Collaborative user modeling for enhanced content filtering in recommender systems

Heung-Nam Kim[1], Inay Ha[2], Kee-Sung Lee[2], Geun-Sik Jo[2], Abdulmotaleb El-Saddik[1]

[1] School of Information Technology and Engineering, University of Ottawa, 800 King Edward, Ottawa, Ontario, K1N 6N5, Canada, email: nami4596@gmail.com, abed@mcrlab.uottawa.ca

[2] School of Computer and Information Engineering, Inha University, 253 Younghyun-dong, Nam-gu, Incheon (402-751), Korea, email: {inay, lks}@eslab.inha.ac.kr, gsjo@inha.ac.kr

* Corresponding author. Heung-Nam Kim

Tel: 1-613-562-5800 ext. 6248, Fax: 1-613-562-5664, Email: nami4596@gmail.com

Abstract

Recommender systems, which have emerged in response to the problem of information overload, provide users with recommendations of content suited to their needs. To provide proper recommendations to users, personalized recommender systems require accurate user models of characteristics, preferences and needs. In this study, we propose a collaborative approach to user modeling for enhancing personalized recommendations to users. Our approach first discovers useful and meaningful user patterns, and then enriches the personal model with collaboration from other similar users. In order to evaluate the performance of our approach, we compare experimental results with those of a probabilistic learning model, a user model based on collaborative filtering approaches, and a vector space model. We present experimental results that show how our model performs better than existing alternatives.

Keywords. *Collaborative user modeling, Recommender system, Personalization, Content-based user model*

1. Introduction

The prevalence of Web 2.0 technologies and services enable end-users to be producers as well as consumers of content. Even on a daily basis, an enormous amount of textual content, such as online news, research papers, blog articles, and wikis are generated on the Web. It is getting more difficult to make automatic recommendations to a user related to his/her preferences, not only because of the huge amount of information but also because of the difficulty of automatically grasping his/her interests [6]. Recommender systems, which have emerged in response to the above challenges, provide users with recommendations of content suited to their needs. There are two widely used approaches among recommender systems, content-based filtering and collaborative filtering. The traditional task in collaborative filtering is to predict the utility of a certain item for the target user from the opinions of other similar users, and thereby make appropriate recommendations [17]. On the other hand, content-based filtering provides recommendations by comparing representations of content contained in an item to those of a user's interest content, ignoring opinions of other similar users [15]. Collaborative filtering has an advantage over content-based filtering in situations where it is hard to analyze the underlying content, e.g., music, videos, and photos. Because collaborative filtering process is only based on historical information about whether or not a given target user has previously preferred an item, analysis of the actual content, itself, is not necessarily required.

Nevertheless, collaborative filtering suffers from a fundamental problem, namely the *cold start problem*, which can be divided into *cold start items* and *cold start users* [21]. Several researchers have offered proposals dealing with the challenge of addressing this problem [9][15][18][21]. In a collaborative filtering-based recommender system, an item cannot be recommended until a large number of users have previously rated it. This is known as a cold start item. This problem applies to new items generated every few minutes and can be partially alleviated by content-based technology. In the case of domains such as textual documents, content-based filtering has proven to be effective in locating textual content relevant to a specific content information need [5][9]. However, content-based filtering also encounters limitations for a cold start user, similar to collaborative filtering. A cold start user describes a new user that joins a recommender system and has presented few opinions

(i.e., the user has insufficient preference history). With these situations, the system is generally unable to make high quality recommendations.

We address these issues by introducing a collaborative approach to user modeling for enhancing content filtering. Our goal is to build a robust user model that can be applied to personalized recommender systems. By capturing a user's content of interest, we can discover the preference patterns and terms existing in the user's content of interest. In addition to partially overcome the cold start user problem, we propose an enrichment method of the personal model in collaboration with other similar users.

This paper presents three specific contributions toward user modeling in recommender systems. First, we propose a new method of building a user model, allowing understanding and filtering of the user's interests. We then present a method of a collaborative enrichment of user interests in dealing with the cold start problem. Second, we propose how the individual model can be applied to personalized recommendations relevant to the user's needs. We incorporate collaborative characteristics into a content-based approach. Third, we provide detailed experimental evaluations with real datasets and investigate how collaborative user models work in terms of improving the recommendation performance.

The subsequent sections are organized as follows: Section 2 summarizes previous studies related to user modeling and personalized recommendations. In Section 3, we describe the notations and method to build the initial user model. We then describe a collaborative approach for modeling user interests and recommending content in Section 4. Next, Section 5 describes the implemented system and interface. In Section 6, we present the effectiveness of our approach in terms of its performance. Finally, conclusions are presented and future work is discussed in Section 7.

2. Related work

In personalized recommender systems, two main approaches have been developed: a content-based filtering approach and a collaborative filtering approach. Following the proposal of GroupLens [17], the first system to generate automated recommendations, collaborative filtering approaches have seen the widest use in a large number of information filtering problems relating to such things as movies,

books, music, online news, TV programs, and research papers. Despite success and popularity, collaborative filtering encounters several limitations, including the sparsity of the data, scalability, the cold start problem, and untrustworthy users. A number of researchers have addressed these problems using content-based filtering [3].

Content-based filtering methods, which are another well-known technique in recommender systems, have been developed using learning procedures. These procedures require training data to identify personal preferences (user model) from information objects and their content. *Webmate* tracked documents of interest to the user and exploited the vector space model using the TF-IDF (Term Frequency–Inverse Document Frequency) method [5]. Schwab et al. [22] explored the use of a classification approach to recommend articles relevant to the user profile, such as *NewsDude*. In *NewsDude*, two types of user interests are used: short-term and long-term interests. To avoid recommendations of very similar documents, a short-term profile is used. For the long-term interests of a user, the probabilities of a document are calculated using Naïve Bayes approach to classify a document as interesting or not. Instead of learning from users' explicit information, *PVA* [4] learned a user profile implicitly without user intervention. The user profile is represented as a keyword vector in the form of a hierarchical category structure. In *Newsjunkie* [10], a novelty-analysis algorithm is employed to present novel information for users by identifying the novelty of articles in the contexts of articles they previously reviewed. Lihua et al. [12] proposed a method of modeling multiple user interests by using a self-organizing map neural network with a changeable network structure. *SiteIF* [13] proposed using word sense-based document representation to build a model of the user's interests. A filtering procedure was employed to dynamically predict new documents based on a semantic network.

Collaborative and content-based filtering methods have unique advantages and disadvantages. Therefore, some studies combine these techniques in developing hybrid recommender systems [3]. Berkovsky et al. [1] presented a method of user modeling data integration for the purposes of a specific recommendation task, referred to as the mediation of a user model. By importing and integrating data collected from other recommender systems, four types of user model mediation are presented: cross-user, cross-item, cross-context, and cross-representation. In [2], the same authors

presented mediated user models that are transformed from collaborative filtering to content-based recommender systems. In [7], a content-collaborative hybrid recommender system is proposed that exploits WordNet-based user profiles to capture the semantics of user interests. Similar to our approach, the authors generated the neighborhood of a user through content-based methods. Melville et al. [15] followed a two-stage approach. First they applied a naive Bayesian classifier as content-based predictor to complete the rating matrix, and then they re-estimated ratings from this full rating matrix by collaborative filtering. *CinemaScreen* [18] reversed the stages. It executed content-based filtering on a result set generated through collaborative filtering.

Although the above-mentioned studies combine collaborative and content-based filtering approaches to exploit the benefits of each and lessen the disadvantages, our approach takes a different stance. Differing from earlier work, we automatically identify meaningful or useful patterns in building a user model. In addition, rather than utilizing explicit user feedback such as numeric ratings assigned to content, our aim is to build a robust user model implicitly inferred by the system from observing user behavior. Through the identification of useful patterns of a user in collaboration with other similar users, we discover content relevant to the user's needs.

3. Building a personal user model

The capability to learn users' preferences is at the heart of a personalized recommender system. In order to provide proper recommendations to users, personalized recommender systems require user models of characteristics, preferences, and needs. This information is typically referred to in the literature as a User Model (UM) [2]. Additionally, since every user can have different interests, feature selection for representing users' interests should be personalized and performed individually for each user [14]. In this section, we describe our approach to building a personal user model that is driven by the user's content of interest.

