ISSN 1746-7659, England, UK
Journal of Information and Computing Science
Vol. 3, No. 2, 2008, *pp*. 090-096

A Collaborative Approach for User Profile Capturing in Ubiquitous Environments

Zhiwen Yu [1,2+] and Xingshe Zhou [2]

[1] Academic Center for Computing and Media Studies, Kyoto University, Japan
[2] School of Computer Science, Northwestern Polytechnical University, P. R. China

(*Received December 22, 2006, accepted February 13, 2006*)

Abstract. User profile capturing plays an important role in service personalization, but is challenging to accomplish in ubiquitous computing environments. This paper proposes a collaborative approach to capture user profile. The approach is based on Master-Slave architecture, of which master side is a device with strong capabilities, such as workstations and PCs, and slave devices are low-cost, low-performance, mobile terminals. The complex profile processing, e.g. learning and merging are conducted in the master device. The slave devices are responsible for observing user behavior and uploading feedback information to the master device. The master device is designed to support multiple user profile capturing methods: explicit input, implicit learning, and dynamic merging. Our experimental results show the effectiveness of our approach and main algorithms.

Keywords: user profile capturing, personalization, ubiquitous computing, collaborative

1. Introduction

The rapid advances of communication technologies precipitate more and more embedded computing devices, such as laptops, PDAs, cell phones, etc., to be used in a wireless environment. This has led to a shift from traditional computer-centric to human-centric information access mode, which is known as *Ubiquitous Computing*. A major trend and requirement in today's information service is personalization. The capability to capture user profile is at the heart of a personalized service.

Capturing user profile for service personalization in ubiquitous computing is extremely challenging due to several factors. First, the poor human-machine interactivity of pervasive devices, especially the low-cost electronics such as phones and PDAs, causes user explicitly inputting his profile to be nearly impossible. Second, the limited computing power and storage of low-performance devices such as laptops make complex machine learning algorithm impractical. Finally, the mobility of user and devices causes insufficient feedback information that can be obtained for updating user preference. In pervasive computing environment, multiple devices are attached to a user and used at anytime anywhere. Each of the devices can gather user feedback information, but may be very fractional.

Fortunately, people nowadays usually own a device with strong capabilities, such as office workstation, personal computer, etc. In this paper, we therefore address the above challenges by proposing a collaborative centralized learning approach with the "strong capability device" been used in conjunction with the low-cost, low-performance, mobile user devices for user preference acquisition and update. Our approach collects fragments of user feedback information in different pervasive devices so as to build an abundant feedback repository, and then use a strong capability device to implicitly learn user preference from the feedback repository. It can also dynamically merge group user profile by considering interests of most of the members.

The rest of this paper is organized as follows. Section 2 discusses existing related research with this paper. Section 3 presents our approach. The system architecture, description model, user profile implicit learning, and user profile dynamic merging are described. The prototype implementation and performance evaluation are provided in Section 4. Finally, Section 5 concludes this paper and points out several issues for future research.

[+] Corresponding author. Tel.: +81 090-9891-1552
E-mail address: yu@ccm.media.kyoto-u.ac.jp.

2. Related Work

How to initiate and update user profile is a difficult problem. Much related work has been done in this area. Existing methods on user profile capturing can be divided into three categories based on different sources:

- Explicit input User explicitly inputs or modifies his profile manually, typically by clicking on items in a Graphical User Interface (GUI). It is simple and accurate, but cannot adapt to user's interests changing.
- Explicit feedback User feedbacks his evaluation to what system had recommended explicitly. It is clear, but need the system to provide interfaces.
- Implicit feedback System watches the user behavior and analyzes the user's viewing history (automatic user profile adaptation).

The explicit techniques (explicit input and explicit feedback) can reflect abrupt interest changes. But user rarely expresses his/her preferences to the system actively. Furthermore, the explicit techniques are 'static', and cannot adapt to changing user tastes. The implicit learning is capable of reflecting gradual interest changes, and can adapt to user's changing interests. But it is lacking of reflection to abrupt interest changes. Based on these reasons, many researchers have proposed hybrid or integrated approaches, which have better performance than single capturing method. TV3P [1] employs an implicit and explicit profiling scheme for personalized TV experiences by integrating explicit input/modification, explicit feedback, and implicit feedback. Philips TV recommender [2] encapsulates three user information streams: implicit viewing history, explicit preferences, and feedback information on specific shows into adaptive agents for program recommendation. P-EPG [3] adopts both implicit and explicit feedback for user modeling.

