Enabling Personalized Recommendation on the Web Based on User Interests and Behaviors

Yi-Hung Wu Yong-Chuan Chen Arbee L. P. Chen
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan 300, R. O.C.
Email: alpchen@cs.nthu.edu.tw

Abstract

The dramatic growth of the Web has brought about the rapid accumulation of data and the increasing possibility of information sharing. As the population on the Web grows, the analysis of user interests and behaviors will provide hints on how to improve the quality of service. In this paper, we define user interests and behaviors based on the documents read by the user. A method for mining such user interests and behaviors is then presented. In this way, each user is associated with a set of interests and behaviors, which is stored in the user profile. In addition, we define six types of user profiles and a distance measure to classify users into clusters. Finally, three kinds of recommendation services using the clustered results are realized. For performance evaluation, we implement these services on the Web to make experiments on real data/users. The results show that the average acceptance rates of these services range from 71.5% to 94.6%.

1. Introduction

Owing to the booming development of the Web, users have to face a variety and a large number of web pages and often waste a lot of time on searching. To alleviate the difficulty, many tools have been developed and used on the Web. Search engines are the most popular tools and usually provide two basic functions. One is to collect the descriptions of web pages and organize them as an index. The other is to find the most relevant web pages against the index for user queries. The techniques of *information retrieval* are commonly used for building search engines. In general, the search engine aims at providing users a way to search all over the Web.

The concept of *information filtering* [3] is another way to tackle this problem. In a filtering system, users are associated with *profiles* that describe what they need, while data are represented in the same form of user profiles. Comparing data with profiles, the users who are interested in the data are identified and informed. This concept has been applied to various information systems on the Internet, e.g., *SIFT* [21] and *SIFTER* [12].

Instead of searching data for users, the filtering system finds the matched profiles for given data. Therefore, it can save the costs on data indexing and perform as efficiently when the volume of data grows. Moreover, the filtering system aims at providing users a way to share information with others. Most of the researches attempt to solve the major issue on how to select the matched profiles for given data. Two kinds of approaches have been presented in the literature, i.e., *content-based* and *collaborative filtering*.

Content-based filtering. The first approach identifies and provides the relevant data for the users based on the similarity between data and profiles. The critical problem is how to index the user profiles for efficient matching. The techniques for text retrieval [19] can be applied to profile indexing. Yan et al. [22] introduce the *inversion-based* methods, while we propose the *signature-based* methods in [20] to reduce the storage costs of indexes. Since the content-based filtering approach is dominated by contents, its effectiveness depends on whether the profiles well describe the user's information needs. If the users do not provide precise descriptions (e.g., the naïve users), the filtering process may lead to undesired results. Moreover, this approach is unable to provide unexpected but interesting findings for users.

Collaborative filtering. The second approach identifies the relevant users who own similar profiles and provides the data they like to each other. Rather than the similarity between data and profiles, this approach measures the similarity between profiles. The chief concern of this research is how to cluster the user profiles for effective filtering. *BIRCH* [23] and *DBSCAN* [5] are two methods that can incrementally cluster multi-dimensional data points.

0-7695-0957-6/01 $10.00 © 2001 IEEE

Since the collaborative filtering approach is dominated by user clusters, its effectiveness depends on whether the clustering of profiles correlates the users well. If the system cannot derive meaningful user clusters, the filtering process may lead to undesired results. Moreover, this approach may provide unexpected findings for users due to its nature of information sharing.

The filtering approaches are useful in personalizing the system interactions with the individual users. Example applications range from keystroke prediction [7] and TV listing [17], to web navigation [8] and search [9]. In this paper we focus on the *personalized recommendation* service, which provides recommendation for individual users on the Web. *Tapestry* [6] and *GroupLens* [10] allow users to comment on Netnews documents and get the ones recommended by the others. In these systems, users have to specify their profiles explicitly (e.g., ratings of the data). However, it is inconvenient for the users who often change their interests to update their profiles frequently. A way to derive the user profiles implicitly will be helpful. The techniques for *machine learning* and *data mining* [4] are commonly used. *Letizia* [11] and *Amalthaea* [13] are two example systems that learn to imitate user browsing. They do not require the intervention of users. Shapira et al. [16] also suggest a two-phase filtering process. In addition, Pazzani and Billsus [14] apply the techniques of *relevant feedback* to revise the user profiles.