Before going into further detail, the notation and definitions required for understanding our approach are introduced. Let $C = \{c_1, c_2, \ldots, c_n\}$ be the set of all content, $T = \{t_1, t_2, \ldots, t_m\}$ be the set of all index terms, and $U = \{u_1, u_2, \ldots, u_l\}$ be the set of distinct users. The content c_j is a set of terms, each of which may appear in multiple content with different weights that quantify the importance of

the term for describing the content. In our study, a weight $w_{i,j}$ associated with a pair (t_i, c_j) (i.e., a term t_i of a content c_j) is computed by a fairly common type of TF-IDF weighting scheme [19]. To build a personal user model, potentially representative of user interests, we initially need some information given by the user, called user feedback. The most common ways to obtain the feedback is to use information given explicitly or to get information observed implicitly from the user's interaction [26]. Explicit feedback requires a user to evaluate content and indicate how relevant or interesting specific content is to him/her using like/dislike (a binary scale) or numerical ratings. Even though explicit feedback helps us to capture user preferences accurately, there is a serious drawback in that users do not tend to provide enough feedback. Users are generally not motivated to provide their feedback if they do not receive immediate benefits, even when they would profit in the long-term [26]. Therefore, in our study, we take implicit feedback into consideration in the sense that the system automatically infers the user's preferences from the user's behaviors [1][6][26]. In general, the preference indicator of implicit feedback can be represented as a form of a co-occurrence pair (u_h, c_j), where $u_h \in U$ is a user and $c_j \in C$ is specific content. The co-occurrence pair implies that user u_h viewed, clicked, collected, or bookmarked content c_j. While implicit feedback on specific content by a user does not necessarily mean that he/she likes the content, we assume that the co-occurrence pairs of the user are his/her interest content, implicitly.

3.1 Modeling user interests by text mining

Our approach to modeling user interests mainly consists of three steps: extracting terms, mining frequent patterns, and pruning patterns. In this section, we present the steps to initially build a personal user model in detail.

The first step in user modeling is the extraction of the terms from interest content that have been preprocessed by removing stop words and stemming words [16]. After extracting terms, each interest content c_j is represented as a vector of attribute-value pairs as follows:

$$c_j = \{(t_{1,j}, w_{1,j}), (t_{2,j}, w_{2,j}), ..., (t_{m,j}, w_{m,j})\} \quad (1)$$

where $t_{i,j}$ is the extracted term in c_j and $w_{i,j}$ is the weight of t_i in c_j. $w_{i,j}$ is computed by the static TF-IDF term-weighting scheme [19] and defined as follows:

$$w_{i,j} = \frac{f_{i,j}}{\max_l f_{l,j}} \times \log \frac{n}{n_i} \quad (2)$$

where $f_{i,j}$ is the frequency of occurrence of term t_i in content c_j, n is the total number of content pieces in the collections, and n_i is the number of content pieces in which term t_i occurs. The weight indicates the importance of a term in representing the content.

The second step is to mine frequent term patterns from the interest content of each user. Since every user has different interests, content used for the mining process must be selected individually for each user. Frequent patterns are a set of terms that appear frequently together in a set of a user's interest content. For example, if a set of terms {recommendation, collaborative, personalization, filtering} appear frequently together in a user's set of interest content, the set of those terms is a frequent pattern for the user. In the data mining research literature, frequent patterns are typically defined as patterns that occur at least as frequently as a predetermined minimum support (*min_sup*) [11]. In our study, we apply the mining process based on the following assumption: each transaction corresponds to an interest content of a user, items in a transaction are terms extracted from the content, and a transaction database corresponds to a user's set of interest content. Therefore, if the pattern support of pattern p_k (Definition 1) that is composed of at least l ($l \geq 2$) different terms, is above *min_sup*, i.e., $PS_u(p_k) > min\_sup$, then pattern p_k is referred to as a frequent term pattern. We denote a set of frequent term patterns for user u as $\boldsymbol{F}_u$.

Definition 1 (Pattern Support, PS) Let I_u be user u's set of interest content and pattern $p_k = \{t_1, t_2, \ldots, t_n\}$ be a set of terms such that $p_k \subseteq T$ and $n \geq 2$. A content piece c_j is said to contain pattern p_k if and only if $p_k \subseteq c_j$. *Pattern support* for pattern p_k in I_u, written as $PS_u(p_k)$, is the ratio of content in I_u that contains pattern p_k. That is, $PS_u(p_k)=f_u(p_k) / | I_u |$, where $f_u(p_k)$ indicates the occurrence frequency of pattern p_k in I_u.

Once the frequent patterns are mined, in the third step we remove the patterns containing unnecessary terms from the set of frequent term patterns. To this end, we define the importance of each term in representing a certain pattern, called the *pattern weight*. Formally, for a given pattern $p_k \in \boldsymbol{F}_u$, the *pattern weight* of p_k for user u, denoted as $PW_u(p_k)$, is computed by:

$$PW_u(p_k) = \frac{1}{|p_k|} \cdot \sum_{i \in p_k} \mu_{i,u} \quad (3)$$

where $\mu_{i,u}$ is the mean weight for term t_i in I_u and is computed as follows:

$$\mu_{i,u} = \frac{1}{|\mathrm{I}_u(i)|} \times \sum_{j \in I_u(i)} w_{i,j} \quad (4)$$

where $I_u(i)$ is the set of interest content for user u containing term t_i and $w_{i,j}$ is the weight of term t_i in content c_j. For any pattern p_k in $\boldsymbol{F}_u$, we determine the patterns for which the *pattern weight* is greater than the minimum pattern weight, *min_pw*, and model user preferences based on the identified patterns, collectively called a Personalized Term Pattern. In addition, terms that appear within personalized term patterns are called Personalized Terms.

Definition 2 (Personalized Term Pattern, PTP) A *personalized term pattern* is defined as a frequent term pattern for which the *pattern weight* is greater than the minimum pattern weight *min_pw*, i.e., $p_k \in F_u$ and $PW_u(p_k) > min\_pw$. A set of *personalized term patterns* for user u is denoted as PTP_u such that $PTP_u = \{(p_k, PS_u(p_k)) | PW_u(p_k) > min\_pw \wedge p_k \in F_u\}$.

Definition 3 (Personalized Term, PT) A *personalized term* is a term that occurs within *personalized term patterns*. The set of *personalized terms* for user u is denoted as PT_u. In addition, the vector for PT_u is represented by $\vec{PT_u} = (\mu_{1,u}, \mu_{2,u}, \ldots, \mu_{t,u})$, where t is the total number of personalized terms and $\mu_{i,u}$ is the mean weight for term t_i, which is computed by Equation (4).

The formal description of the model for user u, $\boldsymbol{M}_u$, is as follows: $\boldsymbol{M}_u = \langle PTP_u, PT_u \rangle$, where PTP_u models the interest patterns (Definition 2) and PT_u models the interest terms (Definition 3). And the model is stored in a prefix tree structure, which is inspired by a frequent-pattern tree (FP-tree) [11], to save memory space, explore relationships of terms, and retrieve PTPs having some PTs efficiently.

For example, if five personalized term patterns are found, as shown in Table 1, after mining the content of interest for user *u*, the tree structure of the model for user *u* is then constructed as follows. All PT_u are stored in the header table and sorted in order of descending frequency of terms since there are better chances that more prefix terms can be shared [11].

(Table 1 about here)

First, we create the root of the tree, labeled with "null". For the first term pattern, $\{t_1, t_2, t_3\}$ is inserted into the tree as a path from the root node, where t_2 is linked as the child of the root, t_1 is linked to t_2, and t_3 is linked to t_1. *PS* and *length* of the pattern ($PS(p_1)$=0.56, *length*=3) are then attached to the last node t_3. The nodes linked together in the path imply that the nodes (terms) contained in the pattern co-occur frequently in the user's interest content. For the second pattern, since its term pattern, $\{t_1, t_2, t_3, t_4\}$, shares a common prefix $\{t_2, t_1, t_3\}$ with the existing path for the first term pattern, a new node t_4 is created and linked as a child of node t_3. Thereafter, $PS(p_2)$ and $length(p_2)$ are attached to the last node t_4. The third, fourth, and fifth patterns are inserted in a manner similar to the first and second patterns. To facilitate tree traversal, a header table is built, in which each term points to its occurrence in the tree via a *node-link*. Nodes with the same *term-name* are linked in sequence via such *node-links*. Finally, the model for user *u* is constructed as shown in Fig. 1. Note that the built tree is a compact data structure for representing the whole interest patterns and terms of user *u* by sharing personalized terms in the personalized patterns.

