Reproducibility Score for Computational Artifacts

Rakan Alseghayer, Panos K. Chrysanthis, and Bruce R. Childers

Department of Computer Science, University of Pittsburgh

Abstract

Experiments for computationally-driven science are often difficult to reproduce due to under-specified parameters, limited metadata, and/or complex software. In computer science, Artifact Evaluation (AE) has emerged to incentivize authors to provide documentation, metadata, and packaging to make experiments more reproducible. AE often relies on third-party peer review to check the consistency of experimental artifacts with claims. In this paper, we take a preliminary step to examine whether machine learning can be used to construct models that predict the reproducibility of results. In addition, these models generate reproducibility scores as vectors that qualify and explain the reproducible results. These reproducibility scores can also facilitate the finding and reusability of the computational artifact that produced the results. Using data from manual peer review, we show that such models indeed hold promise for automatic artifact evaluation.

1 Introduction

Motivation With the growth of the computationally-driven science, it has become difficult to re-evaluate and verify findings and outcomes. A recent study reported that 70% of researchers could not faithfully reproduce other studies' results [5]. Another study showed that experimental computer science (CS) faces a similar challenge [3]. For computational evaluation, the issue is few researchers share the code and data artifacts necessary for re-evaluation, such as interview scripts, datasets, source code, and programming environment specification. Consequently, the scientific research process suffers from the lack of collaboration and reuse.

In computer science, researchers are acknowledging the challenge. Many ACM and IEEE computer science conferences have adopted *Artifact Evaluation* [6] (AE) over the last several years to incentivize better experimental practices, including the reuse of data and code artifacts. For example, ACM SIGMOD 2008 was the first database conference that adopted peer review and re-execution of the software and data artifacts behind claims [4]. Artifact Evaluation is a separate process conducted after the acceptance of the academic work. It is the authors choice whether to engage or not, and it is regulated by a separate committee. To this end, AE has a spectrum of outcomes: from *availability*, testing the execution of the code associated with a paper; to *repeatability*, testing the code associated with the paper against the data sets used by the authors; to *workability*, running different/more experiments with different or more parameters than shown in the respective papers; to *reusability*, the code can be extended and used in subsequent studies [7].

Significance There are many factors that contribute to this challenge of reproducibility. The factors vary from full algorithm specification and datasets used, to different environment specifications, to the analysis and reporting of the results. These clearly hinder the AE, which is currently manual, labor intensive, and costly. Our vision is to automatically identify papers that are likely

to be reproducible, and hence, pass AE. We aim to address the question: *Can simply analyzing the aspects characterizing an artifact lead to a prediction of its reproducibility and reusability?*

Specifically, there is a need for an automated solution that evaluates artifacts, and furthermore, gives clues on why certain work is reproducible, potentially reproducible, or non-reproducible (e.g., hardware requirements). Having interpretable clues on how a paper is reproducible not only accelerates AE and increases expeirmental confidence, but also guides researchers on what and/or how to reuse academic work. We envision these clues to be in the form of a *reproducibility score* (RS), which will 1) allay the manual effort of AE, 2) stimulate the AE of academic work that is encapsulated with experimental artifacts, and 3) aid the academic community in how to build upon and reuse existing work.

Opportunity Advances in machine learning (ML) vary from basic statistical techniques, such as Linear Regression, to more complicated and hand-crafted techniques, such as deep neural networks, which make classification and prediction tasks more efficient, accurate, and accessible to users with different backgrounds. Intuitively, one can see a potential in utilizing such techniques in the context of automating the process of AE. Thus, the first step of realizing our vision is an evaluation pipeline that utilizes ML to train model(s), which take a research article (e.g., journal articles, conference papers, posters, and reports) as an input and produce a reproducibility score.

Contributions In this paper, we present our reproducibility score pipeline. In particular:

- We describe our approach in building a pipeline that produces an RS vector (Section 2).
- We present a preliminary study on the feasibility of our proposed solution (Section 3).

2 Approach

Typically, an academic article documenting a finding consists of descriptions of the proposed solution, methodology, developed algorithms, and explanations of the experimental evaluation that include the parameters used, the analysis of the results presented in figures, tables, and appendices. In addition, such articles can contain other contextual information that are not related to the documented finding, such as the publication venue, the names of the authors, the year of publishing, etc. Having this information in hand and analyzing it can lead to some clues on whether the work is reproducible, potentially reproducible, or non-reproducible.

Having that in mind, we propose a pipelined solution (Fig. 1) that is based on training ML model(s) that take a set of inputs that correspond to a document, and produces a vector of quantifiable values that constitute a *reproducibility score* (RS) that expresses how reproducible or non-reproducible the work is. The details of this pipelined solution are discussed next.