On the Web, user requests are often logged as the browsing history. Obviously, the browsing history is a good source to indicate what the users want [15] [18] and to derive the user profiles. In this paper, we employ the techniques of association rule mining [1] to derive such profiles. Our method emphasizes on the adaptability to the large amount and dynamic nature of web data. Two kinds of profiles representing user interests and behavior respectively are derived and six types of profiles are defined. Furthermore, we devise a distance measure and the method to classify the user profiles into clusters. Finally, three kinds of recommendation services using the clusters are realized. For performance evaluation, we implement these services on the Web to make experiments on real data/users.

The rest of this paper is organized as follows. In Section 2, we introduce our system for personalized recommendation and the definitions of user interests and behaviors. In addition, we illustrate the mining algorithms and the six types of user profiles. The clustering method is detailed in Section 3. In Section 4, we describe three ways of recommendation based on the clusters. Due to the lack of space, we brief the performance evaluation in Section 5 but omit the experiment details. At last, we remark the conclusion and future works.

2. Mining user interests and behaviors

2.1. System architecture

We implement an online thesis system (abbreviated as *OTS*) to provide both the content-based and collaborative filtering services. For simplicity, we limit the data of the OTS to the papers taken from the publication server on the Web. Figure 1 shows the architecture of the OTS.

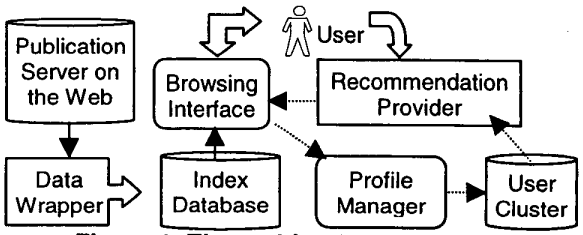

Figure 1: The architecture of the OTS

Before the OTS starts, the *data wrapper* collects the bibliographic data from the publication server to build the *index database*. After that, the *browsing interface* helps users browse the entire database by categories. If the user downloads a paper, the OTS will pass its categories to the *profile manager*, which can derive the user interests and behaviors. The users are divided into clusters based on to the predefined types of user profiles and our clustering method. Whenever the user requests for recommendation, the *recommendation provider* will deduce the papers to be recommended from the user clusters. Note that the OTS always keeps the lists of papers that have been read by the users. Therefore, a paper is never recommended to the user who has already read it.

2.2. User profiles

As described above, we consider the data browsed by the users as a good source to derive profiles. However, a variety of descriptions about the data can be utilized for profile contents. Considering the data of the OTS, several fields are candidates to be used, such as titles, URLs, author names, keywords, categories, and abstracts. In this paper, we assume that the category of a paper motivates the user to read it and adopt only the categories of papers to constitute the basic elements of user profiles.

Each time a user enters the OTS, the categories of papers downloaded by the user are recorded. We define a *transaction* as the set of categories recorded during a period that the user uses the OTS. All the categories in a transaction are equally treated. Therefore, the OTS does not demand explicit user interactions. For each user, the categories that appear in his/her transactions are recorded. For each category, the first and the last transactions that contain it are also recorded. In this way, each category will be associated with the number of occurrences since its first transaction, which can be used to compute its *support* (see Section 2.3). For each user, we identify a set of categories that have frequently appeared in the transactions to represent the user interests, e.g., {artificial intelligence,

database research, operating systems}.

Interest profile. Let *support(c)* denote the support of category c. Given a threshold α, the *interest profile* of a user (denoted by I) is defined as follows:

$$I = \{c \mid \text{support}(c) \geq \alpha\} \cdots\cdots (1)$$

On the other hand, we also consider the user behaviors that are related to more than one category. We define the *N-category set* as the set of N categories recorded in the same transaction. For example, a transaction with three categories ABC produces three 2-category sets AB, BC and AC and one 3-category set ABC. For simplicity, in this paper we only consider the 2-category sets and denote the one with two categories, say c and d as [c,d]. Note that our approach can be easily extended to the N-category sets where N is larger than 2.

For each user on the OTS, the 2-category sets that appear in his/her transactions are recorded. For each 2-category set, the first and the last transactions that contain it are recorded. Each 2-category set is again associated with the number of occurrences since its first transaction. For each user, we identify a set of 2-category sets that have frequently appeared in the transactions to represent the user behaviors, e.g., {[artificial intelligence, database research], [database research, operating systems]}.

Behavior profile. Let *support([c,d])* denote the support of 2-category set [c,d]. Given a threshold β, the *behavior profile* of a user (denoted by B) is defined as follows:

$$B = \{[c,d] \mid \text{support}([c,d]) \geq \beta\} \cdots\cdots (2)$$

The values of the two thresholds α and β in Equation (1) and (2) can be different. Therefore, the categories in the behavior profile are not necessarily included in the interest profile. In the rest of this paper, we call the categories in the interest profile the *interests* and the 2-category sets in the behavior profile the *behaviors*.