(Fig. 1 about here)

4. Collaborative user modeling for content filtering

In this section we describe how to enrich the model for a specific user. The model $\boldsymbol{M}_u$ described in Section 3 is referred to the initial user model for user *u*. This model can be applied immediately to generate content recommendations. However, diverse patterns for user *u* cannot be discovered via the mining process in the case where the user has a small number of interest content. This is known as a

cold start user. With this situation, initial personalized term patterns may not be sufficient to represent user preferences, and thus our approach is generally unable to make high quality recommendations. In addition, when we only use the initial model for recommendations, it is hard to recommend to the user novel content of value aside from the usual set. For the above reasons, we propose an enrichment method of the user model via personalized term patterns of like-minded users.

4.1 Content-based neighborhood formation

The main goal of neighborhood formation is to identify a set of user neighbors, *k* nearest neighbors, which is defined as a group of users exhibiting interest terms similar to those of the target user. A typical collaborative filtering recommender system encounters serious limitations for finding a set of users, namely the sparsity problem [7][15]. The sparsity problem occurs when available data is insufficient to identify similar users (neighbors) due to the immense amount of content. In practice, even when users are very active, the result of rated content is only a small proportion of the total number of content. Accordingly, it is often the case that a pair of users has nothing in common, and hence the similarity cannot be computed. Even when the computation of similarity is possible, it may not be very reliable, because insufficient information is processed. To this end, in our study, we select the best neighbors by using the personalized terms, PT, of each user. In order to find *k* nearest neighbors, the cosine similarity, which quantifies the similarity of a pair of vectors according to their angle, is employed to measure the similarity values between a target user and every other user. As noted in Definition 3, the personalized terms of a pair of users, *u* and *v*, are represented as *t*-dimensional vectors, $\vec{PT}_u$ and $\vec{PT}_v$ respectively. Therefore, the similarity between a pair of users, *u* and *v* is measured by Equation (5).

$$sim(u,v) = \cos(\vec{PT}_u, \vec{PT}_v) = \frac{\sum_{k=1}^{t} \mu_{k,u} \times \mu_{k,v}}{\sqrt{\sum_{k=1}^{t} \mu_{k,u}^{\ 2}} \times \sqrt{\sum_{k=1}^{t} \mu_{k,v}^{\ 2}}} \quad (5)$$

The similarity score between a pair of users is in the range [0, 1] and the higher a user's score, the more similar he/she is to the target user. After computing the all-to-all similarity between users, we

define the set of nearest neighbors of each user u as an ordered list of k users $N(u) = \{v_1, v_2,\ldots, v_k\}$ such that $u \notin N(u)$, and $sim(u,v_1)$ is the maximum, $sim(u,v_2)$ is the next maximum etc. [20].

4.2 Collaborative enrichment of user interests

Once we have identified the set of the nearest neighbors for a certain user u, his/her initial model $\boldsymbol{M}_u = \langle PTP_u, PT_u \rangle$ is enriched from the neighbors. The basic idea of enriching the model of the user u starts from assuming that the user is likely to prefer similar patterns that have been discovered from the neighbors with similar tastes. The patterns discovered from more similar users contribute more to enriching the model of the target user. For example, if the pattern, such as {personalization, recommender}, frequently appears in interest content of a user, he/she might also be interested in the pattern, such as {personalization, recommender, collaborative, filtering}, that frequently appears in interest content of users similar to him/her. This enrichment process is particularly effective to some users who do not contain interest terms and patterns in their user model, such as the cold start users.

We elaborate on the general idea of the enrichment process in the following. Let $N(u) = \{v_1,v_2,\ldots,v_k\}$ be a sorted neighbor list of target user u, PTP_u be a set of personalized term patterns for user u, and PTP_v, $v \in N(u)$, be a set of personalized term patterns for neighbor user v of user u. Firstly, we choose neighbor user v in descending order of similarity between target user u and neighbors.

(Fig. 2 about here)

For each pattern p_i in PTP_u, *specific patterns* of p_i in PTP_v are identified. Given two patterns p_i and p_j, p_i is said to be a *general pattern* of p_j if and only if p_i is a subset of p_j, i.e., $p_i \subset p_j$. On the contrary, p_j is said to be a *specific pattern* of p_i. For example, let $p_1 = \{t_4, t_5\}$ be the personalized terms pattern for user u such that $p_1 \in PTP_u$, and $PTP_v = \{p_2, p_3, p_4, p_5\}$ be the set of PTPs for user v such that $p_2 = \{t_2, t_4, t_5\}$, $p_3 = \{t_4, t_5, t_8\}$, $p_4 = \{t_2, t_4, t_5, t_7\}$, and $p_5 = \{t_7, t_8\}$ as shown in Fig. 2.

Since pattern p_2, p_3, and p_4 contain the entire terms of pattern p_1, they are said to be a specific pattern. Several specific patterns that occur in the PTPs of neighbor v, PTP_v, may be found. For

efficient enrichment, we only consider specific patterns which have higher pattern support than that of the general pattern. Assume that the pattern support for p_1, p_2, p_3, and p_4 is 0.41, 0.5, 0.47, and 0.35, respectively (i.e., $PS_u(p_1)=0.41$, $PS_v(p_2)=0.5$, $PS_v(p_3)=0.47$, and $PS_v(p_4)=0.35$). In this case, only pattern p_2 and p_3 is used for enriching the model of user u if they are not PTPs for user u, as can be seen in Fig. 3. Patterns such as p_2 and p_3 are called *Collaborative Term Patterns* (CTPs) for target user u. An enriched model for user u by neighbor user v is built, as shown in Fig. 4.

(Fig. 3 about here)

(Fig. 4 about here)

Finally, a set of collaborative patterns is identified from *k nearest neighbors*, with respect to target user u. Note that the collaborative term pattern for the target user is not allowed to be redundant. That is, if the same patterns that were previously enriched by neighbor v are also discovered from another neighbor h such that $sim(u,v) \geq sim(u,h)$, for $v \neq h$, those patterns are pruned.

The enriched model for user u is defined as a triple $\boldsymbol{M}^+_u = \langle PTP_u, CTP_u, PT^+_u \rangle$ where PTP_u is the set of personalized term patterns for user u, CTP_u is the set of collaborative term patterns for user u, and PT^+_u is the set of interest terms that occur within either the personalized patterns or the collaborative patterns, respectively. In the enriched model $\boldsymbol{M}^+_u$, PTP_u models the interest patterns of user u whereas CTP_u models the enriched interest patterns by the neighbors of user u.

Definition 4 (Collaborative Term Pattern, CTP) Let p_i be a personalized term pattern for target user u, $p_i \in PTP_u$, and p_j be a personalized term pattern for neighbor v such that $p_j \in PTP_v$, and $v \in N(u)$. We define the set of *collaborative term patterns* for user u, denoted as CTP_u, as the set of neighbor patterns p_j such that $p_i \subset p_j$, $p_j \notin PTP_u$, and $PS_u(p_i) \leq PS_v(p_j)$.

4.3 Personalized content recommendation

After the model is enriched, we are ready to provide recommendations for new content that a user has not previously read. Based on the enriched model for each user, we recommend to the user the *top-N* ranked content that he/she might be interested in reading. To this end, the most important task in

personalized recommendation is to generate a prediction, that is, speculation about how much a certain user would prefer unseen content. In our study, we consider matched patterns, that is, how many interest patterns in a user model are contained in the new content. Formally, the numeric score of the target user u for the content c_n, denoted as $P_{u,n}$, is obtained as follows:

$$P_{u,n} = \frac{\sum_{p_k \in (PTP_u \cup CTP_u)} B_n^{p_k}}{N_u} \cdot \frac{\sum_{p_k \in (PTP_u \cup CTP_u)} | p_k | \times \omega_u^{p_k} \times B_n^{p_k}}{\sum_{P_k \in (PTP_u \cup CTP_u)} \omega_u^{p_k}} \quad (6)$$

where N_u is the total number of patterns in both PTP_u and CTP_u, and $B_n^{p_k}$ is binary variable for determining whether or not pattern p_k occurs in content c_n. That is, $B_n^{p_k}$ is 1 if pattern p_k appears in content c_n and 0 otherwise, and $\omega_u^{p_k}$ represents the weighted pattern support of p_k for user u, which is given by:

$$\omega_u^{p_k} = \begin{cases} PS_u(p_k) & \text{if } \mathrm{p_k} \in \mathrm{PTP_u} \\ PS_v(p_k) \times sim(u,v) & \text{if } \mathrm{p_k} \in \mathrm{CTP_u}, \mathrm{p_k} \in \mathrm{PTP_v} \end{cases} \quad (7)$$

The main concept of prediction dictates that interest patterns in the model of the target user are a good estimate of the preference for the selected content. The more the content contains the patterns in the model, the higher rank the content obtains. This scheme can also make recommendations for new content added regularly to the system, known as the *new item problem* in collaborative filtering [24], as well as support serendipitous recommendations [25]. Recommender systems relying exclusively on a user's interest content can only recommend content highly related to that which the user has previously selected. It is hard to recommend novel content that are different from anything the user has previously read before. This is known as the problem with overspecialization [18][24]. In our approach, by utilizing the enriched patterns from neighbors with similar tastes, we can make content to be a higher rank in the recommended set that the content contains the collaborative (enriched) patterns valuable to the target user, even though the patterns are not directly discovered from the user's interest content.