Given a dataset of academic articles, namely in PDF format (and possibly Latex files, figures, and supplemental material), there are many reliable tools that extract text and figures from PDF files. Moreover, databases for academic articles are well indexed, which helps in references extraction from academic work in PDF format. Nonetheless, this remains a non-trivial task, as pre-processing such dataset and transforming it into computationally ready form is a significant step that affects the whole accuracy of the solution. For example, accurately extracting the textual data into plain-text format, extracting numerical information from an image (i.e., figure), and textual data from a table. Therefore, it is worth dedicating some effort to get accurate raw data out of academic documents to feed to the next step in our proposed pipeline (Fig. 1).

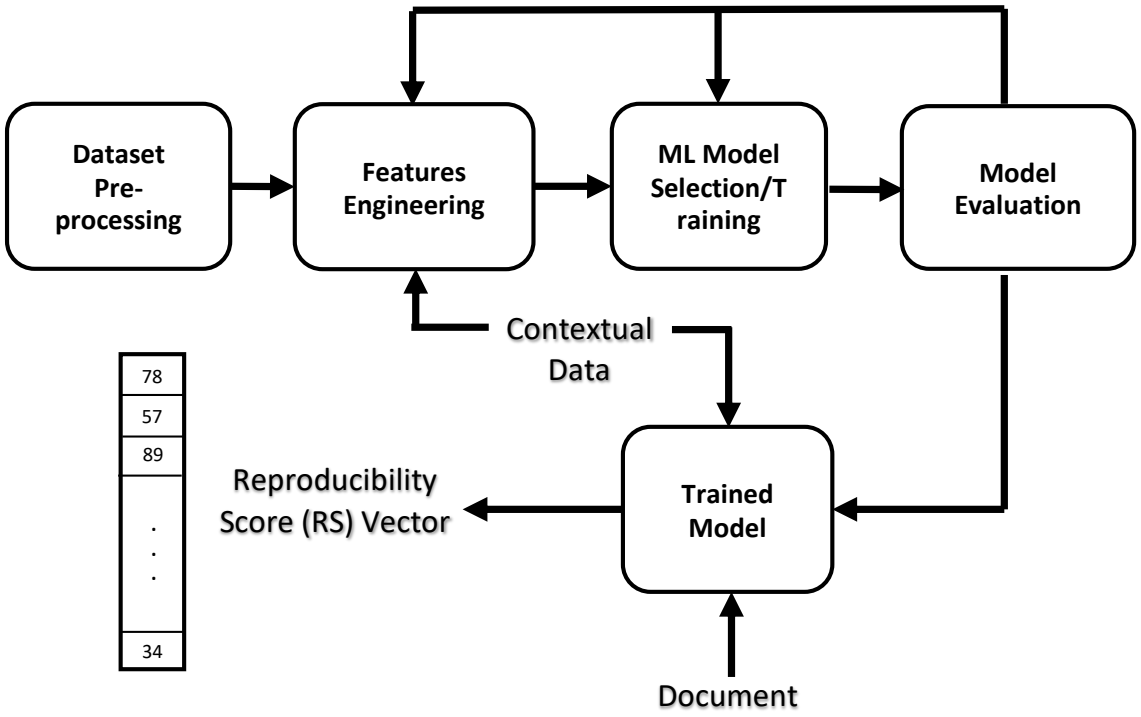

Figure 1: The proposed solution pipeline

The pre-procesing step results in arranging some information in computational form that can be fed to ML models. This step in ML terms is referred to as features engineering. It is crucial to be aware of the time and human analysis effort it takes to discuss the nature of the features needed to be extracted from the raw data. This depends significantly in the first place on observing and analyzing the human insights that lead to reproducibility of an academic work. Those features could be in the form they were originally extracted, after normalization, or computationally manipulated (such as dimensionality reduction, LDA, Matrix Factorization, etc. [1]). Moreover, it is reasonable that such features could be inferred from contextual information that are not withdrawn from the document, such as the authors h-index, the authors collaboration with other authors, etc. Extracting and hand crafting such features to feed to ML classifiers and predictors is a delicate task, because it affects the accuracy and the results of the models predictions. Thus, experimenting and researching this step is necessary to reach a reliable and feasible predictions.

After obtaining the proper ML features, the task of choosing and training a supervised ML classification and/or prediction model is of the essence. Also, since our proposed pipeline is not optimizing the task of classifying the academic work and producing the reproducibility score (RS) vector, we begin by picking efficient and powerful, yet simple, models that perform well with textual data, such as Linear Regression, Support Vector Machines (SVM), and Random Forest Classifier [1]. After establishing a baseline, the next step is to experiment with deep neural networks for textual representations, and convolutional neural networks for image and figures classification. Then, the

evaluation of the trained ML model results in either using it due to its high accuracy, or go back and enhance some aspects in the previous pipeline steps.