2.3. Incremental mining

In the following, we introduce our methods to derive the user profiles from the transactions, respectively. Due to the accumulation of transactions, the profile derivation may incur high overheads if we adopt the traditional data mining algorithm [1]. Therefore, we propose new methods instead, which derive the profiles incrementally.

Interest table. The core of our method is the *interest table*, which is built for each user to keep the categories in the transactions. An interest table consists of four columns, while each row of the table refers to a category. An example with four transactions is shown in Figure 2. The Category column lists the categories that appear in the transactions. The First and Last columns record the first and the last transactions that each category appears, respectively. Finally, the Count column keeps the number of occurrences for each category since its first transaction. Take the four transactions in Figure 2 as an example. Since category c appears in T1, T2 and T4, its count has a value 3. Moreover, the first and the last transactions of category c are T1 and T4, respectively.

Transactions	Category	First	Last	Count	Support
T1 {a, c, e}	A	T1	T1	1	N/A
T2 {b, c, e, f}	B	T2	T4	2	67%
T3 {d, e, f}	C	T1	T4	3	75%
T4 {b, c, d}	D	T3	T4	2	100%
	E	T1	T3	3	75%
	F	T2	T3	2	67%

Figure 2: Four transactions and the interest table

From the interest table, we compute the *support* of a category as follows (where c is a category and T is the current transaction):

$$\text{support}(c) = \frac{\text{Count}(c)}{T - \text{First}(c) + 1} \cdots\cdots (3)$$

This formula indicates that only the transactions after the first transaction of a category are considered in the support measure. In other words, we ignore the effects of the transactions before the first transaction of a category. Consider the scenario that a category first appears in T91 and then continually shows up from T92 to T100. By the traditional approach, its support is small (10%). However, our formula computes the support at 100% to reveal that this category appears frequently in the recent transactions.

By Equation (3), when a category first appears, its support will be always 100%, which is not reasonable. Therefore, we define a threshold called the *minimal count* (denoted by γ) to filter out the categories with very small counts. On the other hand, we observe that the support of a category can be very low if its first transaction is very far from the current transaction. Consider the scenario that a category first appears in T1, disappears from T2 to T91, and continually shows up from T92 to T100. By Equation (3), its support is just 10%. However, the fact that this category appears frequently in the recent transactions may imply that the user is getting interested in it recently. Therefore, we define another threshold called the *expired time* (denoted by λ) to restrict the interval between the last and the current transactions. When this interval exceeds λ both the First and the Last columns of the category are changed to the current transaction.

The rightmost column of Figure 2 lists the support of each category. In this example, the thresholds α, γ, and λ are set to 75%, 2, and 4, respectively. We do not compute the support of category a since its count is less than γ. As a result, the interest profile {c, d, e} is derived.

When a new transaction T5 arrives, we recompute the supports of the categories that appear in T5 or the interest profile to update the interest table. As shown in Figure 3, we recompute the supports of all the categories except for category a. By α, we get the new interest profile {b, c, e, f}.

Note that category d is removed from the interest profile because its new support is lower than α.

Transactions	Category	First	Last	Count	Support
T1 {a, c, e}	a	T1	T1	1	N/A
T2 {b, c, e, f}	b	T2	T5	3	75%
T3 {d, e, f}	c	T1	T5	4	80%
T4 {b, c, d}	d	T3	T4	2	67%
T5 {b, c, e, f, g}	e	T1	T5	4	80%
	f	T2	T5	3	75%
	g	T5	T5	1	N/A

Figure 3: The interest table after T5 arrives

Behavior Table. To derive the behavior profile, we also build the *behavior table* for each user to keep the 2-category sets in the transactions. The behavior table is similar to the interest table except that each row refers to a 2-category set and the Category column is changed to the 2-Category Set column. Figure 4 shows the behavior table for the transactions in Figure 2.