Once the content predictions about the target user, which the user has not previously read, are computed, the content are sorted in order of descending predicted value $P_{u,n}$. Finally, the set of N

ordered content elements with the highest values are identified for user u. This is the set of content recommended to user u (*top-N* recommendation).

Definition 5 (Top-*N* recommendation) Let C be the set of all content, X_u be the content list that user u has previously collected or added to his preference list (interest content), and Y_u be the content list not previously read by user u, $Y_u = C - X_u$ and $X_u \cap Y_u = \varnothing$. Given a pair of content elements c_i and c_j, $c_i \in Y_u$ and $c_j \in Y_u$, content c_i will be of more interest to user u than content c_j if and only if the prediction score $P_{u,i}$ of the target user u for the content c_i is higher than that of content c_j, $P_{u,i} > P_{u,j}$. *Top-N* recommendations for user u identifies an ordered set of N content, $TopN_u$, that will be of interest to user u such that $|TopN_u| \leq N$, $TopN_u \cap X_u = \varnothing$, and $TopN_u \subseteq Y_u$.

5. System implementation

Based on the requirements defined in Section 3 and 4, we developed a prototype system to support personalized content recommendations, named PRCUM (Personalized Recommendations via Collaborative User Model). The PRCUM system is divided into four main types of tasks: (a) Observing relevance feedback of a given user, (b) Modeling user interests from observed content, (c) Enriching user interests from nearest neighbors, and (d) Generating content recommendations for a given user. An overall system process for personalized content recommendations is shown in Fig. 5.

(Fig. 5 about here)

PRCUM first requires the user to sign in with his/her username and password, and then it allows the user to add content to a preference list and monitors the user's browsing inside the system. Because PRCUM cannot make recommendations to the user before building the individual model, it delays recommendations until the model is of a sufficient size and has been successfully built. The user can adjust the desired model parameters, such as the minimum support (*min_sup*), the minimum pattern weight (*min_pw*) and the number of nearest neighbors (k). Once the model has been built, PRCUM allows the user to enter his/her personalized pages and proposes to him/her a list of

recommended content. The GUI of PRCUM is implemented using C# and the server side is implemented using MySQL 5.0 and PHP 5.2 in an Apache 2.2 environment.

(Fig. 6 about here)

The GUI mainly consists of four frames: a menu frame, a favorite frame, a recommendation frame and a main frame. By interacting with the menu frame, users can choose the functions of PRCUM rendered by the main frame. As one of the principal functions in PRCUM, the recommendation frame provides a list of recommended content, a list of nearest neighbors, and recently added interest content. And the favorite frame is used for jumping to content in favorites previously registered in PRCUM. Users can maximize (display) or minimize (hide) the recommendation frame and the favorite frame according to their preference. Fig. 6 shows a snapshot of the user interface for the PRCUM system.

6. Experimental evaluation

In this section, we empirically evaluate the proposed approach and compare its performance against that of the benchmark algorithms. All experiments were performed on a *Dual Xeon* 3.0 GHz, 2.5GB RAM computer running the MS-Window 2003 server.

6.1 Datasets

We use two test datasets for our comparative experiments. The first dataset is taken from *NSF* (National Science Foundation) research award abstracts [16]. The original dataset is too large to be used in practice and thus we selected award abstracts with topics highly related to computer science. The selected dataset contains 974 unique abstracts (i.e., content) and 9,823 unique terms were obtained from the abstracts. In addition, we collected 9,845 preference histories (i.e., interest content of users) from 78 users. We refer to this dataset as *NSF*.

The second dataset comes from *MovieLens*, which is a web-based research recommendation system (www.movielens.org). The original dataset does not contain any information about movie content, and thus we extracted the textual descriptions (i.e., genres, keywords, summary) for each movie from

the IMDb database (www.imdb.com). Though the dataset contains numerical ratings, we ignored these and binarized them as follows: if a certain user's movie rating is larger than his/her average rating we set the rating to 1 (i.e., the interest movie of the user), or 0 otherwise. Thereafter, we removed users who had less than 20 ratings. The binarized dataset consists of 50,318 ratings on 1,682 movies from 658 users. We refer to this dataset as *MLens*. Table 1 briefly describes our datasets.

(Table 2 about here)

6.2 Evaluation design and metrics

To evaluate the performance of the recommendations, we randomly divided the dataset into *a training set* and *a test set*. The users' interest items were split into *a test set* with 10 items per user (i.e., 780 items for *NSF* and 6,580 items for *MLens*) and *a training set* with the remaining content (i.e., 9,065 items for *NSF* and 43,738 items for *MLens*) that was to used to learn and build a model of each user.

In order to evaluate the performance of our approach, we implemented the following: i) a user-based collaborative filtering method *UCF* [20], ii) an item-based collaborative filtering method, which employs cosine-based similarity *ICF* [8], iii) a probabilistic learning algorithm termed *NB* that applies the multinomial event model of a *naïve Bayes assumption* [14], and iv) a TF-IDF vector-based algorithm *VT* [5]. For the content recommendation process, in the case of *NB*, content were ranked using the calculated probability values, whereas they were ranked using the calculated cosine similarity for *VT*. For *UCF* and *ICF*, the proximity between users or items was measured by cosine-based similarity and items were ranked using the weighted sum using the similarity as the weight. Our *top-N* recommendation strategy (M^+) was then compared with the benchmark algorithms. We adopted two evaluation measures that are defined as follows:

Hit Rate (HR) In the context of *top*-N recommendations, the *hit-rate*, a measure of how often a list of recommendations contains items that the user is actually interested in, was used for the evaluation metric [8]. The *hit-rate* for user *u* is defined as:

$$HR(u) = \frac{\left| Test_u \cap TopN_u \right|}{\left| Test_u \right|} \tag{8}$$

where $Test_u$ is the item list of user u in the test data and $TopN_u$ is the *top-N* recommended item list for user u. Finally, the overall HR of *top-N* recommendation for all users is computed by averaging the personal $HR(u)$ in the test data.

Reciprocal Hit Rank (RHR) One limitation of the *hit-rate* measure is that it treats all hits equally regardless of the ranking of recommended content. In other words, a content item that is recommended with top ranking is treated equally with an item that is recommended with Nth ranking. To address this limitation, we adopted *the reciprocal hit-rank* metric described in [8]. The *reciprocal hit-rank* for user u is defined as:

$$RHR(u) = \sum_{i_n \in (Test_u \cap TopN_u)} \frac{1}{rank(i_n)} \tag{9}$$

where $rank(i_n)$ refers to the recommended ranking of item i_n within the *hit set* of user u. That is, hit content that appear earlier in the *top-N* list are given more weight than later ones. Finally, the overall RHR for all users is computed by averaging the personal $RHR(u)$ in the test data. The higher the RHR, the more accurately the algorithm recommends items.

6.3 Experimental results

In this section, we present detailed experimental results. The performance evaluation is divided into three dimensions. The effect of the neighbor size on the performance of model enrichment is first evaluated, and then the effectiveness of model enrichment is evaluated in comparison with the initial user model. Finally, the accuracy of content recommendations is evaluated in comparison with the benchmark methods. In the experiments, *min_sup* and *min_pw* was set to 0.1 (10%) and 0.5, respectively.

6.3.1 Experiments with neighborhood size

The following experiment investigates the effect of the enriched model through the neighborhood. And the number of recommended items N was set to 10 for each user in the test set. As noted in a number of previous studies, the size of the neighborhood influences the recommendation quality of

neighborhood-based algorithms. Therefore, different numbers of user neighbors were used for model enrichment: 10, 20, 30, 40, 50, and 60.

Table 3 summarizes the results of RHR and HR for the *NSF* dataset. With respect to HR, we observe that HR tends to improve slightly as the neighborhood size increases from 10 to 20; beyond this point, any further increase of the model size did not affect the performance. Interestingly, RHR was poorer for a neighborhood size of 30, 40, 50, and 60 than for a size of 20.