By following standard methods and applying standard metrics, we analyze and assess the model feasibility and validity. Those methods ensure that the trained model will behave to its best when it is being used with real world data (data never seen before). From this step, we can go back to the model selection step and try a different model, then retrain it to assess how it differs from the previous one. Therefore, it is common to go back to feature engineering step, and re-craft, include, and/or eliminate some features, and then retrain the model and/or select a different one. Eventually, we end at the model evaluation step in order to decide which model to choose and what features to consider. Some of the metrics that are used for model evaluation and assessment are: Root Mean Squared Error (RMS), Precision-Recall, and F1-Score measure [1]. Reaching a satisfactory trained model with proper accuracy, the output of that ML model is a *RS* vector that contains sub-scores for different aspects of a paper; each sub-score is a quantifiable numerical value that reflects an aspect of a reproducible work. The vector as a whole explains whether the work is reproducible or non-reproducible.

3 Case Study

As a first step to validate our approach, we experimented with open-source dataset about the repeatability in computer systems research [3]. The dataset of Collberg et al. consists of academic articles that vary in type (i.e., conference papers, journal papers, posters, and abstracts) and was compiled and curated as part of a study to evaluate the "Rebuildability" of the articles. The study tested whether it is possible to build the software code that was used in the experiments of the articles.

The study had four possible categories for each paper: does the code build in less than 30 minutes, does the code build in more than 30 minutes, the code builds according to the authors' claim, and the code does not build and the authors did not claim that.

For our proof of concept, we considered code that builds to indicate reproducibility, and code that does not build to indicate non-reproducibility. In essence, these are labels for training and validation. We excluded articles that did not have code from the original study.

The dataset we experimented with contains 388 document, where 201 were Non-Rebuildable and 187 were Rebuildable. As a first approximation, we extracted three features from the raw dataset: the code location (i.e. link found in the *article* or the code was found using a *web* search); the type of the article (i.e., conference papers, journal papers, posters, and abstracts); and the document page count. The first two features are categorical and the last one is numeric.

Next we trained three different classifiers using the 10-k cross-validation technique. These classifiers were Linear Regression, Support Vector Machine (SVM), and Random Forest Classifier. We tried many combinations of feature selection and used four metrics: *Accuracy*, *Precision*, *Recall*, and *F1-Score*.

	Accuracy	**Precision**	**Recall**	**F1-Score**
Linear Regression	0.82	0.81	0.77	0.76
SVM	0.83	0.81	0.77	0.76
Random Forest	0.83	0.81	0.77	0.76

Table 1: Average results from testing the trained models with 10-k cross-validation

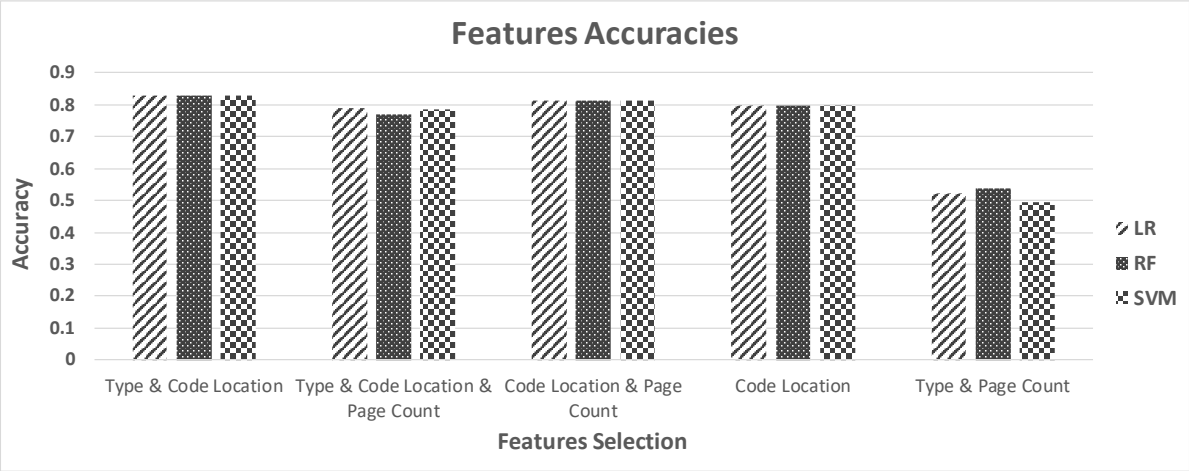

Figure 2: Average results from testing the trained models with 10-k cross-validation

The accuracy results are summarized in Fig.2. Since the feature combination that yields the best accuracy was the article type and the code location, we show the details of the results from the training models for that combination in Table 1. Furthermore, we show the distribution of the different features in the dataset in Fig. 3.