2-Category Set	First	Last	Count	Support
[a,c]	T1	T1	1	N/A
[a,e]	T1	T1	1	N/A
[b,c]	T2	T4	2	67%
[b,d]	T4	T4	1	N/A
[b,e]	T2	T2	1	N/A
[b,f]	T2	T2	1	N/A
[c,d]	T4	T4	1	N/A
[c,e]	T1	T2	2	50%
[c,f]	T2	T2	1	N/A
[d,e]	T3	T3	1	N/A
[d,f]	T3	T3	1	N/A
[e,f]	T2	T3	2	67%

Figure 4: The behavior table for Figure 2

Similarly, we compute the supports of the 2-category sets by Equation (3). To avoid the overestimation and underestimation, we also apply the same thresholds γ and λ to the derivation of behavior profile. In Figure 4, only the supports of three 2-category sets (i.e., [b,c], [c,e], [e,f]) are computed since all others' counts are less than γ. If β is set to 60%, the behavior profile {[b,c], [e,f]} is derived.

2.4. Types of user profiles

Based on the user profiles derived above, we define the *I-B diagram* for each user to represent his/her interests and behaviors. Figure 5 shows an example of the I-B diagram.

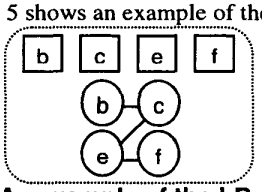

Figure 5: An example of the I-B diagram

In the I-B diagram, each rectangle refers to a category in the interest profile, while each line connecting two circles stands for a 2-category set in the behavior profile. As Figure 5 depicts, the interest profile contains b, c, e, and f, while the behavior profile includes [b,c], [c,e], and [e,f]. Based on the I-B diagram, we classify the user profiles into six types, where each indicates a type of user interests and behaviors. Recall that I and B refer to the interest and behavior profiles, respectively. We use $\underline{B}$ to denote the sets of all the categories that appear in B. Figure 6 shows the examples for the six types of profiles.

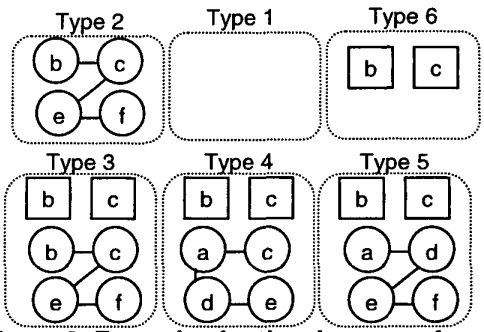

Figure 6: Examples for the six types of profiles

Type 1: ($I = \emptyset$) and ($\underline{B} = \emptyset$). We consider the users who read papers without regularity as the first type. For instance, a graduate student studies the papers in various categories to look for research topics. Because this type of users seldom focuses on the same categories, they are associated with empty profiles.

Type 2: ($I = \emptyset$) and ($\underline{B} \neq \emptyset$). Users of this type do not prefer any category, but often browse the papers of two categories in the same transactions. For instance, a student studies the relation between Artificial Intelligence and Database, so s/he repeatedly browses the papers of these categories during a short period of time. As a result, only the behavior profile is derived.

Type 3: ($I \neq \emptyset$) and ($\underline{B} \neq \emptyset$) and ($I \subseteq \underline{B}$ or $\underline{B} \subseteq I$). Users of this type are interested in particular categories and they do not browse the papers in other categories. For example, a student whose interests are Artificial Intelligence and Database focuses on the two categories. Sometimes, s/he repeatedly browses the papers in other categories together with these categories in the same transactions. In this case, all the categories in the interest profile appear in the behavior profile.

Type 4: ($I \neq \emptyset$) and ($\underline{B} \neq \emptyset$) and ($I \not\subset \underline{B}$) and ($\underline{B} \not\subset I$) and ($I \cap \underline{B} \neq \emptyset$). This type is similar to Type 3 except that some categories in the behavior profile do not appear in the interest profile, and vice versa. Considering the student in the previous example, this time s/he seldom browses the papers in other categories together with Database in the same transactions. Therefore, only some categories (e.g., Artificial Intelligence) are the overlap between interest and behavior profiles.

Type 5: ($I \neq \varnothing$) and ($\underline{B} \neq \varnothing$) and ($I \cap \underline{B} = \varnothing$). This type is also similar to Type 3 and 4 except that all the categories in the behavior profile do not appear in the interest profile, and vice versa. Considering the same student with two interests again, this time s/he repeatedly browses the papers of Operating Systems and Neural Networks in the same transactions to prepare a report. Therefore, the overlap between the interest and behavior profiles is empty.

Type 6: ($I \neq \varnothing$) and ($\underline{B} = \varnothing$). Users of the last type prefer particular categories but seldom browse the papers of two categories in the same transactions. Finally, only the interest profile is derived.