(Table 3 about here)

(Table 4 about here)

We further examined the performance of the *MLens* dataset. Similar results to *NSF* were obtained for *MLens*, as can be seen in Table 4. For example, when the neighborhood size is 30, this provides a reasonably good performance for both HR and RHR.

These results were affected by the fact that a neighborhood with a small size provides enough collaborative term patterns for each user. Recall that patterns are selected for enriching collaborative term patterns according to the nearest-order of neighbors, and thus redundant patterns generated by farthest neighbors are pruned. Another reason might be that we were only looking for a small number of recommended content (i.e., $N=10$). That is, once the number of nearest neighbors is relatively large, the rank of recommended content for each user is barely changed by any further increases in the number of nearest neighbors. In practice, recommender systems make a trade-off between recommendation accuracy and real-time performance efficiency by pre-selecting a number of nearest neighbors. In consideration of both accuracy and computation cost, we selected 20 and 30 as the neighborhood size for *NSF* and *MLens* model enrichment, respectively, in subsequent experiments.

6.3.2 Effect of model enrichment

This section investigates the effect of the enriched model M^+ of each user in more detail, by comparing the results obtained by the initial model M of each user. We performed an experiment with N values of 10, 20, and 30 and examined the average number of collaborative term patterns of users.

In the case of *NSF*, we found that 192 patterns had been enriched for each user, whereas the average number was 234 for *MLens*.

(Fig. 7 about here)

Fig. 7 presents the results of the experiment. The results demonstrate that the enriched model provides considerably improved HR values on all occasions, compared to the initial model. For example, the enriched model M^+ achieves 8.7% and 11.4% average improvement for *NSF* and *MLens*, respectively, in terms of HR, compared to the initial model *M*. Similar conclusions are implied by the RHR results as well. More importantly, we found that the enriched model outperforms the initial model in all cases that the number of recommended content is small. When *N* is 10, the enriched model obtains an RHR value of 0.489 and 0.388 for *NSF* and *MLens,* respectively, whereas the initial model demonstrates an RHR value of 0.358 and 0.281, respectively. This is particularly important, since users tend to click on content with higher ranks. We conclude that the collaborative model has significant advantages in terms of improving both the recommendation accuracy and the recommendation ranking.

6.3.3 Comparisons with other methods

To experimentally evaluate the performance of *top-N* recommendation, we calculated the hit rate (HR) and the reciprocal hit rank (RHR) obtained by *NB*, *VT*, *UCF*, *ICF* and M^+. We selectively varied the number of returned items N from 10 to 30 with an increment of 10. According to previous studies for collaborative filtering, the neighborhood size of *UCF* and *ICF* was set to 50.

Fig. 8 shows the results of RHR and HR for the *NSF* and *MLens* dataset, showing how M^+ outperforms the benchmark methods. As the number of recommended items *N* increases, the HR and RHR values tend to increase. Comparing the results achieved by M^+ and the benchmark algorithms, for both test sets, the HR value of the former was found to be superior to that of the benchmark methods in all cases. In the *NSF* dataset, on average, on all occasions, M^+ outperforms *VT*, *NB*, *UCF* and *ICF* by 6.7%, 16.7%, 7% and 8.5%, respectively. And for the *MLens* dataset, M^+ obtains 11.1%, 12.7%, 4.2%, and 4.2% improvement compared to *VT*, *NB*, *UCF*, and *ICF*, respectively. With respect

to RHR, similar results are demonstrated. More interestingly, M^+ significantly outperforms the other methods when a relatively small number of content items were recommended. For the *MLens* dataset, in the case of N=10, our method outperforms all of the other methods, whereas for the *NSF* dataset, only *VT* achieves comparable results. That is, M^+ provides more suitable content with a higher rank in the recommended content set, and thus can provide better quality of content for the target user than the other methods.

(Fig. 8 about here)

Ideally, recommender systems should provide a wide range of desirable content for users. Therefore, we continued to analyze the number of content items for which the methods, except for *NB*, could not provide any predictions for a user (i.e., the prediction value of the target user for the content was zero). Recall that *NB* and *VT* is a class of content-based filtering, whereas *UCF* and *ICF* is a class of collaborative filtering. Strictly speaking, our approach is closely connected with content-based filtering due to the dependence of content characteristics (i.e., content-based user models, content-based neighbors, content-based enrichments, and content-based recommendations). The results of the *NSF* dataset were that 2.6%, 7.1%, 7.1% and 2.9% of items for *VT*, *UCF*, *ICF* and M^+ could not be predicted, respectively. For the *MLens* dataset, 0.12%, 0.29%, 0.68% and 0.13% of items for *VT*, *UCF*, *ICF* and M^+ could not be predicted, respectively. As noted previously, such results are due to the fact that the collaborative filtering approaches, *UCF* and *ICF*, can only make predictions for items that at least a few users have rated. On the other hand, *VT* and M^+ can only make predictions for items that contain terms in the target user model, although they never suffer from cold start items.

These comparison experiments show that our collaborative model effectively and consistently improves the recommendation quality.

7. Conclusions and future work

Automated recommender systems are becoming widely used as a solution for reducing information overload of diverse domains. In this paper we presented a new and unique method for modeling user

interests via a collaborative approach of users. It also provides enhanced recommendation accuracy. The major advantage of the proposed modeling method is that it supports not only identification of each user's useful patterns but also enrichment of valuable neighbors' patterns. As noted in our experimental results, our model obtained better recommendation accuracy compared to the benchmark methods. Moreover, we also observed that our method can provide more suitable content for user preferences, even when the number of recommended items is small. There are common issues that have been mentioned in keyword-based analysis: homonymy and synonymy. We expect to improve our user model further by considering word semantics such as WordNet [23] or ontologies. Therefore, we plan to do further study on semantic user models in recommender systems.

References

[1] S. Berkovsky, T. Kuflik, F. Ricci, Mediation of user models for enhanced personalization in recommender systems, User Modeling and User-Adapted Interaction 18(3) (2008) 245-286.

[2] S. Berkovsky, T. Kuflik, F. Ricci, Cross-representation mediation of user models, User Modeling and User-Adapted Interaction 19 (2009) 35-63.

[3] R. Burke, Hybrid recommender systems: Survey and experiments, User Modeling and User-Adapted Interaction 12 (2002) 331-370.

[4] C.C. Chen, M.C. Chen, Y. Sun, PVA: A self-adaptive personal view agent, Journal of Intelligent Information Systems 18 (2002) 173-194.

[5] L. Chen, K. Sycara, WebMate: Personal agent for browsing and searching. Proceedings of the 2nd international conference on autonomous agents and multi agent systems, 1998, pp. 132-139.

[6] A. Das, M. Datar, A. Garg, Google News Personalization: Scalable online collaborative filtering. Proceedings of the 16th international conference on World Wide Web, 2007, pp. 271-280.

[7] M. Degemmis, P. Lops, G. Semeraro, A content-collaborative recommender that exploits WordNet-based user profiles for neighborhood formation. User Modeling and User-Adapted Interaction 17 (2007) 217-255.

[8] M. Deshpande, G. Karypis, Item-based top-n recommendation algorithms. ACM Transactions on Information Systems 22(1) (2004) 143-177.

[9] S. Flesca, S. Greco, A. Tagarelli, E. Zumpano, Mining user preferences, page content and usage to personalize website navigation. World Wide Web: Internet and Web Information System 8(3) (2005) 317-345.

[10] E. Gabrilovich, S. Dumais, E. Horvitz, Newsjunkie: Providing personalized news-feeds via analysis of information novelty. Proceedings of the 13th international conference on World Wide Web, 2004, pp.482-490.

[11] J. Han, J. Pei, Y. Yin, Mining frequent patterns without candidate generation: A Frequent-Pattern Tree Approach. Data Mining and Knowledge Discovery 8 (2004) 53-87.

[12] W. Lihua, L. Lu, L. Jing, Li. Zongyong, Modeling user multiple interests by an improved GCS approach, Expert Systems with Applications 29 (2005) 757-767.

[13] B. Magnini, C. Strapparava, User modelling for news web sites with word sense based techniques, User Modeling and User-Adapted Interaction 14 (2004) 239-257.

[14] A. McCallum, K. Nigam, A comparison of event models for naïve Bayes text classification. Proceedings of AAAI-98 workshop on learning for text categorization, 1998, pp.41-48.