All above solutions are mainly proposed for TV or desktop environment, but not for ubiquitous computing environment. Panchanathan [4] concludes that user preferences in ubiquitous environment will not be acquired through traditional explicit interactions but implicit exchanges. Our collaborative approach for user profile capturing employs not only explicit input, but implicit learning and dynamic merging. It differs from and perhaps outperforms previous work in several aspects. First, it learns user preference by utilizing overall feedback information as opposed to other traditional methods that just use partial feedback information in one device. Second, our approach can relieve pervasive devices with limited resources from computation-consuming and storage-consuming learning tasks. Third, it can generate a common user profile when a group of users want to access personalized service together. Finally, it can make full use of existed user profile residing in other devices. For instance, a user has watched television program through TV set for a long period, so the user profile in the TV set may be comprehensive and correct. When the user wants to watch program through his PDA, he can export his/her preference information from the TV set, and no further learning is required.

3. Our Approach

3.1. Collaborative Architecture

Fig. 1 shows the schematic architecture of the collaborative capturing approach. There are four interfaces between master device and slave devices in the process. They are depicted as follows:

1) When the user wants to access personalized service through his/her pervasive mobile devices, the slave device first connects with his/her master device asking for the latest user profile.

2) Then the master device sends user profile to the slave device.

3) During the application running, the slave device observes user behavior for a specific service, and sends the feedback information to the master device.

4) When the user profile is updated, the master device sends the latest user profile to the slave device.

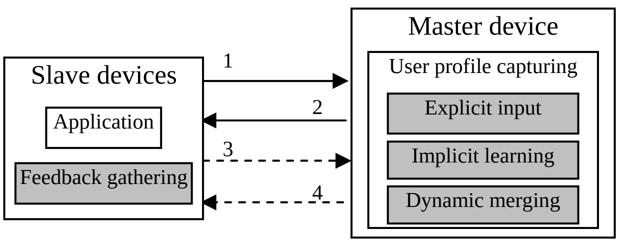

Fig. 1: System schematic architecture

The master device is designed to support multiple methods for user profile capturing: explicit input, implicit learning, and dynamic merging. Explicit input provides Graphic User Interface for the user to input interests when registration or modify preference after login. Implicit learning analyzes user viewing history or feedback information generally applying probability statistics, and then updates user preference. Dynamic merging deduces common interest of a group by merging individual user preferences into a common one.

The working spot of a device in pervasive computing environment may be at home, in office, on a train, or on road, etc. A diverse set of network technologies are used by different slave devices to communicate with master device. For instance, the master device may be a PC at home. The slave devices may include an interactive TV, PC in office, laptop, PDA, and cellular phone. To communicate with the master device, the interactive TV at home can use IEEE 802.11/Bluetooth, PC in office can use Ethernet, laptop can use ISDN, PDA and cellular phone can use GPRS.

3.2. Description Model

Three kinds of user-related information are involved in the system: user feedback for a specific service, user preference, and usage history. Since these kinds of information are exchanged from device to device, the description model should be publicly adopted (standard-oriented), flexible, and easily parsed and used by automated systems. Therefore, we adopt XML to represent these kinds of information.

The structure and semantics of these three kinds of information are described using the MPEG-7 Description Schemes. MPEG-7 offers such description schemes for modeling user preference and usage history [5]. The UserActionList DS in MPEG-7 can be used to describe user feedback, the FilteringAndSearchPreferences DS can be used to describe user preference information, and the UsageHistory DS can be used to describe usage history.

Fig. 2 illustrates a simple example of MPEG-7 based user feedback description. The feedback is generated in a service of watching TV. The name of user’s action type is “View”. The time that the action took place is “2004-09-07T15:05:00”. The total duration of the program is 45 minutes, and the user’s real watching time is 10 minutes. The content identifier is “01-mnf-100900” specified by the element of “ProgramIdentifier”, which identifies a media object uniquely.

```
…
<UserActionList>
 <ActionType><Name>View</Name></ActionType>
 <UserAction>
  <ActionTime>
    <ActionMediaTime>
      <MediaTimePoint>2004-09-07T15:05:00</MediaTimePoint>
     <MediaDuration unit=“Minute”>45</MediaDuration>
    </ActionMediaTime>
    <ActionGeneralTime>
      <Duration unit=“Minute”>10</Duration>
    </ActionGeneralTime>
  </ActionTime>
  <ProgramIdentifier organization=“MyIDOrg” type=“MyIDType”>01-
mnf-100900
  </ProgramIdentifier>
 </UserAction>
</UserActionList>
…
```

