate significant performance improvements when compared to their TF_{quant} counterparts. Our findings also shed light on the subtle trade-offs and advantages of different hardware platforms and model configurations, providing valuable insights for future research and applications. Looking forward, we aim to refine and improve quantization strategies for FPGA implementations. This endeavor seeks to mitigate accuracy losses due to the quantization process, all while continuing our dedication to developing robust and dependable on-device AI applications designed for fluid flow estimation tasks.

Acknowledgments. The authors gratefully acknowledge the financial support provided by the Federal Ministry for Economic Affairs and Climate Action of Germany for the RIWWER project (01MD22007C).

References

1. Aguasvivas Manzano, S., et al.: Toward smart composites: small-scale, untethered prediction and control for soft sensor/actuator systems. arXiv e-prints arXiv:2205.10940 (2022)
2. Balaji, A.N., Peh, L.S.: Ai-on-skin: towards enabling fast and scalable on-body AI inference for wearable on-skin interfaces. Proc. ACM Hum. Comput. Interact. **7**(EICS), 1–34 (2023)
3. Becker, T., Krause, D.: Softsensorsysteme-mathematik als bindeglied zum prozessgeschehen. Chem. Ing. Tec. **82**(4), 429–440 (2010)
4. Flores, T., et al.: A TinyML soft-sensor for the internet of intelligent vehicles. In: 2022 IEEE International Workshop on Metrology for Automotive, pp. 18–23. IEEE (2022)
5. Graziani, S., Xibilia, M.G.: Deep learning for soft sensor design. In: Pedrycz, W., Chen, S.-M. (eds.) Development and Analysis of Deep Learning Architectures. SCI, vol. 867, pp. 31–59. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-31764-5_2
6. Noori, N., Waag, T., Viumdal, H., Sharma, R., Jondahl, M., Jinasena, A.: Non-newtonian fluid flow measurement in open venturi channel using shallow neural network time series and non-contact level measurement radar sensors. In: SPE Norway Subsurface Conference. OnePetro (2020)
7. Phung, K.H., Tran, H., Nguyen, Q., Huong, T.T., Nguyen, T.L.: Analysis and assessment of lorawan. In: 2018 2nd International Conference on Recent Advances in Signal Processing, Telecommunications & Computing, pp. 241–246. IEEE (2018)
8. Qian, C., Einhaus, L., Schiele, G.: ElasticAI-Creator: optimizing neural networks for time-series-analysis for on-device machine learning in IoT systems. In: Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems, pp. 941–946 (2022)
9. Qian, C., Ling, T., Schiele, G.: ElasticAI: creating and deploying energy-efficient deep learning accelerator for pervasive computing. In: 2023 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops), pp. 297–299. IEEE (2023)
10. Sharma, A., et al.: Recent trends in AI-based intelligent sensing. Electronics **11**(10), 1661 (2022)

11. Sun, Q., Ge, Z.: A survey on deep learning for data-driven soft sensors. IEEE Trans. Industr. Inf. **17**(9), 5853–5866 (2021)
12. Tomperi, J., Rossi, P.M., Ruusunen, M.: Estimation of wastewater flowrate in a gravitational sewer line based on a low-cost distance sensor. Water Pract. Technol. **18**, 40–52 (2022)

Cultural Influence on Autonomous Vehicles Acceptance

Chowdhury Shahriar Muzammel(✉), Maria Spichkova, and James Harland

RMIT University, Melbourne, Australia
s3987367@student.rmit.edu.au

Abstract. Autonomous vehicles and other intelligent transport systems have been evolving rapidly and are being increasingly deployed worldwide. Previous work has shown that perceptions of autonomous vehicles and attitudes towards them depend on various attributes, including the respondent's age, education level and background. These findings with respect to age and educational level are generally uniform, such as showing that younger respondents are typically more accepting of autonomous vehicles, as are those with higher education levels. However the influence of factors such as culture are much less clear cut. In this paper we analyse the relationship between acceptance of autonomous vehicles and national culture by means of the well-known Hofstede cultural model.

Keywords: Autonomous vehicles · Cultural aspects · Hofstede model

1 Introduction and Motivation

Autonomous vehicles (AVs) and intelligent transport systems are an increasing trend globally. These include driverless cars, trains and trams, as well as more adventurous proposals such as pilotless air taxis for up to 4 people. A key factor in the success of any such system, especially for mass transit, is acceptance and trust in these systems by the general public. Hence it is important to analyse and understand attitudes towards AVs, and what may be influencing them.

There are several existing studies examining this issue. These include a case study on public opinion on the development of an intelligent transport system in Saudi Arabia [2,39], surveys conducted in Australia [3], Turkey [13,26], Jordan [1], Greece [17,33], etc. These studies demonstrate that perceptions of AVs may be influenced by a number of factors, including age, level of education, gender, and culture of the survey participants. It is noticeable that these various studies generally show a similar trend with respect to age and level of education, i.e. that younger respondents are generally more open than older ones to the usage of autonomous vehicles, and that respondents with university degrees are more accepting of AVs than those without such qualifications. However, there appears to be no such clear consensus with respect to cultural aspects.

