User Preference Modeling from Positive Contents for Personalized Recommendation

Heung-Nam Kim[1], Inay Ha[1], Jin-Guk Jung[1], and Geun-Sik Jo[2]

[1] Intelligent E-Commerce Systems Laboratory,
Department of Computer Science & Information Engineering, Inha University
{nami, inay, gj4024}@eslab.inha.ac.kr
[2] School of Computer Science & Engineering, Inha University,
253 Yonghyun-dong, Incheon, Korea 402-751
gsjo@inha.ac.kr

Abstract. With the spread of the Web, users can obtain a wide variety of information, and also can access novel content in real time. In this environment, finding useful information from a huge amount of available content becomes a time consuming process. In this paper, we focus on user modeling for personalization to recommend content relevant to user interests. Techniques used for association rules in deriving user profiles are exploited for discovering useful and meaningful patterns of users. Each user preference is presented the frequent term patterns, collectively called PTP (Personalized Term Pattern) and the preference terms, called PT (Personalized Term). In addition, a content-based filtering approach is employed to recommend content corresponding with user preferences. In order to evaluate the performance of the proposed method, we compare experimental results with those of a probabilistic learning model and vector space model. The experimental evaluation on *NSF research award* datasets demonstrates that the proposed method brings significant advantages in terms of improving the recommendation quality in comparison with the other methods.

1 Introduction

Thanks to technological developments related to the Internet and the World Wide Web, anyone living in today's information society can access a wealth of content and information on the web. However, in accordance with the massive growth of the Internet, users have to contend with an immense and huge amount of content, and often waste time trying to find content relevant to their interests. In addition, with the advent of blogs and RSS (Really Simple Syndication), a tremendous amount of content is generated overnight. Even if a user subscribes to content of interest, failing to read subscribed content for even a single day makes users feel overwhelmed the following day. Recommender systems have been issued as a solution to the problem of information overload [10]. In addition, user modeling for efficient personalization has become a key technique in recent information filtering systems [7, 9].

In this research, we focus on user modeling for personalization to recommend contents relevant to user interests. We exploit the techniques of data mining in deriving user preferences for discovering useful and meaningful patterns of users, collectively

V. Corruble, M. Takeda, and E. Suzuki (Eds.): DS 2007, LNAI 4755, pp. 116–126, 2007.

called PTP (**P**ersonalized **T**erm **P**attern). By capturing users' contents of interest, we mine the frequent term patterns and the preference terms existing in the user's contents of interest. The main objective of this research is to develop an effective method that provides high-quality recommendations of content relevant to user interests. In addition, we employ a content-based filtering approach to recommend content that is similar to personalized term patterns.

The subsequent sections of this paper are organized as follows: The next section contains a brief overview of some related work. In section 3, we describe our approach for modeling user preference and filtering contents. A performance evaluation is presented in section 4. Finally, conclusions are presented and future work is discussed in section 5.

2 Related Work

This section briefly explains previous studies related to user modeling and personalized recommendation. Two approaches for recommender systems have been discussed in the literature, *i.e.*, a content-based filtering approach and a collaborative filtering approach. The traditional task in the 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 [10]. Instead of computing the similarities between the users, the content-based filtering systems recommend only the items that are highly relevant to the single user profile by computing the similarities between the items and the user preference [9]. This research focuses only on the content-based filtering for personalized recommendations. Personalized recommender systems based on a single user have been developed learning procedures and need to use training data to identify personal preference from information object and their contents. *Webmate* tracks user interests from his positive information only (i.e., documents that the user is interested in) and exploits the vector space model using TF-IDF method [3]. A classification approach has been explored to recommend articles relevant user profile, such as *NewsDude* and *ELFI* [4, 5]. In *NewsDude*, two types of the user interests are used: short-term interests and long-term interests. To avoid recommendations of very similar documents, short-term profile is used. For the long-term interests of a user, the probabilities of a document are calculated using Naïve Bayes to classify a document as interesting or not interesting. Instead of learning from users' explicit information, *PVA* learns a user profile implicitly without user intervention, such as relevance feedback, and represents it as keyword vector in the form of a hierarchical category structure [8], similar to *Alipes* [6]. In *Newsjunkie*, novelty-analysis algorithm is employed to present novel information for users by identifying novelty of articles in the contexts of articles they have already reviewed [12]. Although these systems have their own method to building a user model, they do not deliberate on concurrence of terms and offer the ability to identify meaningful or useful patterns, which are important features for representing articles or contents [13]. For example, when content contains 'apple Macintosh computer', the semantic of 'apple' are discriminated from those of apple in 'apple pie'. Likewise, mouse in 'optical mouse' implies not an animal but an input device of computers. Therefore, our motivation is to develop a learning algorithm which supports the identification of useful patterns of a user.