3. Clustering user interests and behaviors

3.1. Clustering framework

To provide the collaborative filtering service, we design a two-phase clustering method. At first, we assign each user to one of the six types based on the I-B diagram. In this way, the users who own similar profiles are grouped together. Next, for each type, we classify the users into finer clusters by a distance measure. The framework of our clustering method is demonstrated in Figure 7.

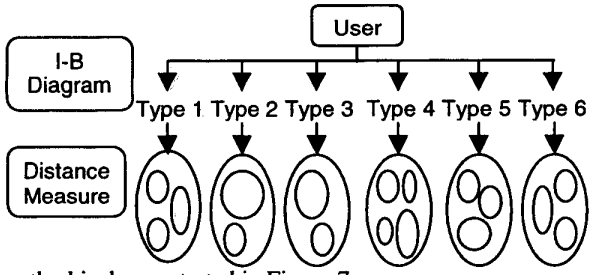

Figure 7: The framework of our clustering method

3.2. Distance measure

After assigning the user to a type, we store the I-B diagram as a square matrix called the *I-B matrix*. Each dimension of the I-B matrix corresponds to one of the categories that appear in the I-B diagrams. The diagonal of the I-B matrix refers to the categories in the interest profile, while the remaining parts stand for the 2-category sets in the behavior profile. Each cell of the I-B matrix is associated with a number 1 or 0 to indicate whether the corresponding interest or behavior exists or not.

Take the I-B diagrams in Figure 8 as an example. Both users have two interests and three behaviors. The corresponding I-B matrixes are shown at the right-hand side of Figure 8. Due to the symmetry of a matrix, we utilize the right-upper triangle of the I-B matrix for keeping the behaviors. To measure the distance between user profiles, we first transform the I-B matrix into a vector named the *I-B vector*. The I-B vector is a bit string composed of all the 0's and 1's in the I-B matrix. In our approach, we collect these numbers from the I-B matrix row by row.

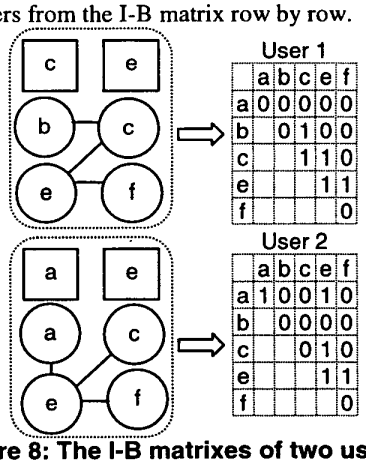

Figure 8: The I-B matrixes of two users

Considering the I-B matrix of user 1 in Figure 8, the numbers on the five rows are 00000, 0100, 110, 11, and 0, respectively. By concatenation, we obtain the I-B vector 000000100110110. In this way, we also derive the I-B vector 100100000010110 from the I-B matrix of user 2. At last, we compute the *distance* between two I-B vectors as follows (where A_i (B_i) is the i-th element of vector A (B)):

$$\text{distance}(A,B) = \sqrt{\sum_{i=1}^{n} \left(A_i - B_i\right)} \cdots\cdots (4)$$

Notice that the number of dimension n is determined by the number of categories that appear in the I-B diagrams for all users. Moreover, the computation in the radical sign corresponds to the counting of differences between the given vectors. For the previous example, the number of differences between two I-B vectors is 4. By Equation (4), we compute the distance at 2.

3.3. Dynamic clustering

In the following, we introduce our methods to apply the distance measure on clustering. Due to the dynamic nature of web users, the clustering will incur high overheads if we simply reclassify the I-B vectors as needed. Therefore, we propose a method to cluster the I-B vectors dynamically.

The core of our method is the *centroid*, representing the Euclidean mean of all the I-B vectors in a cluster. To classify an I-B vector, we compute the distances between this vector and the centroids instead of all the I-B vectors. As shown in Figure 7, we start the second phase after assigning the user to one of the six types. At this moment, we have two cases to continue the clustering.

- ✧ *No cluster existed*: The vector to be classified forms a new cluster and also the centroid of the new cluster.
- ✧ *Some clusters existed*: We compute the distance between

the vector to be classified and the centroid of each cluster and identify the cluster whose centroid leads to a minimal distance. According to a threshold δ, we have two alternatives to classify this vector:

1. If the minimal distance is not larger than δ, the vector is assigned to the corresponding cluster. Moreover, its centroid is also updated by considering the new vector.
2. If the minimal distance is larger than δ, the vector itself forms a cluster and also the centroid of the new cluster.