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 538–547, 2024.
https://doi.org/10.1007/978-3-031-63992-0_37

Contribution: In this paper, we present our analysis of the acceptance of AVs on the basis of Hofstede's cultural model [20]. This model has been widely used for various analyses, including those in IT and Software Engineering.[1].

2 Background: Hofstede's Model

There are several views on how the term *culture* can be defined. It is often analysed from many perspectives, including national, regional, ethnic, religious, linguistic, and social class. One of the most popular aspects is *national culture* [20,40]. There are several approaches to define and model national and organisational culture [19,21,25]. One of the most accepted definitions and models of culture was introduced by Hofstede [20]. Hofstede's model covers more than 80 countries and has been applied in many culture-related studies in the field of IT and Software Engineering (and elsewhere) for comparison between cultures and to understand the impact of cultural differences [4–10,12,14,24,34,36,37]. Hofstede defines **"national culture"** as *"the collective programming of the mind which distinguishes the members of one group or category of people from another"*. In this sense, national culture refers to the beliefs and values that distinguish one nation from another.

Hofstede's model consists of six dimensions:

- **Power Distance Index (PDI):** the degree to which people within a society expect and accept that power is unequally distributed;
- **Individualism/Collectivism (IDV):** the degree to which people within a society collaborate with each other;
- **Masculinity/Femininity (MAS):** the degree to which the society stresses achievement or nurture;
- **Uncertainty Avoidance Index (UAI):** the degree to which people within a society feel either uncomfortable or comfortable in new or unknown situations;
- **Long-/Short-term Orientation (LTO):** the degree to which people within a society are linked to its own past while dealing with the present and the future challenges; and
- **Indulgence/Impulses (IND):** the degree to which people within a society have fun and enjoy life without restrictions and regulations.

Scores for each dimension vary between 0 to 100, with 50 as an average. We obtained the latest Hofstede's values corresponding to the countries associated with this study from their official website known as "Hofstede-Insights", as shown in the Table 1. The analysis used by Hofstede is that if a score is below the average, the culture is considered low on that dimension. For example, Australia, with a score of 38, is considered low on the PDI dimension. The record of scores for all dimensions results in what is called the "cultural profile" for a

[1] According to Google Scholar, Hofstede's work has been cited more than 243,000 times, retrieved 11/10/2023.

specific culture. Israel currently lacks a score in the IND dimension, so it is mentioned as not available (N/A). Pakistan has a score of 0 in the IND dimension, characterising it as a notably restrained society.

Table 1. Hofstede's PDI, IDV, MAS, UAI, LTO, IND scores

Country	PDI	IDV	MAS	UAI	LTO	IND
Australia	38	90	61	51	21	71
Canada	39	80	52	48	36	68
China	80	20	66	30	87	24
France	68	71	43	86	63	48
Germany	35	67	66	65	83	40
India	77	48	56	40	51	26
Indonesia	78	14	46	48	62	38
Israel	13	54	47	81	38	N/A
Japan	54	46	95	92	88	42
Pakistan	55	14	50	70	50	0
Poland	68	60	64	93	38	29
South Korea	60	18	39	85	100	29
UK	35	89	66	35	51	69
USA	40	91	62	46	26	68

3 Cultural Dimensions Vs. Acceptance of AVs

Some existing studies have concluded that there is a dependency between AV acceptance and the Hofstede dimensions. In [42], the authors proposed a few hypotheses regarding the acceptance of AVs, drawing on the Hofstede cultural model. They conducted a case study in South Korea and compared the results with another case study [31] covered Australia, Japan, UK, China, USA, and India. The results of their cross-cultural analysis indicated that high scores of PDI and LTO *positively influence* AV acceptance (i.e. higher scores of cultural dimension mean higher AV acceptance), while high scores of IDV, MAS, UAI, and IND have a *negative influence* (i.e., higher scores of cultural dimension mean lower AV acceptance). For simplicity of the further analysis we denote this as

$$PDI^{+},\ IDV^{-},\ MAS^{-},\ UAI^{-},\ LTO^{+},\ IND^{-}$$

The results of another study [35], which covered Japan, UK, and Germany, indicated that the population of countries with higher values of PDI and MAS

dimensions will be more accepting towards AVs. The authors found no influence of IDV, UAI, LTO, or IND dimensions, i.e.

$$PDI^{+},\ IDV^{0},\ MAS^{+},\ UAI^{0},\ LTO^{0},\ IND^{0}$$

Thus, both [35,42] agree regarding PDI^{+}, but disagree regarding the potential impact of other dimensions.