As shown in Fig. 3, it is clear that the most important feature is the code location, since it showed the largest bias between the Rebuildable work and the Non-Rebuildable. Furthermore, in Table 1, the accuracy (and other metrics) of the models are almost identical and relatively low. However, there is a clear potential that experimenting with more classifiers/predictors and hand crafting further features will lead to more accurate results.

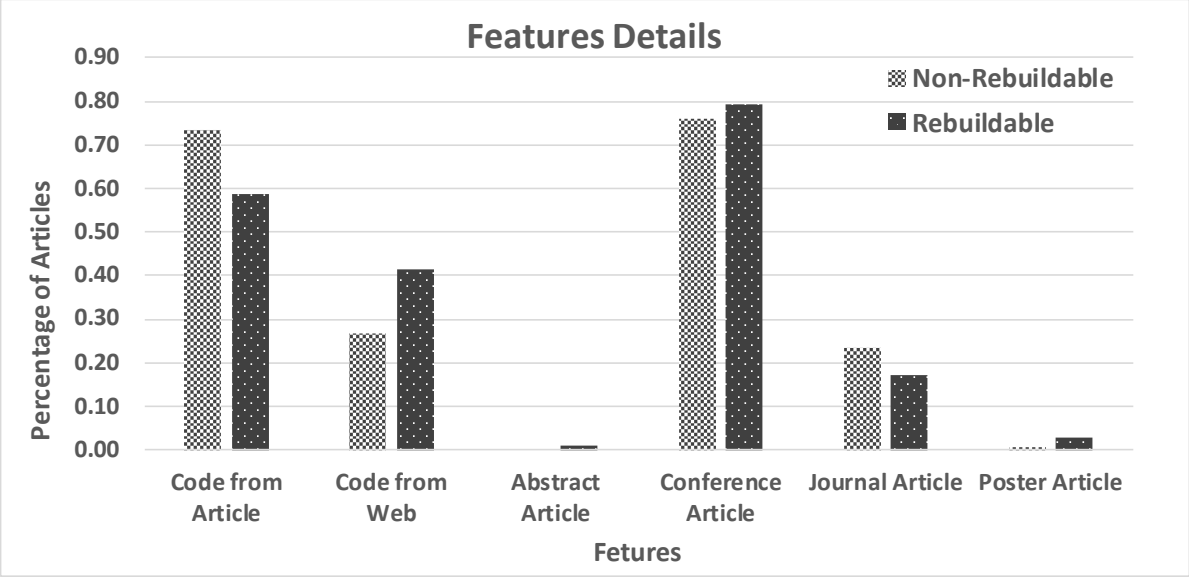

Figure 3: Features statistics in the dataset

4 Conclusions

In this paper, we examined whether an ML pipeline can be used to predict whether a scholarly paper is reproducible. Our approach is a first step toward generating a reproducibility score that captures the factors influencing whether a paper's results are likely to be reproducible. Preliminary investigation suggests this modeling approach is promising to accurately classify a paper as reproducible or non-reproducible.

In our preliminary investigation, the reproducibility score basically reflected the three features extracted from the raw dataset. Our next step is to enhance the reproducibility score with more features that will enhance the accuracy of the predictions and enable multi-level classification of the papers. Furthermore, we are planning on delving into more sophisticated ML models and techniques, such as deep learning and causal modeling. Additionally, we aim to enhance the pipeline to incorporate a recommending component that learns from historical data and generates suggestions on what contextual features—not incorporated in the document, such as researchers' reputations—could enhance the ML model(s) accuracy.

References

[1] Christopher M. Bishop. 2006. Pattern Recognition and Machine Learning (Informa- tion Science and Statistics). Springer-Verlag, Berlin, Heidelberg.

[2] B .R .Childers and P. K. Chrysanthis. 2018. Artifact Evaluation: FAD or RealNews?. In 2018 IEEE 34th International Conference on Data Engineering (ICDE). 16641665.

[3] Christian Collberg and Todd A. Proebsting. 2016. Repeatability in Computer Systems Research. Commun. ACM 59, 3 (Feb. 2016), 6269. https://doi.org/10. 1145/2812803

[4] I. Manolescu, L. Afanasiev, A. Arion, J. Dittrich, S. Manegold, N. Polyzotis, K. Schnaitter, P. Senellart, S. Zoupanos, and D. Shasha. 2008. The Repeatability Experiment of SIGMOD 2008. SIGMOD Rec. 37, 1 (March 2008), 3945.

[5] M. Baker, "1500 scientists lift the lid on reproducibility", Nature, vol. 533, pp. 452-454, May 2016.

[6] S. Krishnamurthi and J. Vitek. 2015. The real software crisis: repeatability as a core value. Commun. ACM 58, 3 (February 2015), 34-36.

[7] ACM. Artifact Review and Badging. Retrieved from https://www.acm.org/publications/policies/artifact-review-badging.