3 User Preference Modeling for Content Recommendation

The proposed method is divided into three phases: an observation phase, a user modeling phase, and a content filtering phase. Fig. 1 provides a brief overview of the proposed approach.

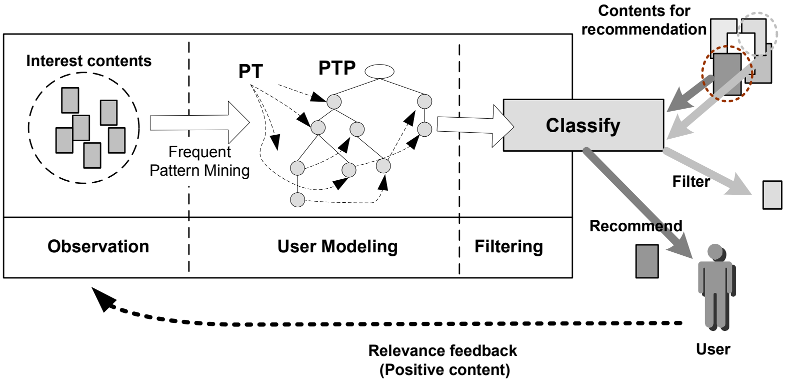


Fig. 1. Overview of the proposed method for personalized content recommendations

3.1 Modeling User Preference

The capability to learn users' preferences is at the heart of a personalized recommender system. Additionally, since every user has different interests, feature selection for representing users' interests should be personalized and be performed individually for each user [9]. In this section, we describe our approach to modeling user preference, which is mined from the user's preferred contents (positive contents).

The first step in user modeling is the extraction of the terms from positive contents that have been preprocessed by: removing stop words and stemming words [15]. After extracting terms, each positive content C_j is represented as a vector of attribute-value pairs as follows:

$$C_j = \{(T_1, w_{1,j}), (T_2, w_{2,j}), \ldots, (T_m, w_{m,j})\}$$

where T_i is the extracted term in C_j and $w_{i,j}$ is the weight of T_i in C_j, which is computed by static TF-IDF term-weighting scheme [1] and defined as follows:

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

where $f_{i,j}$ is the frequency of occurrence of term T_i in content C_j, n is the total number of contents in the collections, and n_i is the number of contents in which term T_i occurs. The weight indicates the importance of a term in representing the content. All weight values of terms, $w_{i,j}$, in a positive content C_j are sorted in descending order. The first K terms (Top-K terms) are selected as content C_j features and used to mine frequent term patterns that occur at least as frequently as a predetermined minimum support, i.e., $PS > min\_sup$ [16].

Definition 1 (Pattern Support, PS). Let $T = \{T_1, T_2, \ldots , T_m\}$ be a set of terms, I_u be a set of contents of interest of user u where each content C is a set of terms such that $C \subseteq T$. Let pattern P_k be a set of terms. A content C is said to contain pattern P_k if and only if $P_k \subseteq C$. *Pattern support* for pattern P_k, $PS(P_k)$, in I_u is the ratio of contents in I_u that contain pattern P_k.

In this paper, each transaction corresponds to a positive content of a user and items in transaction are terms extracted from the content. For effective mining of the term patterns, we should choose a minimum support threshold. A high *min_sup* discards more patterns, and thus remaining term patterns may not be sufficient to represent user preference. In contrast, a low *min_sup* includes many noise patterns. Therefore, the threshold is chosen heuristically through experiments.