Table 2. Acceptance of AVs: Combined results of case studies

Country	Ref.	*Acc*, %		Average *Acc*, %	Median deviation
		PAV	FAV		
Australia	[31]		41		
Japan	[35]	40.2	30.2	48.515 (PAV)	8.314 (PAV)
	[23]	56.83			
	[31]		41	35.6 (FAV)	5.4 (FAV)
UK	[35]	33.4	30.7	35.35 (FAV)	4.65 (FAV)
	[31]		40		
Germany	[35]	29	25.8		
Israel	[23]	74.3			
Indonesia	[32]		42 (SAV)		
	[11]	50.3	59.2		
Pakistan	[41]		71 (SAV)		
	[29]	64	25.8		
China	[41]		86 (SAV)		
	[31]		76		
USA, Canada and Israel	[18]	66	53		
USA	[31]		44	43 (FAV)	1 (FAV)
	[22]		42		
India	[31]		80		
Poland	[15]		68		
France	[28]		68.1		
South Korea	[42]		82.2		

We conducted a secondary study to analyse (wrt. Hofstede dimensions) the combined results of 12 studies covering 14 countries: Australia, China, Canada, Germany, France, India, Indonesia, Israel, Japan, Pakistan, Poland, South Korea, UK, and USA, see [11,15,18,22,23,28,29,31,32,35,41,42]. The overview of the core results are presented in Table 2, where *Acc* denotes acceptance of AVs.

Please also note that two of the studies, conducted in Indonesia [32] as well as in China and Pakistan [41] analysed AV acceptance for very specific settings: the use of AVs for shared autonomous vehicles (SAVs), i.e. as a part of a

shared mobility service, such as a ride-hailing service, car-sharing platform, or autonomous shuttle.

In what follows, we use the following abbreviations:

- *FAV* refers to a fully automated vehicle, which is also sometimes called a *Level 5 AV*. An FAV can perform all driving tasks under all conditions without any human intervention.
- *PAV* refers to a partially automated vehicle, which corresponds to a *Level 2, 3, or 4 AV*, having varying degrees of automation but still require some level of human involvement.

In some of the studies, the authors analyse acceptance of FAVs, however, some studies present more nuanced approached where the acceptance of partial automation is analysed as well.

Noticeably, even when the results of different case studies within the same culture/country differ, this difference is not as large as in all countries. To understand the difference, we calculated the median deviation between those common countries of our collected case studies. Among several studies covering the same culture, the USA has the least (1%) and Japan has the greatest (8.314%) median deviation. However, if we compare the results across all 12 studies, the differences in level of acceptance will be substantial: the lowest level has been identified in Germany (PAV: 29%, FAV: 25.80%) by [35], and the highest level has been identified in China for shared vehicles (86%, [41]) and in South Korea (82.2%, [42]). Also, if we compare results of a study [18] conducted in USA and Israel (where the acceptance of AVs was analysed in general, over both countries) with the results of the studies conducted in USA only [22,31] and Israel only [23], we see that the results of combined study are in between of the results for each separate country. This leads us to a conclusion matching the finding presented in [42] that culture plays an important role in accepting AVs, as different national cultures have different level of acceptance of AVs.

Figure 1 presents scatter diagrams to analyse potential correlations among cultural dimensions and AV acceptance rates, where the blue, green and yellow circles present acceptance of FAV, PAV and SAV respectively, based on the data from Tables 1 and 2. For the countries covered by multiple studies, we presented the average value of the acceptance rate. Based on the examination of the identified case studies, we have extracted specific attributes that appear to impact the acceptance of AVs: gender-related preferences, perception of AV-related risks, social influence, and ease of use. In what follows we analyse potential correlations of these attributes with Hofstede's cultural dimensions.

Gender-Related Preferences: For Japan, the authors of [35] identified males are more interested than females in AVs, but for the UK and Germany [35], there is not enough evidence for gender-related preference. For USA [30], gender was identified as a factor influencing acceptance, contributing to 40.75% of the variance, with males exhibiting a higher likelihood of accepting the technology than females. If we compare Hofstede's scores for Japan and USA with the scores for