Fig. 2: Example of MPEG-7 based user feedback description

3.3. User Profile Implicit Learning

The implicit learning algorithm deduces and updates user preference by using a hybrid learning approach. It is designed to integrate user viewing history and user real-time feedback for preference learning. A time-driven learning algorithm is designed for update user profile through compiling statistical analysis on user viewing history, which is aggregated from all kinds of media playing devices, e.g. PC, television, PDA. It is built based on relevance feedback and Naïve Bayes classifier approach. The profile learning by utilizing user real-time feedback is an event-driven algorithm. When a real-time feedback (e.g. switching between channels or programs) happens, the learning algorithm will launch. The details of the user profile implicit learning approach were described in [6].

3.4. Dynamic User Profile Merging

Services in ubiquitous environment (e.g. watching TV) are often consumed by a group of users, e.g. a family, roommates in a student dormitory, friends in a party, etc. Therefore, the common interest of the group users is needed for the purpose of group-oriented recommendation. The user profile merging algorithm can deduce the group preference by merging individual user preferences into a common one [7]. The key technology of the strategy is based on total distance minimization. Whenever a subgroup of users wants to access personalized services together, the profile merging strategy can dynamically generates the common profile for them.

4. Implementation and Evaluation

4.1. Prototype Implementation

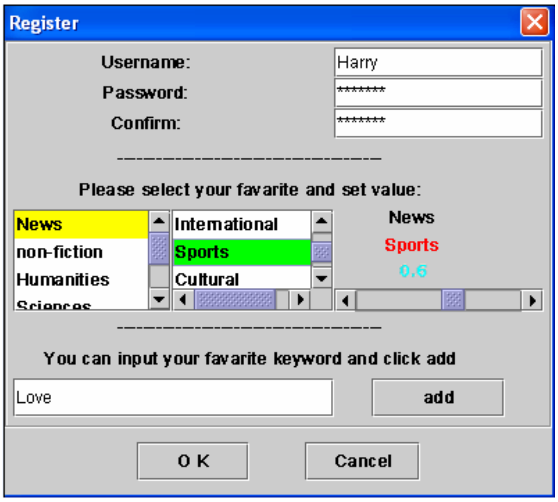

Fig .3: Explicit user preference imput/modification

We implemented the proposed collaborative user profile capturing approach, and built a ubiquitous personalized multimedia service to demonstrate it. The prototype system involved an IBM server (xSeries 235, 2.8 GHz Pentium 4 CPU, 512 MB RAM), a workstation (Dell Precision 340, 2.4 GHz Pentium 4 CPU, 512 MB RAM running Windows XP), and a laptop (Dell Inspiron 5000, PIII 600 MHz CPU, 128 MB RAM running Windows 2000). The IBM server was used as a multimedia service provider. Multimedia content and description metadata were stored in a file repository and an XML database Xindice [8] respectively. The workstation acted as the *master* device, and the laptop simulated the *slave* device. A hybrid user profile capturing scheme integrated with explicit input, implicit learning, and dynamic merging was deployed in the master device. A Pocket PC and a cell phone were simulated on the laptop, also working as slave devices. The prototype system was developed with Java. The application running on the slave device was developed following the Mobile Information Device Profile (MIDP) provided by J2ME. The J2ME [9] architecture comprises a variety of configurations, profiles, and optional packages for building Java applications that meet the requirements of a particular range of devices.

(a) (b)

Fig. 4: User interactions on the slave device

Fig. 3 is the user interest input/modification tool running on the master device. It provides the GUI interface for the users to input or modify their initial interests of TV program. In this example, the user "Harry" chooses the class "News", which consists of several subclasses, e.g. "International", "Sports", "Cultural", and etc. Here he selects "Sports", and finally sets the preference weight as 0.6. This means Harry prefers programs about news, especially sports news. Harry also adds a feature “Love” into the feature lexicon.

A scenario of user interactions on the slave device is illustrated in Fig. 4. A poster of the film “Gone With the Wind” was recommended to the user (see Fig. 4a), and three options were provided, which are “Record”, “View”, and “Delete”. The user chose to view the image. Then the image is presented to the user (see Fig. 4b).

