POSTER ABSTRACT

Profile and Context Filtering of Streaming Data for a Mobile Personal Assistant [*]

Alan Berfield
Dept. of Computer Science
University of Pittsburgh
Pittsburgh, PA 15260, USA
alandale@cs.pitt.edu

Jonathan Beaver
Dept. of Computer Science
University of Pittsburgh
Pittsburgh, PA 15260, USA
beaver@cs.pitt.edu

Panos K. Chrysanthis
Dept. of Computer Science
University of Pittsburgh
Pittsburgh, PA 15260, USA
panos@cs.pitt.edu

ABSTRACT

A key component in an ubiquitous computing environment is a personal assistant that can provide a link between devices and services and a human user. A large part of interactions in such an environment is expected to be using XML data. In this paper we present the use of profiles and context to obtain useful information from streaming documents. The user profile contains all the preferences and information on typical behavior for the user. This information can be entered by the user and learned over time. As the profile is never shared or transmitted, strict user privacy is maintained. Context information includes current day and time, and the user's location. We have implemented our ideas in an intelligent personal assistant, iPA, within the context of mobile commerce.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering, Selection process*; H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*User profiles and alert services*

General Terms

Algorithms, Design, Experimentation, Performance

Keywords

User profiles, mobile computing, machine learning, XML

1. INTRODUCTION

In this paper, we present an idea for an intelligent personal assistant, iPA, one that knows and works for a specific user. We make use of private profiles [5] and current user context to retrieve information for the user from streaming XML data in a mobile, ubiquitous computing environment. Our idea, however, does not require the user to ask for anything. Instead, the iPA learns about the user from what it observes of her actions and from anything she has explicitly told it. Using this information, the iPA will let the user know when something that would interest her is nearby, even when not asked. In this way, it becomes like a friend who knows about everything going on and knows what the user wants to hear about at the right time and place.

We have implemented our ideas in a prototype iPA within the context of mobile commerce. Documents in this environment are advertisements for stores and services, and include categories and properties of the subject. The iPA presents advertisements likely to be of interest to the user. Profile learning and updating is based on observations of user behavior. We have developed a simple technique for learning that works well despite the limited resources of a mobile platform.

2. PROFILES, DOCUMENTS, CONTEXT

The user profile contains all the preferences and information on typical behavior for the user. It is made up of XML tags that define a list of tasks (activities at specified times) with associated preferences. Preferences are weighted property/value pairs, where positive values indicate likes and negative values indicate dislikes. This information can be explicitly entered by the user or learned by the system over time by observing user activities. A simple example is shown below where the user typically eats between 11 and noon on weekdays, likes Indian, and dislikes Mexican.

```
<day vals=“M,T,W,H,F”>
  <range stime=“11:00” etime=“12:00”>
    <task impt=0.5 category=“Eating”>
      <pref name=“Ethnicity” val=“Indian” weight=6.0/>
      <pref name=“Ethnicity” val=“Mexican” weight=-5.0/>
```

Documents in our commerce environment are XML advertisements for stores and services. Similar to the profile, they have a list of task categories, such as “Eating” or “Shopping” and a list of associated property/value pairs.

Context information includes current day and time, user's location, other environmental properties as well as the presence of others. In our prototype environment, we define that any ads received are for stores or services that are “near” the user.

[*] Funded in part by NSF ITR Medium Award ANI-0325353.

SAC'04, March 14-17, 2004, Nicosia, Cyprus.
Copyright 2004 ACM 1-58113-812-1/03/04 ...$5.00.

3. SYSTEM OVERVIEW

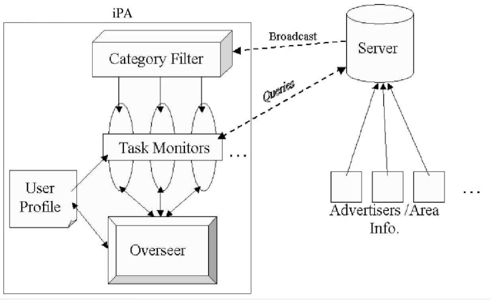

Figure 1: System Architecture and Interaction

We assume a WiFi network in a shopping mall or airport. The servers used in this environment are broadcast servers consisting of two main channels. The first channel is a push-based channel that continuously broadcasts information about items within a server's service area [1, 2]. The broadcast is organized by task categories and includes an index on them to enable power saving at the clients [6, 7, 3]. All servers broadcast on the same channel. The other channel is a pull-based channel where the user can submit queries to the server requesting more detailed information about an item in the broadcast.