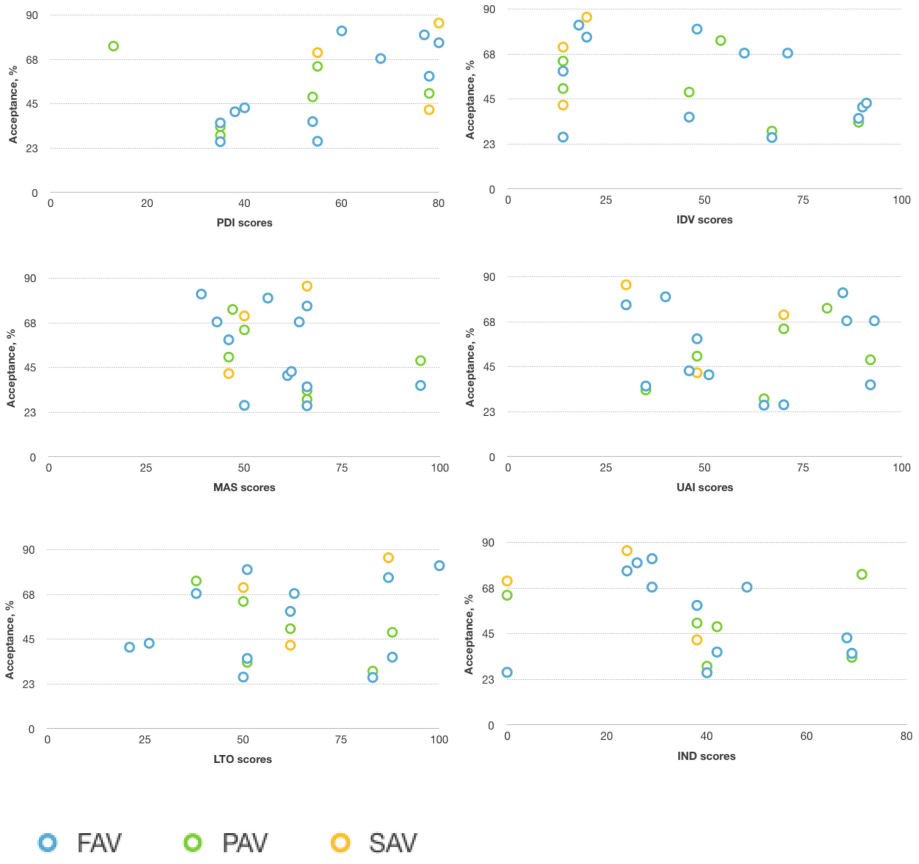


Fig. 1. Hofstede model scores vs. AV acceptance

Germany (see Table 1 for the values), there is no clear correlation between scores and the identified gender-related preferences for MAS dimension.

Risk Perception: In the case study [23], the authors compared the AV-related risk perceptions in Israel and Japan. The authors identified that in Japan, people are much more concerned about AV-related risks than in Israel. If autonomous vehicles can successfully build trust with consumers regarding their safety, consumers may perceive fewer risks associated with using autonomous vehicles. The authors posit that establishing trust is a crucial factor in comprehending the perceived risks linked to accepting autonomous vehicles. They believe that the level of comfort with using autonomous vehicles will increase if they can have confidence and trust in the safety of the AV technology. In software engineering, trust establishment is typically linked to the PDI dimensions, while risk perception is linked to UAI dimension. Therefore, a reasonable hypothesis might be to link the AV trust establishment with PDI, and AV risk perception is linked

with UAI. Figure 1 also illustrates this potential, where in both cases influence is positive, i.e. PDI^{+}, UAI^{+}.

A hypothesis from another study [16] based on Iran says that risk perception has a negative impact on deciding about acceptance of AVs. In that sense, a reasonable hypothesis might be that high UAI countries should have low AV acceptance rate. According to Hofstede's scores and acceptance rates, the majority of countries having high UAIs (i.e., Australia, Indonesia, Japan, Germany, and Pakistan) has low AV acceptance rates. At the same time, France, Poland, and South Korea have high UAI scores and exhibit high acceptance rates. Thus, there might be some correlation between AV risk perception and UAI scores. However, China has relatively low UAI (30), but according to [41] its SAV= 86% (SAV) and according to [31] its FAV=76%. India has medium-low UAI (40), but according to [31], its FAV = 80%. Thus, as mentioned above, it might be a promising research direction to investigate this potential correlation.

Social Influence: Peer pressure or social influence, which represents the opinions of others, can also be another big motivation for accepting AVs [38] positively or negatively. From another survey [16], social influence plays 29% positive role in accepting AVs for Iran. For USA [30], social influence holds the second most significant influence, contributing to 44.54% of user acceptance when it comes to autonomous vehicles (AVs). These are connected to social collaboration. People who see others drive AVs and decide to accept AVs depending on their peer's experience are relying on their peers' past experiences. Social influence might be potentially linked to the IDV dimension: for Collectivist societies it's typical that opinions and votes predetermined by in-group, where Individualist Societies is typical that a personal (individual) opinion is expected.

From Table 1 and Fig. 1, we can see that some countries with higher IDV (Australia, Poland, UK, USA, Germany) have lower acceptance rate, and lower IDV countries (Pakistan, Indonesia, China, South Korea, India) have higher acceptance rate. However, Pakistan has a very low IDV (value 0), but according to [41] its SAV=71%, and according to [29] its PAV = 64% and FAV=25%. If we consider IDV values for all other countries covered by the analysed studies, the results might indicate that IDV scores might negatively influence the acceptance.

Ease of Use: A hypothesis derived from a North American case study [27] suggests that the perceived ease of utilizing the technology positively impacts individuals' attitudes towards the adoption of AVs. In another case study [38] conducted in Malaysia, an examination was carried out to identify the factors influencing consumer acceptance of autonomous vehicles, revealing that ease of use is one of the positive influential factors. However, this influence has no attachment to culture.