There are three situations we need to update the user clusters. When a new user enters our system, we start the clustering process as described above. On the other hand, if a user does not enter our system for a very long time, the OTS will remove his/her profiles from the user clusters. In this case, the centroid of the corresponding cluster is recomputed. In addition, the user who is already clustered may change interests or behaviors. Similarly, the centroids of all the related clusters are recomputed.

4. Personalized recommendation

4.1. Recommendation of interesting papers

As depicted in Figure 1, the data wrapper continually collects the bibliographic data from the Web to enlarge the index database. Therefore, we assume that the users are not willing to read all the papers on the OTS and therefore look forward to the recommendation. When a paper is read at the first time, the OTS will recommend it to all the users whose interest profile contains the same category of this paper. Recall that the OTS does not recommend the paper to the user who has read it. In this way, the users will get the new papers of the categories they are interested.

4.2. Recommendation from the users with similar interests

For each user, it is reasonable to recommend the papers read by the users in the same cluster. In this paper, we consider two ways to realize this idea. One is based on the users with similar interests, while the other is based on the users with similar behaviors (Section 4.3). When a user issues a request for recommendation, the OTS will identify the overlaps between his/her interest profile and the others' in the same cluster. As a result, the OTS recommends the papers whose categories are included in these overlaps. The OTS also skips the papers that the user has already read.

Consider the example with three users in Figure 9. The papers are denoted as p1, p2, p3, etc. When user 1 issues a request for recommendation, the OTS first identifies the overlapping interests (i.e., category c for user 2 and category e for user 3) and then recommends only p11 because user 1 has already read p3. Note that p9 read by user 3 will not be recommended because category c is not included in the interest profile of user 3.

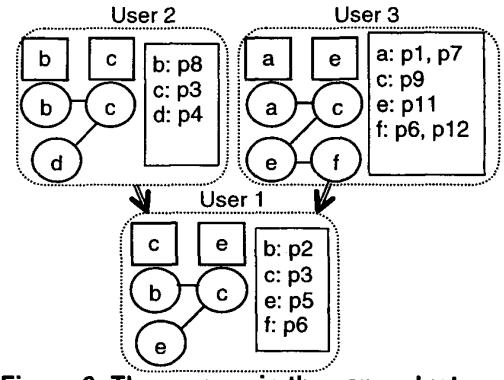

Figure 9: Three users in the same cluster

4.3. Recommendations from the users with similar behaviors

This recommendation is based on the papers read by the users with similar behaviors. When a user issues a request for recommendation, the OTS will identify the overlaps between his/her behavior profile and the others' in the same cluster. As a result, the OTS recommends the papers whose categories are included in these overlaps. Similarly, the OTS will skip the papers that the user has already read.

Considering the previous example, the OTS identifies the overlapping behaviors (i.e., [b,c] for user 2 and [c,e] for user 3) and recommends p8 (from user 2), p9 and p11.

5. Performance evaluation

5.1. Experimental settings

Our experiments are based on the OTS and real data/users. For the experiments, we collect and classify 230 papers into 14 field. Each field is further divided into several categories. The bibliographic data include titles, URLs, author names, keywords, categories, and abstracts, which provide users hints whether they should download the papers or not.

We invite 13 students to do the experiments. When a user enters the OTS, s/he can arbitrarily browse the bibliographic data by categories. Whenever the user downloads a paper, the OTS will record the information in order to derive the user profile. On the other hand, the user can also issue requests for recommendation. According to the papers recommended, the user has to explicitly respond to the OTS with his/her opinions on them. Three grades, i.e., good, moderate, and bad, are used as the responses. If the user does not respond to the OTS, the corresponding responses are set to moderate.

Based on the responses collected from the experiments, we define three measures to evaluate the performance of

our approach. Let N_g, N_m, and N_b denote the number of good, moderate, and bad responses, respectively. N is the total number of recommendations. We define the *hit ratio*, the *average ratio* and the *miss ratio* as follows:

$$\text{hit ratio} = \frac{N_g}{N}, \text{average ratio} = \frac{N_m}{N}, \text{miss ratio} = \frac{N_b}{N} \quad \cdots\cdots(5)$$

5.2. Experimental results

For clarity, we denote the recommendation services as R1, R2, and R3, respectively. Since the user clusters have great impacts on the performance of R2 and R3, we first make experiments on different settings of the threshold δ. After that, the comparisons of three recommendation services are made. Based on our previous observations, the values of thresholds α and β are set to constants 0.4 and 0.25, respectively.