The iPA has three main parts in addition to the profile: Category Filter, Task Monitors, and Overseer. The category filter parses the XML format ads received from the broadcast [4] and distributes them to task monitors based on their categories. The task monitors are more specific filters, each dedicated to a certain topic or task to be carried out by the user. They assign numerical rankings to each ad based on how it corresponds with the user profile. The overseer is the portion of the iPA dealing with human interaction and task management. Task management involves the creation and destruction of task monitors based on learned user behavior, resource limitations, or direct user requests. The overseer is also responsible for initiating profile updates for learning.

4. LEARNING

While many sophisticated learning techniques exist [8, 9], they typically require a great deal of time and resources. Given the very limited resources of our iPA hardware (a cell phone or PDA), such techniques can not be used. Instead we have created a very simple and fast method using the tags and preference weights in the profile.

Profile learning is initiated by the overseer module and is always the result of observed user behavior. In our prototype, the iPA "observes" the user whenever the user initiates a task or accepts or rejects a ranked advertisement (done by explicitly selecting "Accept" or "Reject").

Upon an acceptance or rejection, the profile task information to which the ad is related is modified as follows: If it does not yet exist, a tag is created for the task. Each property/value pair in the ad's task is created as a preference under the profile task, if such a preference does not yet exist. Those that are created are given an initial weight of either +1.0 or −1.0, depending on whether the ad is being accepted or rejected. Those that already exist, if any, have their weights modified, either adding or subtracting 1.0.

5. EVALUATION

To evaluate our system, we ran a series of preliminary experiments focusing on several tasks (activities at specific times of day) with between 30-50 advertisements related to each task. The user was then given the full list of all advertisements and asked to select their top 10 choices for each task. The system was then given the same set of tasks and advertisements and based on the previously learned profile information created a ranked list of ads for each task. The system rankings were then compared to the user's choices to create a rank similarity metric, RSM. For the metric, 1 would be a perfect match between the two sets while 0 would indicate an opposite ranking and complete mismatch. Random rankings would on average have a RSM of 0.5.

$$RSM = 1 - \frac{\sum_{i=1}^{10} |U_i - S_i|}{\sum_{j=1}^{10} (n + 1 - 2j)}$$

where U_i = User ranking for ad i,
S_i = System ranking for ad i,
n = Total number of ads

We performed our evaluation using 10 users. The average RSM value was 0.93. Our maximum achieved RSM was 1, while the minimum was 0.72.

6. CONCLUSIONS

We have described how a user profile can be used to keep track of user preferences and typical activities. Unlike many applications of profiles, our assistant maintains user privacy by never sharing this profile. The prototype implementation is written in Java and has a small footprint of 54KB, more than adequate for our target PDA platform. The described technique for learning is well suited to our purposes. It obviously requires very little resources, and despite its simplicity, it is able to capture the typical behavior and preferences of a user in our target domain of mobile commerce.

7. REFERENCES

[1] S. Acharya, M. Franklin, and S. Zdonik. Disseminating Updates on Broadcast Disks. *VLDB Conference*, (1996) 354–365.

[2] S. Acharya, M.Franklin, and S.Zdonik. Balancing Push and Pull for Data Broadcast. *ACM SIGMOD Conference*, (1997) 183–194.

[3] R. Agrawal and P. K. Chrysanthis. Efficient data dissemination to mobile clients in e-commerce applications. *Int'l Workshop on E-Commerce and Web Information Systems*, (2001) 58–65.

[4] M. Altinel and M. J. Franklin. Efficient Filtering of XML Documents for Selective Dissemination of Information. *VLDB Conference*, (2000) 53–64.

[5] M. Cherniack, M. J. Franklin, and S. Zdonik. Expressing User Profiles for Data Recharging. *IEEE Personal Communications*, (2001) 6-13.

[6] T. Imielinski, S. Viswanathan, and B. R. Badrinath. Energy efficient indexing on air. *ACM SIGMOD Conference*, (1994) 25–36.

[7] Q. Hu, W. C. Lee, and D. L. Lee. Power conservative multi-attribute queries on data broadcast. *ICDE Conference*, (2000) 157–166.

[8] T. Mitchell. *Machine Learning*, McGraw Hill, (1997).

[9] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*, Prentice-Hall, (1994).