Besides analysis of the above attributes, we also would like to discuss our observations regarding the LTO dimension. Based on Fig. 1, a hypothesis on correlation between AV acceptance and LTO scores might be reasonable: high LTO countries might have higher acceptance rates than low LTO countries. If we analyse Hofstede's scores from Table 1, and acceptance rates (see Table 2, we

can see that majority high LTO countries (South Korea, China, France, Indonesia, Pakistan, and India) have higher acceptance rates with some exceptions (Germany and Japan). So there might be some positive influence on LTO over accepting AVs, i.e. LTO^{+}, but there is no conclusive evidence as there are few exceptions.

4 Conclusions

In this paper, we analysed wrt. Hofstede cultural dimensions the combined results of 12 studies conducted in 14 countries: Australia, China, Canada, Germany, France, India, Indonesia, Israel, Japan, Pakistan, Poland, South Korea, UK, and USA. The aim of this work was to identify *potential* correlations for more focused in-depth analysis.

As result of our analysis, we identified the following potential impacts requiring further investigation:

- PDI^{+}, UAI^{+}, LTO^{+}: PDI/ UAI/ LTO scores might have positive impact on AV acceptance,
- IDV^{-}: IDV scores might have negative impact on AV acceptance,
- MAS^{0}, IND^{0}: MAS and IDV scores have no impact on AV acceptance.

We observed that even when the results of different case studies within the same culture/country differ, this difference is not as large as if we compare different countries, which highlights the correctness of the hypothesis that culture plays an important role in accepting AVs as well as potential for further work in this direction.

References

1. Abudayyeh, D., Almomani, M., Almomani, O., Alsoud, H., Alsalman, F.: Perceptions of autonomous vehicles: a case study of Jordan. World Electr. Veh. J. **14**(5), 133 (2023)
2. Aldakkhelallah, A., Shiwakoti, N., Dabic, T., Lu, J., Yii, W., Simic, M.: Public opinion survey on the development of an intelligent transport system: a case study in Saudi Arabia. In: AIP Conference Proceedings, vol. 2681 (2022)
3. Aldakkhelallah, A., Todorovic, M., Simic, M.: An investigation in autonomous vehicles acceptance. In: Zimmermann, A., Howlett, R., Jain, L.C. (edss) Human Centred Intelligent Systems. Springer, Singapore (2023). https://doi.org/10.1007/978-981-99-3424-9_8
4. Alsanoosy, T., Spichkova, M., Harland, J.: Cultural influence on requirements engineering activities: Australian practitioners' view. In: ISD2019 Proceedings (2019)
5. Alsanoosy, T., Spichkova, M., Harland, J.: A detailed analysis of the influence of Saudi Arabia culture on the requirement engineering process. In: Damiani, E., Spanoudakis, G., Maciaszek, L.A. (eds.) ENASE 2018. CCIS, vol. 1023, pp. 240–260. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-22559-9_11
6. Alsanoosy, T., Spichkova, M., Harland, J.: Cultural influence on requirements engineering activities: a systematic literature review and analysis. Requirements Eng. **25**, 339–362 (2020)

7. Alsanoosy, T., Spichkova, M., Harland, J.: Does our culture influence requirements engineering activities? Vietnamese practitioners' view. In: International Requirements Engineering Conference (RE), pp. 394–395. IEEE (2020)
8. Alsanoosy, T., Spichkova, M., Harland, J.: Exploratory analysis of cultural influences on requirements engineering activities based on stakeholders' profile. Procedia Comput. Sci. **176**, 3379–3388 (2020)
9. Alsanoosy, T., Spichkova, M., Harland, J.: Formal vs. case-study-based approaches for the identification of cultural influences in requirements engineering. In: ENASE, pp. 354–359 (2020)
10. Alsanoosy, T., Spichova, M., Harland, J.: A framework for identifying cultural influences on requirements engineering activities. In: PACIS Proceedings (2020)
11. Ansori, N., Sutarto, A.P., Widyanti, A., Trapsilawati, F., Rifai, A., Woodman, R.: The road to acceptance: a theory of planned behavior analysis of Indonesian public intentions towards autonomous vehicles. In: 14th International Conference on Information and Communication Technology and System (ICTS 2023) (2023)
12. Ayed, H., Vanderose, B., Habra, N.: Agile cultural challenges in Europe and Asia: insights from practitioners. In: ICSE, pp. 153–162. IEEE (2017)
13. Başargan, H.: Driver and pedestrian trust analysis on integration of autonomous vehicles to infrastructure of Turkey. Alphanumeric J. **7**, 25–36 (2019)
14. Borchers, G.: The software engineering impacts of cultural factors on multi-cultural software development teams. In: ICSE, pp. 540–547. IEEE (2003)
15. Dudziak, A., Stoma, M., Kuranc, A., Caban, J.: Assessment of social acceptance for autonomous vehicles in Southeastern Poland. Energies **14**(18), 5778 (2021)
16. Farzin, I., Mamdoohi, A.R., Ciari, F.: Autonomous vehicles acceptance: a perceived risk extension of unified theory of acceptance and use of technology and diffusion of innovation, evidence from Tehran, Iran. Int. J. Hum. Comput. Interact. **39**(13), 2663–2672 (2023)
17. Gerekos, D., Lampiris, N., Kopelias, P., Theofilatos, A., Karkanis, S.: Attitudes and perceptions of Greek drivers towards autonomous and connected vehicles. In: Hellenic Conference on Artificial Intelligence, pp. 1–3 (2022)
18. Haboucha, C.J., Ishaq, R., Shiftan, Y.: User preferences regarding autonomous vehicles. Transp. Res. Part C Emerg. Technol. **78**, 37–49 (2017)
19. Hall, E.T.: Beyond Culture. Anchor (1989)
20. Hofstede, G., Hofstede, G.J., Minkov, M.: Cultures and Organizations - Software of the Mind: Intercultural Cooperation and its Importance for Survival (3 ed.). McGraw-Hill (2010)
21. House, R.J., Hanges, P.J., Javidan, M., Dorfman, P.W., Gupta, V.: Culture, Leadership, and Organizations: The GLOBE Study of 62 Societies. SAGE (2004)
22. Howard, D., Dai, D.: Public perceptions of self-driving cars: the case of Berkeley, California. In: Transportation Research Board 93rd Annual Meeting, vol. 14, pp. 1–16. The National Academies of Sciences, Engineering, and Medicine (2014)
23. Khan, D., Fujiwara, A., Shiftan, Y., Chikaraishi, M., Tenenboim, E., Nguyen, T.A.H.: Risk perceptions and public acceptance of autonomous vehicles: a comparative study in Japan and Israel. Sustainability **14**(17), 10508 (2022)
24. Leidner, D.E., Kayworth, T.R.: A review of culture in information systems research: toward a theory of information technology culture conflict. MIS Q. **30**(2), 357–399 (2006)
25. Lytle, A., Brett, J., Barsness, Z., Tinsley, C., Janssens, M.: A paradigm for confirmatory cross-cultural research in organizational-behavior. Res. Organ. Behav. **17**, 167–214 (1995)