[15] P. Melville, R. J. Mooney, R. Nagarajan, Content-boosted collaborative filtering for improved recommendations. Proceedings of the 18th national conference on artificial intelligence, 2002, pp. 187-192.

[16] M.J. Pazzani, A. Meyers, NSF Research Awards Abstracts 1990-2003. http://kdd.ics.uci.edu/databases/nsfabs/nsfawards.html, (2003).

[17] P. Resnick, N. Iacovou, M. Suchak, P. Bergstorm, J. Riedl, GroupLens: An open architecture for collaborative filtering of netnews. Proceedings of 1994 ACM conference on computer supported cooperative work, 1994, pp. 175–186.

[18] J. Salter, N. Antonopoulos, CinemaScreen Recommender Agent: Combining collaborative and content-based filtering. IEEE Intelligent Systems 21 (2006) 35-41.

[19] G. Salton, C. Buckley, Term weighting approaches in automatic text retrieval. Information Processing and Management 24 (1988) 513-523.

[20] B.M. Sarwar, G. Karypis, J.A. Konstan, J.T. Riedl, Analysis of recommendation algorithms for e-commerce. Proceedings of the 2nd ACM conference on electronic commerce, 2000, pp. 158-167.

[21] A. I. Schein, A. Popescul, L. H. Ungar, D. M. Pennock, Methods and metrics for cold-start recommendations. Proceedings of the 25th annual international ACM SIGIR conference on research and development in information retrieval, 2002, pp. 253-260.

[22] I. Schwab, W. Pohl, I. Koychev, Learning to recommend from positive evidence. Proceedings of the 5th international conference on intelligent user interfaces, 2000, pp. 241-247.

[23] G. A. Miller, WordNet: A Lexical Database for English. Communications of the ACM 38(11) (1995) 39-41.

[24] G. Adomavicius, A. Tuzhilin, Toward the Next Generation of Recommender Systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering 17(6) (2005) 734-749.

[25] J. L. Herlocker, J. A. Konstan, J. Riedl, Explaining collaborative filtering recommendations. Proceedings of the 2000 ACM conference on computer supported cooperative work, 2000, pp. 241-250.

[26] M. Montaner, B. Lopez, J. L. de la Rosa, A taxonomy of recommender agents on the Internet, Artificial Intelligence Review 19(4) (2003) 285–330.

[Tables]

Table 1. After mining user *u*'s content of interest, five personalized term patterns are found.

Pattern-id	PTP	PS	Length
p_1	$\{t_1, t_2, t_3\}$	0.56	3
p_2	$\{t_1, t_2, t_3, t_4\}$	0.51	4
p_3	$\{t_1, t_2, t_5\}$	0.47	3
p_4	$\{t_4, t_5\}$	0.41	2
p_5	$\{t_2, t_3, t_4\}$	0.32	3

Table 2. Datasets used in experimental evaluation.

	Number of users	Number of items	Number of interest items
NSF	78	974	9,845
MLens	658	1,682	50,318

Table 3. HR and RHR with respect to increasing neighborhood size (*NSF*).

Neighbors:	10	20	30	40	50	60
HR	0.1584	0.1636	0.1640	0.1651	0.1655	0.1643
RHR	0.4238	0.4891	0.4889	0.4745	0.4732	0.4732

Table 4. HR and RHR with respect to increasing neighborhood size (*MLens*).

Neighbors:	10	20	30	40	50	60
HR	0.2043	0.2136	0.2255	0.2262	0.2262	0.2288
RHR	0.2822	0.3666	0.3881	0.3881	0.3876	0.3732

[Figures]

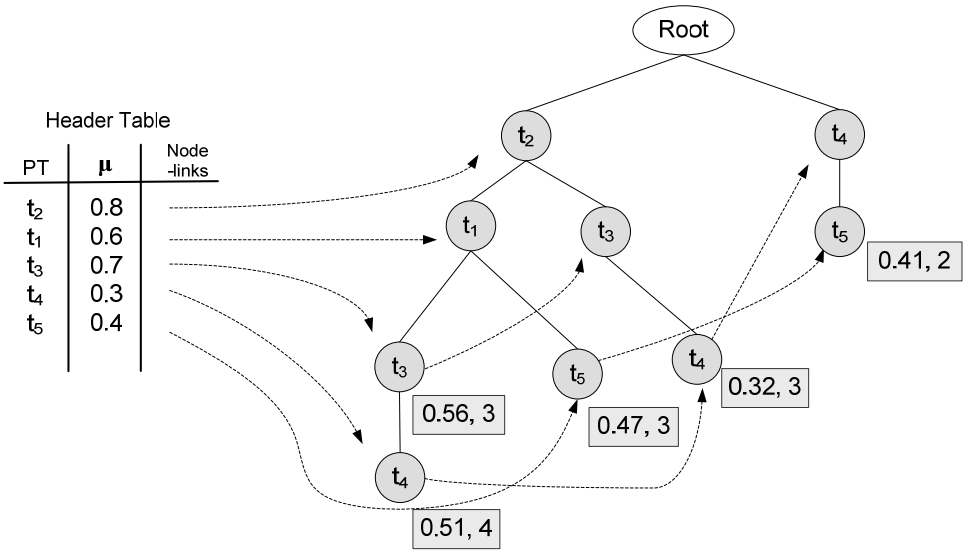

Fig. 1. A tree structure of M_u for personalized term patterns in Table 1.

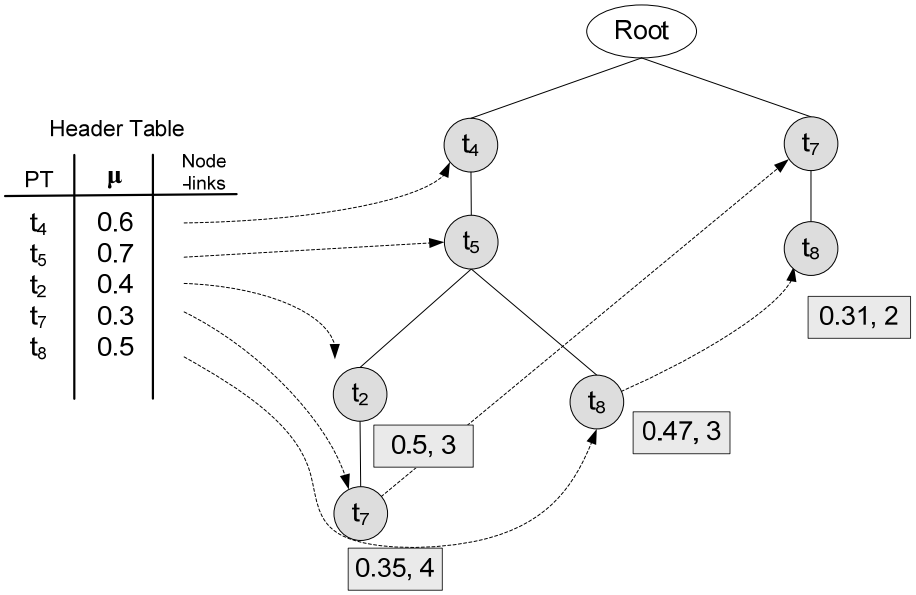

Fig. 2. Initial model for user *v* who is a neighbor of target user *u*.

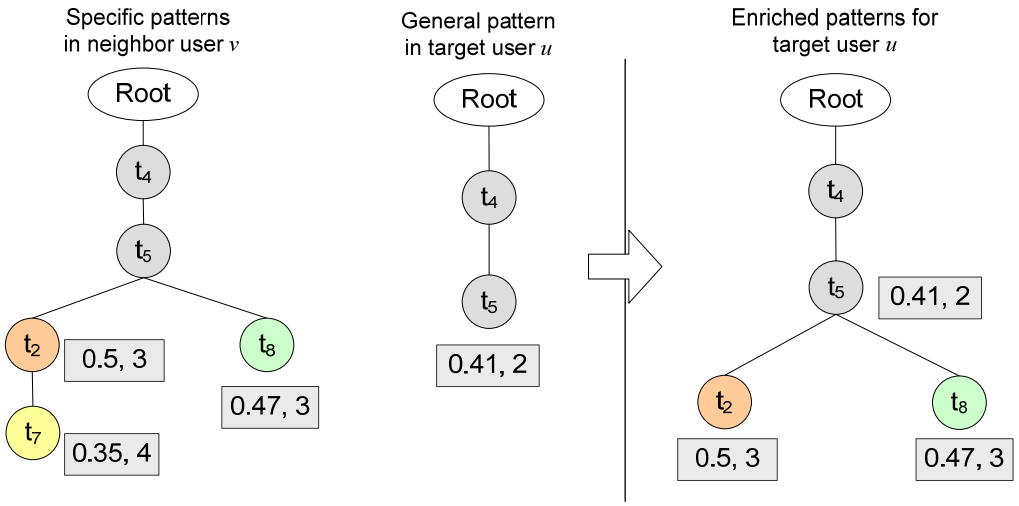

Fig. 3. Specific patterns, general pattern, and enriched patterns.