Once the patterns are mined, a model for user u is defined as a tuple $\mathbf{M}_u$= ($\mathbf{PTP}_u$, $\mathbf{PT}_u$) 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 to save memory space and explore relationships of terms.

Definition 2 (Personalized Term Patterns, PTP). If the pattern support of pattern P_k, that is composed of at least l different terms ($l \geq 2$), satisfies a pre-specified minimum support threshold (*min_sup*), then pattern P_k is a frequent term pattern. *Personalized term patterns* for user u, PTP_u, is defined as a set of frequent term patterns.

Definition 3 (Personalized Term, PT). *Personalized term* is a term that occurs within *personalized term patterns*. The set of *personalized terms* for user u is denoted as PT_u, $PT_u \subseteq T$. 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 of term weight for term T_i and is computed as follows:

$$\mu_{i,u} = \frac{1}{|I_u(i)|} \times \sum\nolimits_{j \in I_u(i)} w_{i,j}$$

where $I_u(i)$ is a set of contents of interest for user u containing term T_i and $w_{i,j}$ is the term weight of term T_i in content C_j.

For example, if five personalized term patterns are found, as shown in Table 1, after mining content of interest for user u, a tree structure of a model for user u is then constructed as follows.

All $\mathbf{PT}_u$ are stored in header table and sorted according to descending their frequency. First, create the root of the tree, labeled with "null". For the first term pattern, $\{T_1, T_2, T_3\}$ is insert into the tree as a path from root node where T_2 is linked as child of the root, T_1 is linked to T_2, and T_3 is linked to T_1. And *PS* and *length* of the pattern*PS(P*$PS(P_1)$=0.56, *length*=3) are then attached to the last node T_3. For the second pattern, since its term pattern, $\{T_1, T_2, T_3, T_4\}$, shares 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

Table 1. After mining content of interest of user u, 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

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, 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, a model for user *u* is constructed as shown in Fig. 2.

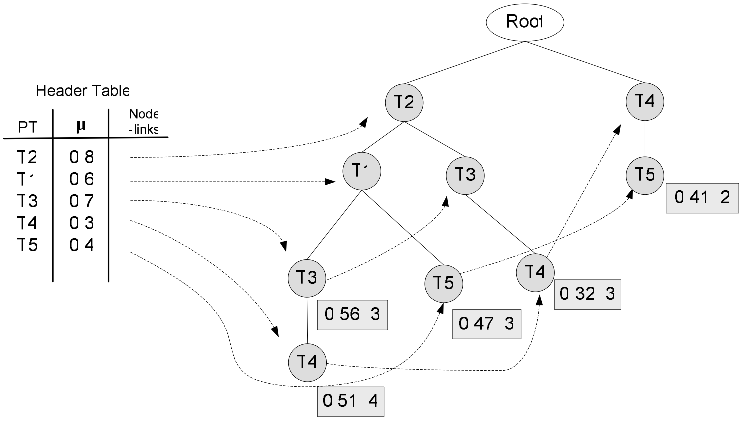


Fig. 2. A tree structure of M_u for personalized term patterns in Table 1

3.2 Personalized Content Filtering

In this paper, we consider two aspects for judging whether content is relevant or irrelevant to the user based on user preference. First, cosine similarity [13, 15], which quantifies the similarity of two vectors according to their angle, is employed to measure the similarity values between new content and **PT** for a user. As noted in Definitions 4, the personalized terms of user *u*, PT_u, are represented as the vector of attribute-value pairs. Further, the term vector for the new content C_n is represented by $\vec{c}_n$ = ($w_{1,n}$, $w_{2,n}$, …, $w_{t,n}$), where the weight $w_{i,n}$ is the TF-IDF value of term T_{i} in content C_n. Therefore, content C_n and **PT** of user *u*, PT_u are represented as t-dimensional vectors, and the cosine similarity for theses two vectors, $\vec{PT}_u$ and $\vec{c}_n$ is measured by equation (1) [15].

$$Sim(u,C_n)=\frac{\overrightarrow{PT_u}\cdot\overrightarrow{c_n}}{|\overrightarrow{PT_u}|\times|\overrightarrow{c_n}|}=\frac{\sum_{k=1}^{t}\mu_{k,u}\times w_{k,n}}{\sqrt{\sum_{k=1}^{t}\mu_{k,u}^{2}}\times\sqrt{\sum_{k=1}^{t}w_{k,n}^{2}}} \tag{1}$$