26. Meidute-Kavaliauskiene, I., Yıldız, B., Çiğdem, Ş, Činčikaitė, R.: Do people prefer cars that people don't drive? A survey study on autonomous vehicles. Energies **14**(16), 4795 (2021)
27. Müller, J.M.: Comparing technology acceptance for autonomous vehicles, battery electric vehicles, and car sharing – a study across Europe, China, and North America. Sustainability **11**(16), 4333 (2019)
28. Payre, W., Cestac, J., Delhomme, P.: Intention to use a fully automated car: attitudes and a priori acceptability. Transp. Res. F Traffic Psychol. Behav. **27**, 252–263 (2014)
29. Sanaullah, I., Hussain, A., Chaudhry, A., Case, K., Enoch, M.: Autonomous vehicles in developing countries: a case study on user's view point in Pakistan. In: Human Factors in Transportation, pp. 561–569. Springer (2017). https://doi.org/10.1007/978-3-319-41682-3_47
30. Saravanos, A., et al.: Investigating end-user acceptance of last-mile delivery by autonomous vehicles in the United States. In: Rauterberg, M., Fui-Hoon Nah, F., Siau, K., Krömker, H., Wei, J., Salvendy, G. (eds.) International Conference on Human-Computer Interaction, pp. 519–532. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-18158-0_37
31. Schoettle, B., Sivak, M.: Public Opinion about Self-driving Vehicles in China, India, Japan, the US, the UK, and Australia. University of Michigan, Tech. rep. (2014)
32. Sitinjak, C., et al.: Assessing public acceptance of autonomous vehicles for smart and sustainable public transportation in urban areas: a case study of Jakarta, Indonesia. Sustainability **15**(9), 7445 (2023)
33. Souris, C., Theofilatos, A., Yannis, G.: Attitudes of Greek drivers towards autonomous vehicles-a stated preference approach. Adv. Transp. Stud. Sect. B **48**, 105–116 (2019)
34. Spichkova, M., Alsanoosy, T., Harland, J.: Impact of organisational culture on the requirement engineering activities. In: International Requirements Engineering Conference (RE), pp. 426–427. IEEE (2021)
35. Taniguchi, A., Enoch, M., Theofilatos, A., Ieromonachou, P.: Understanding acceptance of autonomous vehicles in Japan, UK, and Germany. Urban Plann. Transp. Res. **10**(1), 514–535 (2022)
36. Thanasankit, T.: Requirements engineering - exploring the influence of power and Thai values. Eur. J. Inf. Syst. **11**(2), 128–141 (2002)
37. Thanasankit, T., Corbitt, B.: Understanding Thai culture and its impact on requirements engineering process management during information systems development. Asian Acad. Manage. J. **7**(1), 103–126 (2002)
38. Thayalan, N., Ariffin, S.K.: Examining factors influencing consumer acceptance towards autonomous vehicle in Malaysia. Int. J. Sustain. Strateg. Manag. **10**(1), 1–24 (2022)
39. Todorovic, M., Aldakkhelallah, A.A.A., Simic, M.: The adoption appraisal of autonomous vehicles. In: Zimmermann, A., Howlett, R.J., Jain, L.C. (eds.) Human Centred Intelligent Systems, pp. 125–135. Springer, Singapore (2022). https://doi.org/10.1007/978-981-19-3455-1_9
40. Trompenaars, F., Hampden-Turner, C.: Riding the Waves of Culture: Understanding Diversity in Global Business. Nicholas Brealey International (2011)
41. Wang, Z., Safdar, M., Zhong, S., Liu, J., Xiao, F.: Public preferences of shared autonomous vehicles in developing countries: a cross-national study of Pakistan and China. J. Adv. Transp. **2021**, 1–19 (2021)
42. Yun, Y., Oh, H., Myung, R.: Statistical modeling of cultural differences in adopting autonomous vehicles. Appl. Sci. **11**(19), 9030 (2021)