4.2. Performance Evaluation

We evaluated the performance of the system from three aspects: implicit learning speed, implicit learning efficacy, and dynamic merging efficacy. Learning speed is crucial for mobile devices, because long-time computation will consume large battery power. Learning and merging efficacy reflects how much the learned or merged preference approaches to the user’s real interests, which directly influences the quality of personalization. The data sets used for experimentation and performance analysis were taken from the Internet Movie Database (IMDb) [10].

Experiment 1: We evaluated implicit learning speed by measuring learning response time on the master device and the slave device, i.e., the workstation and the laptop, respectively. We compared the time costs by varying the number of feedback records ranging from 50 to 250. Fig. 5 shows an example of feedback report extracted from the user’s viewing history. The evaluation result is shown in Fig. 6, from which we can see that the user preference learning is computationally intensive and the learning time is largely depending on the size of the feedback records. We also can observe that the learning response time on the laptop takes much longer than that on the workstation, and with the number of feedback records increasing from 50 to 250, the time difference is remarkable. When the size of feedback records reaches 250, the learning time on the laptop is very long (144s), which consumes much battery power and causes inconvenience to the user. But the learning time on the workstation is much less and acceptable. This result proved that the idea of running preference learning on a strong capability device other than pervasive mobile devices is appropriate.

Fig. 5: Feedback report (example)

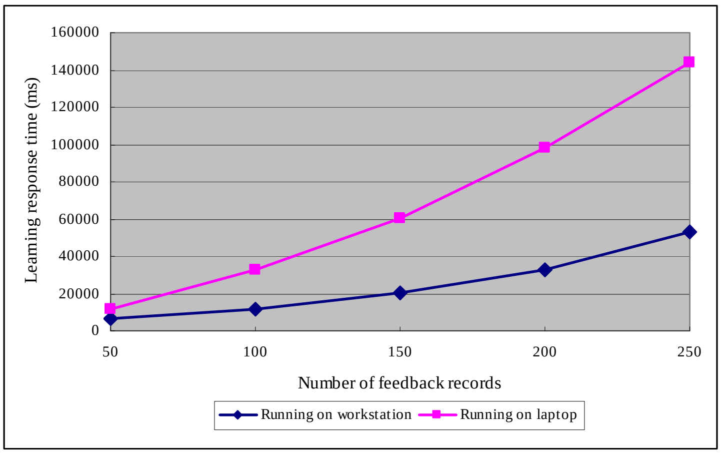

Fig. 6: Experimental result of learning speed

Experiment 2: We used *precision* and *recall* [11] to evaluate the efficacy of user profile learning. Given a time interval, let *interested* denote the multimedia content set which the user is interested in the interval and *recorded* denote the multimedia content set which the system has recorded in the same time interval, then *precision* and *recall* can be defined as follows:

$$precision = \frac{recorded \cap interested}{recorded}$$

$$recall = \frac{recorded \cap interested}{interested}$$

Since the two measures are often conflicting, we used Recall-Precision Graph, which integrates both precision and recall, to evaluate learning effectiveness. In the graph, each dot is a pair of recall-precision value.

We took 50 multimedia contents as training set, and 200 contents as testing set. The testing set was divided into 5 sessions with each session having 40 contents. We first let the user browse all of the movies' title and plot beforehand, and classify them into two classes: *interested* and *not-interested*. Then, the system chose and recorded movies for the user according to the user profile learned. After a session finished, we calculated the values of precision and recall. Fig. 7 shows the evaluation result in the form of Recall-Precision Graph. The result is encouraging with most precision values ranging around 0.8. In two sessions, through varying the relevance threshold, we got the largest precision 1 (where recall value is 0.54) and largest recall 1 (where precision value is 0.75) respectively. The experimental result proved that our system could keep track of user preference changing over time, and perform good filtering effectiveness, i.e. recording most of the movies that the user likes.