The second approach considers matched patterns between the new contents and **PTP** for a user. Formally, the similarity between content C_n and user u is defined in equation (2).

$$Sim(u,C_n)=\frac{|MP|}{|PTP_u|}\times\sum_{P_k\in MP}length(P_k)\cdot PS(P_k) \tag{2}$$

where MP is a set of matched patterns between PTP_u and content C_n. $PS(P_k)$ and $length(P_k)$ refer to the pattern support value and the length of matched pattern P_k, respectively. The main concept of the second scheme dictates that patterns with numerous occurrences in user preference present a greater contribution with regard to similarity than patterns with a smaller number of occurrences.

Definition 4 (Matched Pattern). Let TP_k = {T_1, T_2, ... , T_n} be a set of terms contained in pattern P_k such that TP_k is a subset of personalized terms for user u, $TP_k \subseteq PT_u$. If all terms in contained P_k appear content C_n, $TP_k \subseteq C_n$, then pattern P_k is deemed a *matched pattern* between PTP_u and content C_n.

Each similarity value, which is obtained by using the equation (1) and (2), is normalized to [0, 1] and divided by the maximum similarity value, i.e., $sim(u, C_n)/max_l\ sim(u, C_l)$. Once the similarities between user u and the new contents, which the user u has not yet read, are computed, the contents are sorted in order of descending similarity value. Two strategies can then be used to select the relevant contents to user u. First, if the similarity values are greater than a reasonable threshold value (i.e., $sim(u, C_n)/max_l\ sim(u, C_l) > \theta$), the contents are recommended to user u [3, 5]. Second, a set of N rank contents that have obtained higher similarity values are identified for user u, and then those contents are recommended to user u (Top-N recommendation) [10]. We choose the second approach for filtering the personalized contents.

Definition 5 (Top-N recommendation). Let C be a set of all contents, I_u be a content list that user u has already collected or added to his preference list (positive contents), and NI_u be a content list that user u has not yet read, $NI_u = C - I_u$ and $I_u \cap NI_u = \emptyset$. Given two contents C_i and C_j, $C_i \in NI_u$ and $C_j \in NI_u$, content C_i will be of more interest to user u than content C_j if and only if a similarity value $sim(u, C_i)$ between user u and content C_i is higher than that of content C_j, $sim(u,C_i) > sim(u, C_j)$. Top-N recommendations for user u identifies an ordered set of N contents, $TopN_u$, that will be of interest to user u such that $|TopN_u| \leq$ N, $TopN_u \cap I_u = \emptyset$, and $TopN_u \subseteq NI_u$.

4 Experimental Evaluation

In this section, experimental results of the proposed approaches are presented. All experiments were carried out on a Pentium IV 3.0GHz with 2GB RAM, running a

MS-Window 2003 server. In order to mine personalized term patterns, FP-growth software implemented by Frans Coenen[1] was used.

The experimental data is taken from NSF (National Science Foundation) research award abstracts [14]. The original data set contains 129,000 abstracts describing NSF awards for basic research from 1900 to 2003. However, the set is too large to be used for experiments, and thus we selected award abstracts from 2000 to 2003, i.e. the selected data set contained 30,384 abstracts and 3,086,090 terms as obtained from the abstracts (cf. 22,236 distinct terms). 10 users participated in the experiments by scrapping only contents relevant to their interests from the total contents (30,384 contents). Whenever they found the content related to their own preferences, they added that content to their preference list. Each user added at least 700 content items. To evaluate the performance of the proposed approaches, we divided the preference contents of the users into *a test set* with exactly 100 contents per user in the test set and *a training set* with the remaining contents. A model $\mathbf{M}_{\mathrm{u}}$ of each user was then constructed using only the *training set*. We assume that each user does not change his/her interests during the experiments if a user preference is learned (static user profile) [9].

The performance was measured by looking at the number of *hits*, and their *ranking* within the *top*-N contents and the overall contents that were recommended by a particular scheme. We computed three quality measures that are defined as follows.