Author Index

A
Adam, Shuaibu Musa II-368
Agate, Vincenzo I-445
Agti, Nadia II-231
Alizai, Muhammad Hamad II-43
Amano, Tatsuya II-280
Araujo, Aleteia II-319
Arrotta, Luca I-375
Asghar, Farhan II-43

B
Bachiega Jr., João II-319
Bader, Lennart I-464
Bahillo, Alfonso I-526
Bai, Hongtao II-105
Banerjee, Abhik I-156, II-319
Banerjee, Rochishnu II-28
Barendregt, Rosaline II-473
Beigl, Michael II-15
Ben Fredj, Khalil I-201
Bettini, Claudio I-375
Bhatti, Naveed Anwar II-43
Bie, Yuhang II-485
Bodenhausen, Jörn I-333
Brunello, Andrea I-46
Bui, Van Vu II-368

C
Cai, Mengnan II-3
Carvalho, Leonardo R. II-319
Cenni, Daniele II-167
Chen, Hsiao-Yuan I-3
Chen, Niansheng I-297
Chen, Xin I-313
Chen, Zhi II-268
Cheng, Songlin I-297
Chibani, Abdelghani II-231
Chowdhury, Farida II-381
Christmas, Shaine I-175
Civitarese, Gabriele I-375
Costa, Breno II-319
Cui, Jiaxu II-105

D
De Paola, Alessandra I-445
Diallo, Alpha I-26
Díez, Luis Enrique I-526
Dolui, Koustabh I-97
Don, Jacob II-341
Du, Cuifeng I-489, II-268
Duan, Zhifei I-489

E
Edirisinghe, Sampath II-86
Eyobu, Odongo Steven I-526

F
Fan, Guangyu I-297
Fatima, Tehreem II-43
Fujii, Hisanori II-204

G
Gan, Qingqing I-429
Gani, Md Osman II-28
Garbinato, Benoît I-26
Georgakopoulos, Dimitrios I-119
Grafflage, Kolja I-333
Gu, Tao II-244
Guan, Quanlong I-489, II-268
Guo, Xiaolei II-511
Gupta, Neelima II-294

H
Han, Qi II-167, II-415
Hara, Takahiro II-219
Harland, James II-538
Hasan, Khalad II-381
Hasan, Mehedi II-428
Havinga, Paul I-201
Heijenk, Geert I-201
Heitmann, Martin II-473
Henze, Martin I-333
Hmissi, Fatma II-393
Hoo, Vichen II-268

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved

A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 549–552, 2024.
https://doi.org/10.1007/978-3-031-63992-0

Hu, Chunyu I-392
Huang, Guang-Li I-156
Huang, Xiujie I-489, II-268
Huang, Yanqiu I-201
Huang, Yiran II-15
Hughes, Danny I-97, II-368

I
Inan, Muhammad Sakib Khan I-119
Islam, A. B. M. Alim Al I-408
Islam, Md Fourkanul II-28
Islam, Nafisa I-408
Ivoghlian, Ameer I-136

J
Jabbari, Amir II-125
Jayaraman, Prem Prakash I-119, I-156, II-319
Jiang, Landu II-244
Jiao, Libo I-313
Jin, Xuhao II-192
Julien, Christine I-3
Junior, Asiimwe Paddy I-526
Jurdak, Raja II-125

K
Kamal, Uday I-408
Kanhere, Salil S. I-464, II-353
Karmakar, Sushanta II-63
Katsuma, Ryo II-204
Kazar, Okba II-231
King, Evan I-3
Krishna, Aneesh II-341
Kummerow, Arne II-503

L
Lee, Kevin I-175
Leung, Victor C. M. I-257
Li, Jinbao II-256
Li, Jing II-511
Li, Shanping II-445
Li, Wang I-313
Li, Zhefu I-489, II-268
Liao, Kewen I-119
Ling, Tianheng II-529
Liu, Changjiang I-489
Liu, Guozheng I-223
Liu, Hao II-179
Liu, Hong I-392
Liu, Meiling II-485
Liu, Mengyao II-368
Liu, Shaoshan I-505
Liu, Sheng I-239
Liu, Wenwen I-257
Liu, Xiaoguang I-257
Liu, Zhaoyi I-97
Liu, Zhen I-429
Lo Re, Giuseppe I-445
Lu, Kezhong II-244
Lunardi, Roben C. I-476
Luo, Cheng II-244
Lv, Shuanghuan II-268
Lyu, Lei I-392