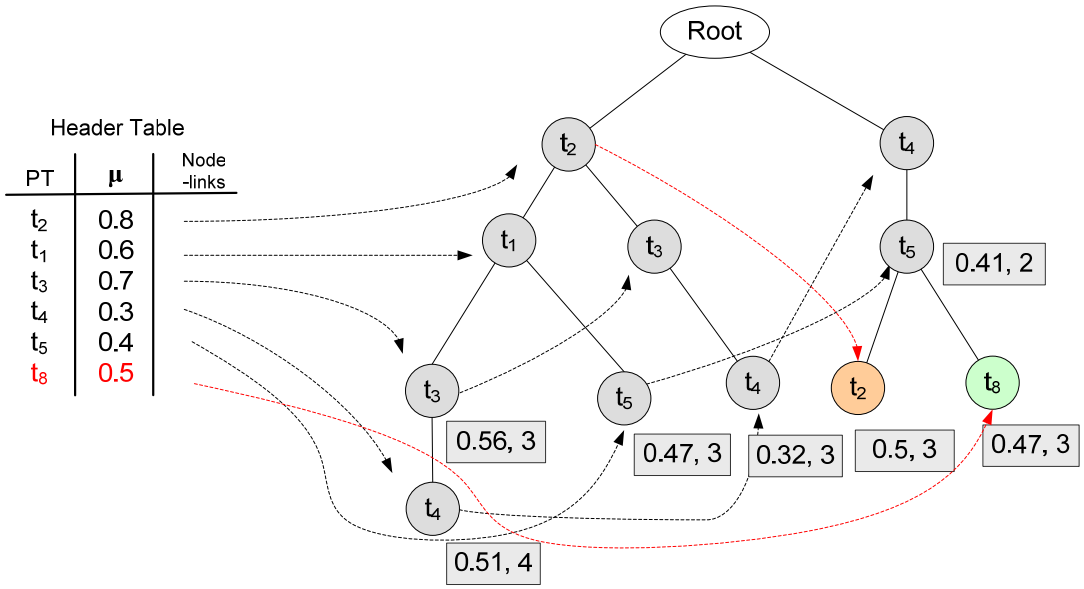

Fig. 4. Enriched user u model, M^{+}_{u}, by neighbor user v.

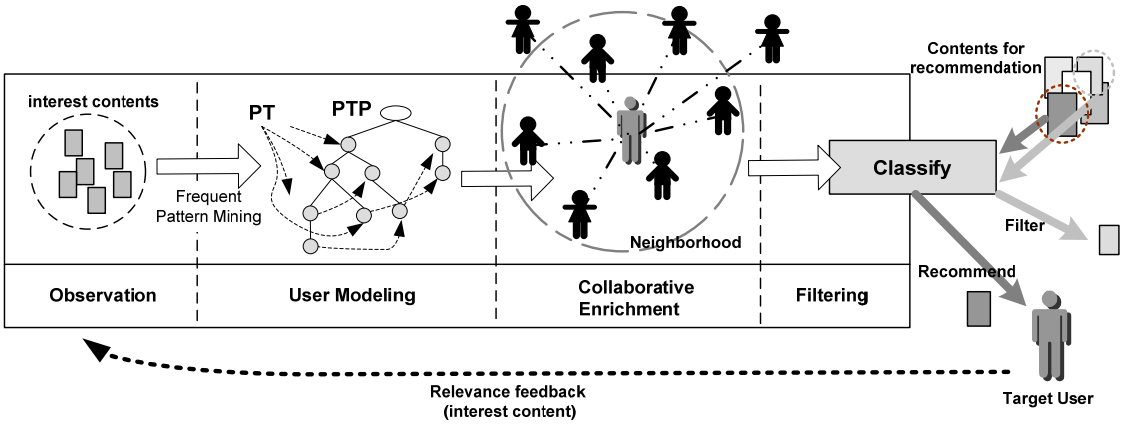

Fig. 5. An overview of PRCUM for content recommendations.

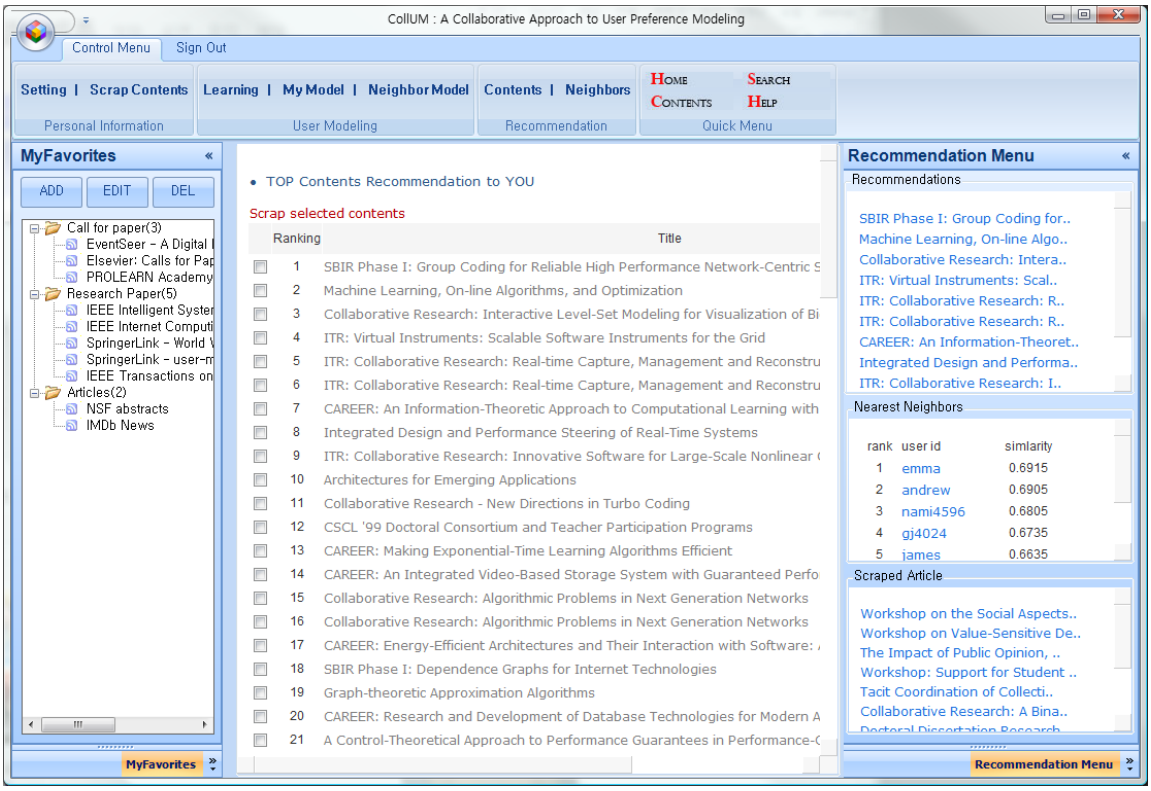

Fig. 6. A snapshot of the user interface for PRCUM.