Hit Rate (HR). In the context of *top*-N recommendations, *hit-rate*, a measure of how often a list of recommendations contains contents that the user is actually interested in, was used for the evaluation metric [6, 10]. The *hit-rate* for user *u* is defined as:

$$HR(u) = \frac{|\,Test_u \hbar \ TopN_u\,|}{|\,Test_u\,|}$$

where $Test_u$ is the content list of user *u* in the test data and $TopN_u$ is a *top*-N recommended content list for user *u*. Finally, the overall *HR* of the *top*-N recommendation for all users is computed by averaging these personal *HR* in 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 contents. In other words, content that is recommended with a top ranking is treated equally with content that is recommended with an *N*th ranking [10]. To address this limitation, therefore, we adopted *the reciprocal hit-rank* metric described in [10]. The *reciprocal hit-rank* for user *u* is defined as:

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

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

[1] The software is available at http://www.csc.liv.ac.uk/~frans/KDD/Software/

Reciprocal Total Rank (RTR). This metric is similar to *the reciprocal hit-rank* but instead of only using the ranking of the *hit set* it uses the ranking of all test data for user *u*. We refer to this as *the reciprocal total rank* for user *u* and is defined as follows:

$$RTR(u) = \sum_{C_n \in Test_u} \frac{1}{rank(C_n)}$$

where *rank(C$_n$)* refers to a recommended ranking of content C_n for user *u* in the test data. Likewise, the overall *RTR* for all users is also computed by averaging the personal *RTR(u)* in test data.

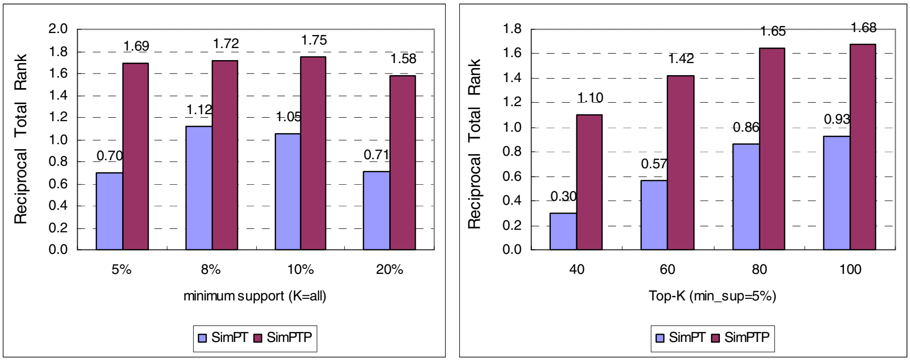
Benchmark Algorithms. In order to compare the performance of the proposed scheme, a probabilistic learning algorithm, which applies a *naïve Bayesian classifier* (denoted as *NB*) [4, 5], and a TF-IDF vector-based algorithm, which is employed in the *Webmate* system (denoted as *Webmate*) [3], were implemented. To make the comparison fair, both of the algorithms were designed to learn users' preferences from positive examples only. For the content filtering process, in the case of *NB*, contents are ranked using the calculated probability value whereas they are ranked using the calculated cosine similarity for *Webmate*. The *top*-N recommendation of our strategy was then evaluated in comparison with the benchmark algorithms.

4.1 Experimental Results

In this section, we present the experimental results of the proposed algorithms. In our algorithms, *SimPT* denotes when equation (1) is used for the similarity method, whereas *SimPTP* denotes the case of equation (2). The performance evaluation is divided into two dimensions. The sensitivity of the two parameters *minimum support* and *Top-K terms* were first determined, and then the quality of the *top*-N recommendations is evaluated.