M
Mackenzie, Brendan J. II-368
Madria, Sanjay II-294
Mahmud, Redowan II-341
Malik, Sidra II-125
Manchanda, Ashish I-156
Mazumder, Aritra II-381
Michiels, Sam I-97, II-368
Mistry, Avinaba I-68
Mistry, Sajib II-341
Montanari, Angelo I-46
Morshchavka, Sergii I-97
Mubassira, Taseen II-428
Muthu Kumara Swamy, Shneka II-415
Muzammel, Chowdhury Shahriar II-538

N
Namvar, Anahita II-353
Nguyen, Thomas II-28
Ning, Wangzhong I-313
Noyce, Hanna M. II-28

O
Olejniczak, Emily M. II-28
Ouni, Sofiane II-393

P
Pallamreddy, Akhil Reddy I-201
Paul, Souradyuti II-63
Peng, Bitao I-429
Pennekamp, Jan I-464
Phuong, Huynh Nguyen Bao II-368

Q
Qian, Chao II-529

R
Rabbi, Forhad II-381
Rademacher, Michael I-333
Rahaman, Masfiqur I-408
Rajashekar, Kolichala II-63
Ramachandran, Gowri II-125
Ranaweera, Chathurika II-86
Rao, Lei I-297
Rasheed, Raihan I-408
Ray, Suprio I-68
Raychoudhury, Vaskar II-28
Riedel, Till II-15
Rizk, Hamada II-280
Rosa, Nelson Souto II-368
Rötzel, Sebastian I-333

S
Sabri, Lyazid II-231
Saccomanno, Nicola I-46
Sakib, Md. Nazmul Hasan I-408
Salcic, Zoran I-136
Salim, Saiful Islam I-408
Schaffeld, Matthias II-520
Schiele, Gregor II-529
Schneider, Jean-Guy I-175
Seahra, Sanjeev I-68
Sharmin, Sadia II-428
Shen, Haifeng I-119
Shen, Xinling II-3
Shen, Zhonghui II-256
Shi, Huiling I-223
Siddiqui, Junaid Haroon II-43
Sidhanta, Subhajit II-63
Silveira, Rodrigo W. I-476
Smith, Roger O. II-28
Soha, Sabiha Tahsin II-381
Song, Xiaoyong I-297
Sorgatz, Christian I-333
Spichkova, Maria II-461, II-538
Spiß, Maria I-464
Su, Tongtong I-257
Sun, Bingyi II-105

T
Tan, Lizhuang I-223
Tang, Bo II-256
Tang, Chang I-223
Tang, Jie I-505
Tang, Ming Jian I-119
Thangarajan, Ashok Samraj I-97, II-368
Thapa, Chandra II-353
Thevaraj, Emildeon I-464
Torres-Sospedra, Joaquín I-46

V
Virga, Antonio I-445
Vogt, Thomas I-333

W
Wadhwa, Manisha II-294
Wang, Changji I-429
Wang, Gang I-257
Wang, Kevin I-Kai I-136, I-355
Wang, Nan II-256
Wang, Qing II-179
Wang, Ruoyu I-429
Wang, Yaoyidi I-297
Wang, Yixiao I-392
Wang, Zehua I-257
Wasson, Barbara II-473
Weerasinghe, Shakthi I-156
Wehrle, Klaus I-464
Weis, Torben II-503, II-520
Wu, Xiaofeng I-489, II-268

X
Xie, Rong II-145

Y
Yamada, Shota II-280
Yamaguchi, Hirozumi II-280
Yang, Bo II-445
Yang, Dingyu I-297
Yang, Jie II-179
Yang, Meihong I-223
Yao, Lina II-219
Ye, Hanting II-179
Ye, Xiaozhou I-355
You, Linlin I-239
Yu, Bo I-505
Yu, Zhaoyang I-257
Yuan, Lin I-392

Z
Zaslavsky, Arkady I-156
Zhang, Dian II-244
Zhang, Guanghui I-489
Zhang, Shuai I-281

Zhang, Shunliang II-511
Zhang, Tian I-489
Zhang, Wei I-223
Zhang, Xiangxie II-179
Zhang, Xichen I-68
Zhang, Yan II-145
Zhang, Yihong II-219
Zhang, Yonggang II-105
Zhang, Zhongmei I-281
Zhao, Haibin II-15
Zhao, Sinong I-257
Zheng, Sunjian I-505
Zhong, Yuansheng I-489
Zhou, Jiyun II-485
Zhou, Yexu II-15
Zhou, Yuren I-239
Zhou, Yuyu II-268
Zhu, Wenwu I-313
Zhuang, Xiaotian II-268
Zorzo, Avelino F. I-476