In our experiments, threshold δ is a variable parameter whose values range from 1.6 to 4.8. Table 1 presents the clustered results for different δ's. In the following, we denote the users as U1, U2, etc. C_i's stands for the clusters.

Table 1: The clustered results for different δ's

Cluster	δ=1.6	δ=2.4	δ=3.2	δ=4.0	δ=4.8
C1	U1	U1	U1,U2	U1-U3	U1-U11
C2	U2	U2			
C3	U3	U3	U3		
C4	U4	U4,U5	U4,U5	U4-U7	
C5	U5				
C6	U6	U6,U7	U6,U7		
C7	U7				
C8	U8	U8	U8	U8,U9	
C9	U9	U9	U9		
C10	U10	U10	U10	U10	
C11	U11	U11	U11	U11	
C12	U12	U12	U12	U12	U12,U13
C13	U13	U13	U13	U13	

From Table 1, we observe that each user forms a cluster when δ is set to 1.6. When the value of δ is tuned to 2.4, some clusters are merged (e.g., U5 is merged with U4). When δ is set to 3.2, the clustered results except for U2 are the same (U2 is merged with U1). We continue to enlarge the value of δ and observe that more users are put together into a cluster. Finally, we obtain two clusters when δ is set to 4.8.

In general, with the growth of threshold δ, the average number of users in a single cluster grows but the number of clusters decreases. In the experiments, we compute the hit ratios for different δ's under R2 and R3 respectively. Table 2 shows the experimental results from R2. In this table, as the threshold δ grows, so the hit ratios for most of the users increase. The increase of the hit ratios results from the enlargement of user clusters. Take U7 in Table 2 as an example. The hit ratio of U7 increases as the threshold δ rises from 3.2 to 4. By detailed checks, we find that the users who are added into this cluster (i.e., U4 and U5) have positive effects on the results of recommendation. It confirms that our system performs well if the users in the same clusters have similar interests and behaviors.

Table 2: The hit ratios for different δ's from R2

R2	δ=2.4	δ=3.2	δ=4.0	δ=4.8
U1		100%	100%	50%
U2		33%	25%	40%
U3				100%
U4	50%	50%	52%	42%
U5	67%	67%	80%	31%
U6	50%	50%	20%	43%
U7	19%	19%	24%	50%
U8				38%
U9			11%	17%
U10				42%
U11				21%
U12				50%
U13				0%

From the previous experiments, we choose 3.2 as the value of threshold δ. Table 3 presents the averages of all the measures for the three kinds of services. For the hit ratio, R2 achieves the highest value, indicating that the recommendation service based on user interests performs better than the one based on user behaviors. On the other hand, R3 gets the lowest hit ratio and the highest miss ratio. Therefore, the recommendation service based on partially matched behaviors may not be a good choice. In addition, R1 gets the lowest miss ratio because it limits the service to the users whose interests contain the categories of the papers recommended.

Table 3: The averages of all the measures

	Hit Ratio	Average Ratio	Miss Ratio	Acceptance Rate
R1	48.5%	46.1%	5.4%	94.6%
R2	53.1%	35.4%	11.5%	88.5%
R3	48.7%	22.8%	28.5%	71.5%

6. Conclusion

In this paper, we define user interests and behaviors on the Web based on the browsing history. A new method for mining user interests and behaviors is proposed. Our method emphasizes the adaptability to the large amount and dynamic nature of web data. Moreover, we define six types of user profiles and describe a two-phase clustering method with the distance measure for the user profiles.

According to the results of user clustering, three kinds of recommendation services are implemented to build the OTS. Finally, we make experiments on real data/users to testify the effectiveness of our approach. The results show that the average acceptance rates of these services range from 71.5% to 94.6%. Based on the progress we have made, more issues are revealed. We list three as follows:

✧ *The extent of the user profiles*: How much information is

enough to constitute the user profiles? In this paper, we consider only the categories of papers. However, the others (e.g., keywords) may also imply clues to the users' interests and behaviors. Therefore, an extension of the profile contents may benefit to our approach.

- ✧ *The effects of N-category sets* (N>2): How will the N-category sets impact on the user profiles? In this paper, we focus on the 2-category sets and find that they bring positive effects. We wonder what the N-category sets will bring to recommendation services. Intuitively, the N-category sets represent more complex behaviors. Therefore, they will describe the user behaviors in more details. In this case, an efficient algorithm for mining all the N-category sets is needed.
- ✧ *The settings of the thresholds*: All the thresholds α, β, γ, and λ have great impacts on the derivation of user profiles. Therefore, a procedure that helps the OTS dynamically make these decisions is required. On the other hand, for the recommendation services based on user behaviors, we will apply the confidence measure in data mining field to assure the effectiveness of the behavior profiles.