4.1.1 Experiments with Minimum Support

As noted previously, minimum support controls the size of $\mathbf{M}_u$. In general, if the size of $\mathbf{M}_u$ is too small, some information may be lost. On the other hand, if it is too large, some noise patterns may be included. Therefore, different *min_sup* values were used for mining personalized term patterns: 5%, 8%, 10%, and 20%. In addition, we selected all terms as the content feature during the mining process (*K*=all). Examining the average number of patterns in the users' $\mathbf{M}_u$, in the case of *min_sup*=5%, we found that 2667 patterns had been mined, whereas the average number was 1049, 490, and 58 in the case of *min_sup*=8%, *min_sup*=10%, and *min_sup*=20%, respectively. The recommendation performance obtained by changing *min_sup* in terms of RTR is shown in Fig. 3 (a). The results demonstrate that, at all *min_sup* levels, *SimPTP* provides more accurate recommendations than *SimPT*. For example, when *min_sup* is set to 10%, *SimPTP* yields a RTR of 1.75, which is the best value, whereas *SimPT* gives a RTR of 1.05. It is observed from the graph that the performance of *SimPTP* is slightly affected by *min_sup* relative to that of *SimPT*. These results indicate that even for a small size of $\mathbf{M}_u$, *SimPTP* provides reasonably accurate recommendations. Note that a suitable size should be selected for vector-based similarity approaches such as *SimPT*.



(a) Minimum support (b) Top-K

Fig. 3. Reciprocal total rank (RTR) according to variation of *min_sup* (a) and *K* (b)

4.1.2 Experiments with Top-K Terms

Theoretically, all terms extracted from the contents can be applied immediately to mine the personalized term patterns. However, the complexity of the learning process is increased by the content feature size. In order to reduce the feature size and refine noise terms, the *K* highest weight terms are selected instead of selecting all terms. In these experiments, *min_sup*=5% was chosen because the sufficient patterns for representing user preference were not discovered at high thresholds of *min_sup* (i.e., *min_sup*=10%, 20%). To evaluate the sensitivity to the value of *K* we performed an experiment with *K* values of 40, 60, 80, and 100. As in the previous experiments, we analyze the average number of patterns mined for users. As a result, patterns of 21, 84, 344, and 1064 were discovered on average when *K* was set to 40, 60, 80, and 100, respectively. That is, the mined patterns were clearly reduced as compared with the number of patterns discovered in the previous experiment (*min_sup*=5%). Fig. 3 (b) depicts the variation of RTR according to the value of *K*. It can be observed from the graph that *SimPTP* yields better RTR than *SimPT*. When we compare the results of RTR achieved by *SimPTP* using *K*=all and *K*=100, *SimPTP* in the case of *K*=100 (RTR of 1.68) offers reasonable performance comparable to that of *K*=all (RTR of 1.69). On the contrary, RTR of the *SimPT* using *K*=100 (RTR of 0.93) is superior to that of *SimPT* using *K*=all (RTR of 0.70). This is particularly important since a small amount of content features leads to low computational requirements.

4.1.3 Comparisons of Performance

For evaluating the *top*-N recommendation, the number of recommended contents (the value of *N*) was increased, and we calculated the hit rate (HR) and the reciprocal hit rank (RHR) achieved by *SimPT*, *SimPTP*, *Webmate*, and *NB*. Table 2 summarizes the results of RHR while Table 3 summarizes the HR of the algorithms as the value of *N* increased from 100 to 500. In general, with the growth of recommended items *N*, HR, and RHR tend increase. Although HR for all algorithms is unsatisfactorily low at a small number of *N*, *SimPTP* provides considerably improved HR on all occasions compared to the benchmark algorithms. Similar conclusions can be made by looking

Table 2. Comparison of the reciprocal hit rank (RHR) as the value of N increases

Algorithms	100	200	300	400	500	Average
SimPT	0.8550	0.9300	0.9813	1.0088	1.0238	0.9598
SimPTP	1.5088	1.6525	1.7250	1.7325	1.7338	1.6705
Webmate	1.3031	1.3942	1.4213	1.4406	1.4512	1.4003
NB	1.1575	1.2413	1.2775	1.2863	1.2925	1.2510

Table 3. Hit rate (HR) as the value of N (number of recommended contents) increases

Algorithms	100	200	300	400	500	Average
SimPT	0.17	0.28	0.41	0.50	0.57	0.38
SimPTP	0.28	0.38	0.57	0.68	0.69	0.52
Webmate	0.14	0.27	0.35	0.42	0.47	0.33
NB	0.16	0.27	0.36	0.39	0.43	0.32