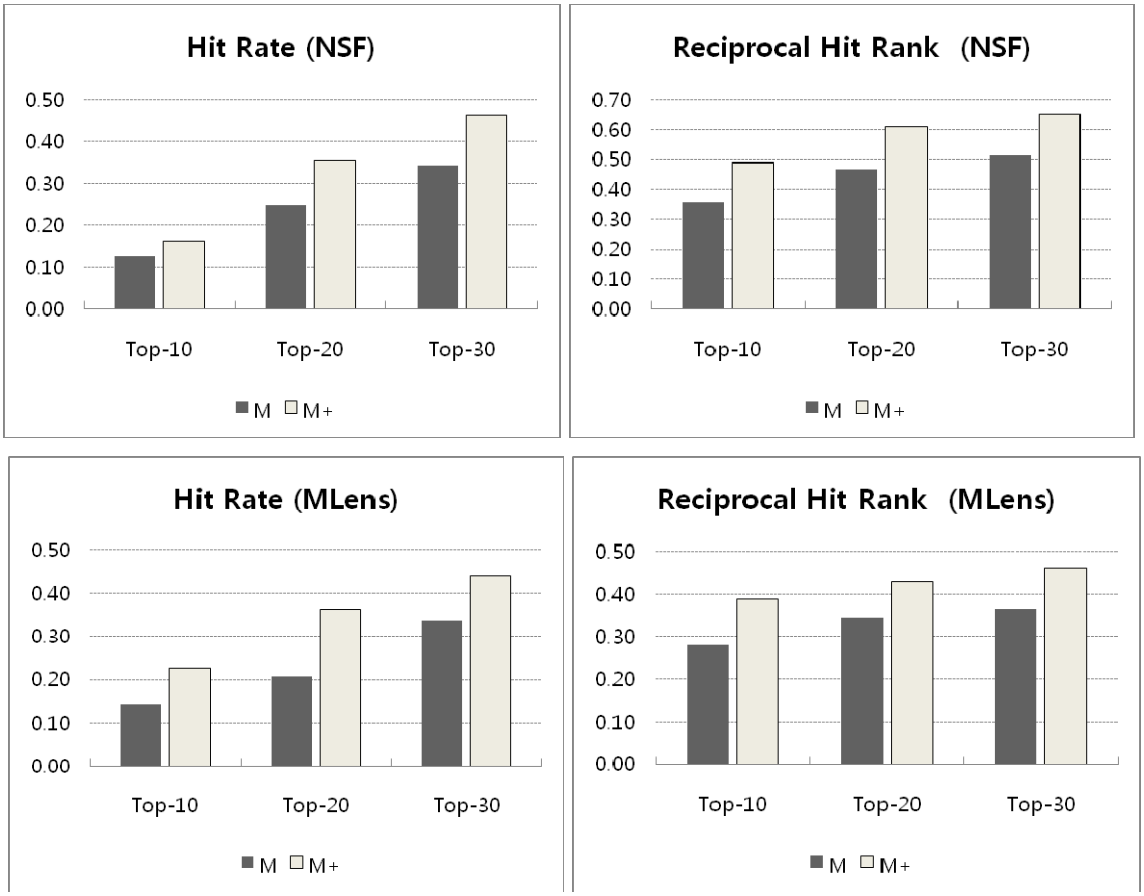

Fig. 7. Comparison of HR and RHR obtained by the initial model and the enriched model.

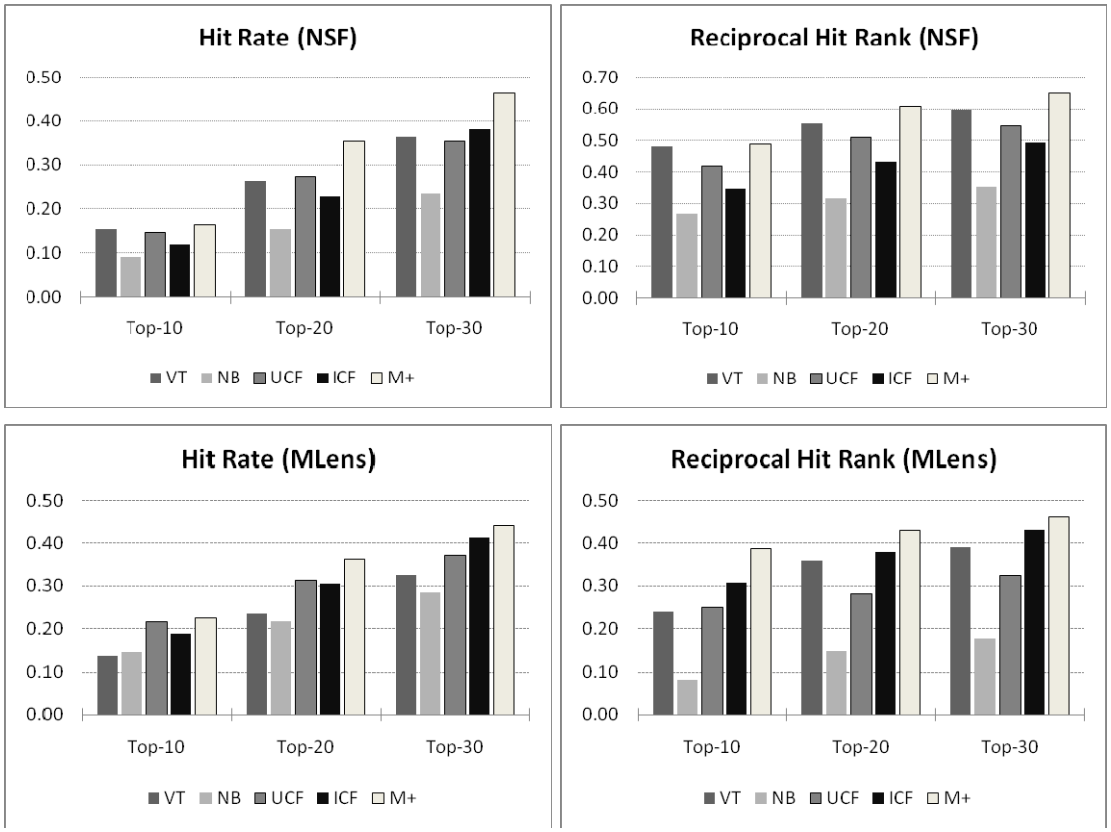

Fig. 8. Comparisons of HR and RHR with respect to increasing *N*.

[Biographical Note]

Heung-Nam Kim is a postdoctoral fellow in the Multimedia Communications Research Laboratory (MCRLab) at University of Ottawa, Canada. His research interests include collaborative filtering, recommender systems, semantic Web, data mining, user modeling, and social networking applications. He obtained his Ph.D. in Computer and Information Engineering from Inha University, Korea.

Inay Ha received the B.S. degree in Computer Science from University of Suwon and the M.Eng. degree in Computer and Information Engineering from Inha University, Korea, in 2007. She is currently working toward the Ph.D. with Intelligent E-Commerce Systems Laboratory (IESL), Inha University. Her research interests include Web mining, social networks, recommender systems, and intelligent e-Learning systems.

Kee-Sung Lee received the B.S. degree in Computer Science from Cheon-An University and the M.Eng. degree in Computer and Information Engineering from Inha University, Korea, in 2005. He is working as the Ph.D. student in Intelligent E-Commerce Systems Laboratory (IESL), Inha University. His research interests include semantic Web, image annotation and retrieval, information visualization and user interface design.

Geun-Sik Jo is a Professor in Computer and Information Engineering, Inha University, Korea. He is the chairman of the school of Computer and Information Engineering at Inha University. He received the B.S. degree in Computer Science from Inha University in 1982. He received the M.S. and the Ph.D. degrees in Computer Science from City University of New York in 1985 and 1991, respectively. He has been the General Chair and/or Technical Program Chair of more than 20 international conferences and workshops on artificial intelligence, knowledge management, and semantic applications. His research interests include knowledge-based scheduling, ontology, semantic Web, intelligent E-Commerce, constraint-directed scheduling, knowledge-based systems, decision support systems, and intelligent agents. He has authored and coauthored five books and more than 200 publications.

Abdulmotaleb El-Saddik University Research Chair and Professor, SITE, University of Ottawa and recipient of the Friedrich Wilhelm-Bessel Research Award from Germany's Alexander von Humboldt Foundation (2007) the Premier's Research Excellence Award (PREA 2004), and the National Capital Institute of Telecommunications (NCIT) New Professorship Incentive Award (2004). He is the director of the Multimedia Communications Research Laboratory (MCRLab). He is Associate Editor of the ACM Transactions on Multimedia Computing, Communications and Applications (ACM TOMCCAP), IEEE Transactions on Multimedia (IEEE TMM) and IEEE Transactions on Computational Intelligence and AI in Games (IEEE TCIAIG) and Guest Editor for several IEEE Transactions and Journals. Dr. El Saddik has been serving on several technical program committees of numerous IEEE and ACM events. He was the general co-chair of ACM MM 2008. He is leading researcher in haptics, service-oriented architectures, collaborative environments and ambient interactive media and communications. He has authored and coauthored two books and more than 200 publications. His research has been selected for the BEST Paper Award at the "Virtual Concepts 2006" and "IEEE COPS 2007". Dr. El Saddik is a Senior Member of ACM, an IEEE Distinguished Lecturer and a Fellow of the IEEE (FIEEE), the Canadian Academy of Engineers (FCAE) and the Engineering Institute of Canada (FEIC).