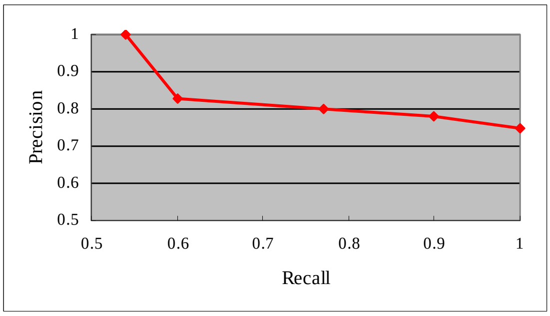

Fig. 7: Experimental result of learning efficacy

Experiment 3: In this experiment, we aimed to evaluate the efficacy of the profile merging by comparing feature weights merged and feature weights learned through the same learning approach for individual user profile. The evaluation method includes 4 steps:

1) Get a group of users;

2) Merge their profiles to generate a common user profile;

3) Show multimedia content to the group, and use the implicit learning approach to capture the group's preferences in terms of feature and its weight;

4) Compare the results (feature weights merged and feature weights learned) by using standard deviation (SD) indicated as follows. The smaller SD is, the better performance the strategy has.

$$SD = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(weightMerged_i - weightLearned_i)^2}$$

We ranked the features in the merged common user profile according to their weights. Then we took the top 10 features and compared their weights with corresponding weights in the learned profile. The results are shown in Table 1. The value of SD is 0.1998. We can observe that the weights of the top 7 features in the merged common user profile are very close to those in the learned user profile, while the weights of the last 3 features vary largely especially the feature "Satellite" and "James Bond". The reason is that in the learning process, there are not many programs involving the feature "Satellite" and "James Bond". We believe that if the period of learning process is sufficient, and the number of testing programs is as large as possible, the performance could be fine.

Feature name	Weight merged	Weight learned
Soccer	1.0000	0.9064
Diego Maradona	0.9329	0.9330
Stephen Chow	0.8893	0.9127
Zinedine Zidane	0.8832	0.8571
Comedy	0.8800	0.7069
Discovery	0.8796	0.8271
Drama	0.8665	0.6712
Satellite	0.8663	0.4200
Ge You	0.6534	0.5098
James Bond	0.6440	0.3299

Table 1. Results of Experiment 3

5. Conclusion

In this paper, we propose a collaborative approach of user profile capturing in ubiquitous computing environments. The basic thought of this approach is to utilize network transfer to remedy limitation of resources in pervasive terminals. However, network bandwidth is also a critical resource in ubiquitous computing. The approach should not just blindly apply a feedback transfer. It should also evaluate the cost/benefit tradeoff, which includes the expected reduction in computing power and storage, and also the expected communicational cost for the transfer. Possible solution is first to evaluate the computing cost as well as communicational cost, and then select an appropriate strategy (learn by itself or transfer the feedback to a strong capability device for centralized learning).

6. Acknowledgments

This work was supported by the National Science Foundation of China, and the Doctorate Foundation of Northwestern Polytechnical University of China.

7. References

[1] Yu Z. W. , Zhou X. S. TV3P: An Adaptive Assistant for Personalized TV. IEEE Transactions on Consumer Electronics 2004; 50(1): 393-399

[2] Kurapati K., Gutta S., Schaffer D., Martino J., Zimmerman J. A Multi-Agent TV Recommender. Proc. of the User Modeling 2001: Personalization in Future TV Workshop. Sonthofen, Germany, 2001.

[3] Ehrmantraut M., Harder T., Wittig H., Steinmetz R.. The Personal Electronic Program Guide - Towards the Pre-Selection of Individual TV Programs. Proc. of CIKM'96. Rockville, Maryland, USA, 1996; 243-250

[4] Panchanathan S. Ubiquitous Multimedia: Bridging the Digital Divide. IEEE Multimedia Magazine 2002; 9(1): 12-15.

[5] ISO/IEC JTC 1/SC 29/WG 11/N3966, Text of 15938-5 FCD Information Technology - Multimedia Content Description Interface - Part 5 Multimedia Description Schemes. Singapore, 2001.

[6] Yu Z. W., Zhou X. S., Yang Z. Y. A Hybrid Learning Approach for TV Program Personalization. Prof. of Intl Conf. on Knowledge-Based Intelligent Information & Engineering Systems 2004. Wellington, New Zealand. 2004; 630-636.

[7] Yu Z. W., Zhou X. S., HaoY. B., Gu J. H. TV Program Recommendation for Multiple Viewers Based on User Profile Merging. User Modeling and User-Adapted Interaction 2006; 16(1): 63-82.

[8] http://xml.apache.org/xindice/

[9] The Java 2 Platform, Micro Edition (J2ME). http://java.sun.com/j2me

[10] The Internet Movie Database. http://us.imdb.com

[11] van Rijsbergen C. J. Information Retrieval. Butterworths, 2nd edition, 1979.