Reference

[1] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," *Proceedings of VLDB Conference*, pp.487-499, 1994.

[2] M. Balabanovic and Y. Shoham, "Fab: Content-based Collaborative Filtering Recommendation," *Communications of the ACM*, 40(3): 66-72, March 1997.

[3] N. J. Belkin and W. B. Croft, "Information Filtering and Information Retrieval: Two Sides of the Same Coin?," *Communications of the ACM*, 35(12): 29-38, 1992.

[4] M. S. Chen, J. W. Han, and P. S. Yu, "Data Mining: An Overview from Database Perspective," *IEEE Transactions on Knowledge and Data Engineering*, 5: 926-938, 1996.

[5] M. Ester, H. Kriegel, J. Sander, M. Wimmer, and X. Xu, "Incremental Clustering for Mining in a Data Warehousing Environment," *Proceedings of VLDB Conference*, 2000.

[6] D. Goldberg, D. Nichols, B. M. Oki and D. Terry, "Using Collaborative Filtering to Weave an Information Tapestry," *Communications of the ACM*, 35(12): 61-70, 1992.

[7] H. Hirsh, C. Basu, and B. D. Davison, "Learning to Personalize," *Communications of the ACM*, 43(8): 102-106, August 2000.

[8] T. Joachims, D. Freitag, and T. Mitchell, "WebWatcher: A Tour Guide for the World Wide Web," *Proceedings of Joint Conference on Artificial Intelligence*, pp. 770-775, 1997.

[9] P. B. Kantor, E. Boros, and et al., "Capturing Human Intelligence in the Net," *Communications of the ACM*, 43(8): 112-115, August 2000.

[10] J. A. Konstan, B. N. Miller, D. Maltz, and et al., "Grouplens: Applying Collaborative Filtering to Usenet News," *Communications of the ACM*, 40(3): 77-87, March 1997.

[11] H. Lieberman, "Letizia: An Agent that Assists Web Browsing," *Proceedings of International Joint Conference on Artificial Intelligence*, pp. 924-929, August 1995.

[12] J. Mostafa, S. Mukhopadhyay, W. Lam, and M. Palakal, "A Multilevel Approach to Intelligent Information Filtering: Model, System, and Evaluation," *ACM Transactions on Information Systems*, 15(4): 368-399, October 1997.

[13] A. Moukas, "Amalthaea: Information Discovery and Filtering Using a Multi-agent Evolving Ecosystem," *International Journal of Applied Artificial Intelligence*, 1997.

[14] M. Pazzani and D. Billsus, "Learning and Revising User Profiles: The Identification of Interesting Web Sites," *Journals of Machine Learning*, 27: 313-331, 1997.

[15] M. Perkowitz and O. Etzioni, "Adaptive Web Sites," *Communications of the ACM*, 43(8): 152-158, August 2000.

[16] B. Shapira, U. Hanani, A. Raveh, and P. Shoval, "Information Filtering: A New Two-Phase Model Using Stereotypic User Profiling," *International Journal of Intelligent Information Systems*, 8: 155-165, 1997.

[17] B. Smith and P. Cotter, "A Personalized Television Listings Service," *Communications of the ACM*, 43(8): 107-111, August 2000.

[18] M. Spiliopoulou, "Web Usage Mining for Web Site Evaluation," *Communications of the ACM*, 43(8): 127-134, August 2000.

[19] I. H. Witten, A. Moffat, and T. C. Bell, *Managing Gigabytes: Compressing and Indexing Documents and Images*, Van Nostrand Reinhold Publisher, 1994.

[20] Y. H. Wu and A. L. P. Chen, "Index Structures of User Profiles for Efficient Web Page Filtering Services," *Proceedings of IEEE Conference on Distributed Computing Systems*, pp. 644-651, April 2000.

[21] T. W. Yan, "SIFT--A Tool for Wide-Area Information Dissemination," *Proceedings of the USENIX Technical Conference*, pp. 177-186, 1995.

[22] T. W. Yan and H. Garcia-Molina, "Index Structures for Selective Dissemination of Information under the Boolean Model," *ACM Transactions on Database Systems*, 19(2): 332-364, 1994.

[23] T. Zhang, R. Ramakrishnan, and M. Linvy, "BIRCH: An Efficient Data Clustering Method for Very Large Databases," *Proceedings of ACM SGIMOD Conference*, pp. 103-114, 1996.