(*K*=all and *min_sup*=10% for *SimPT* and *SimPTP*)

at the RHR results as well. In addition, comparing the results achieved by *SimPT* and the benchmark algorithms, HR of the former found to be superior to that of the benchmark algorithms. However, with respect to RHR, *SimPT* is worse than that of the benchmark algorithms. Overall, *SimPTP* achieves 19% and 36% improvement in terms of RHR on average, compared to *Webmate* and *NB*, respectively, whereas *SimPT* brings 33% and 23% degradation of RHR, respectively. We conclude from this experiment that the proposed strategy for top-N recommendation is effective in terms of improving the performance, although RHR is diminished in the case of *SimPT*.

5 Conclusions

The capability to model users' preferences is at the heart of a personalized recommender system that discriminates interesting information from uninteresting data. In this paper, a new and effective method for learning and modeling user preferences and for filtering contents relevant user interests is proposed. The major advantage of the proposed learning method is that it supports the identification of useful patterns of each user. In order to evaluate the effectiveness of the approach, we compare our experimental results with those of probabilistic learning model and vector space model. The experimental results demonstrate that the proposed method offers significant advantages in terms of improving recommendation quality as compared to the traditional learning algorithms. A research area that is attractive attention at present is collaborative modeling of user preferences among users with similar interest. In addition, we are currently extending our algorithm to allow for changing user interests. Therefore, we plan to further study the techniques of adaptive and incremental learning [6, 13].

References

1. Salton, G., Buckley, C.: Term Weighting Approaches in Automatic Text Retrieval. Information Processing and Management, Vol.24 (1988) 513-523
2. Agrawal, R., Srikan, R.: Fast Algorithms for Mining Association Rules. In Proc. of the 20th Int. Conf. on Very Large Data Bases (1994)
3. Chen, L., Sycara, K.: WebMate: Personal Agent for Browsing and Searching. In Proc. of the 2nd Int. Conf. on Autonomous Agents and Multi Agent Systems (1998) 132-139
4. Billsus, D., Pazzani, M.J.: A hybrid user model for News story classification. In Proc. of the 7th Int. Conf. on User Modeling (1999) 99–108
5. Schwab, I. Pohl, W., Koychev, I.: Learning to Recommend from Positive Evidence. In Proc. of Int. Conf. on Intelligent User Interfaces (2000)
6. Widyantoro, D.H., Ioerger, T., Yen, J.: Learning User Interest Dynamics with a Three-Descriptor Representation. Journal of the American Society for Information Science and Technology, Vol. 52 (2001) 212-225
7. Aggarwal, C.C., Philip, S.Y.:An Automated System for Web Portal Personalization. In Proc. of the 28th VLDB Conference (2002) 1031-1040
8. Chen, C.C., Chen, M.C., Sun, Y.: PVA: A Self-Adaptive Personal View Agent. Journal of Intelligent Information Systems, Vol.18 (2002) 173-194
9. Eirinaki, M., Vazirgiannis, M.: Web Mining for Web Personalization, ACM Transactions on Internet Technology, Vol. 3 (2003) 1-27
10. Deshpande, M., Karypis, G.: Item-based Top-N Recommendation Algorithms. ACM Transac-tions on Information Systems, Vol. 22 (2004) 143–177
11. Han, J., Pei, J., Yin, Y.: Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach. Data Mining and Knowledge Discovery, Vol. 8 (2004) 53-87
12. Gabrilovich, E., Dumais, S., Horvitz, E.: Newsjunkie: Providing Personalized News-feeds via Analysis of Information Novelty. In proc. of the 13th Int. Conf. on World Wide Web (2004) 482-490
13. Chung, S., McLeod, D.: Dynamic Pattern Mining: An Incremental Data Clustering Approach. Lecture Notes in Computer Science, Vol. 3360 (2004) 85-112
14. Pazzani, M.J., Meyers, A.: NSF Research Awards Abstracts 1990-2003. http://kdd.ics.uci.edu/databases/nsfabs/nsfawards.html
15. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. Addison -Wesley (1999)
16. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, 2nd ed. Morgan Kaufmann Publishers (2006)