Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 145636, 11 pages
http://dx.doi.org/10.1155/2015/145636

Research Article

A Hybrid Recommender System Based on User-Recommender Interaction

Heng-Ru Zhang,[1] Fan Min,[1] Xu He,[1,2] and Yuan-Yuan Xu[1]

[1]*School of Computer Science, Southwest Petroleum University, Chengdu 610500, China*
[2]*Lab of Granular Computing, Minnan Normal University, Zhangzhou 363000, China*

Correspondence should be addressed to Fan Min; minfanphd@163.com

Received 8 October 2014; Revised 19 December 2014; Accepted 19 December 2014

Academic Editor: Seungik Baek

Recommender systems are used to make recommendations about products, information, or services for users. Most existing recommender systems implicitly assume one particular type of user behavior. However, they seldom consider user-recommender interactive scenarios in real-world environments. In this paper, we propose a hybrid recommender system based on user-recommender interaction and evaluate its performance with recall and diversity metrics. First, we define the user-recommender interaction. The recommender system accepts user request, recommends N items to the user, and records user choice. If some of these items favor the user, she will select one to browse and continue to use recommender system, until none of the recommended items favors her. Second, we propose a hybrid recommender system combining random and k-nearest neighbor algorithms. Third, we redefine the recall and diversity metrics based on the new scenario to evaluate the recommender system. Experiments results on the well-known MovieLens dataset show that the hybrid algorithm is more effective than nonhybrid ones.

1. Introduction

Recommender systems (RSs) have been extensively studied to present items, such as movies, music, and books [1–3]. There are three main types of recommendation methods: memory-based, model-based, and hybrid [4]. Memory-based methods [4, 5] usually use similarity metrics to obtain the distance between two users or two items. Model-based methods [4, 6, 7] use demographic, content, or aggregated information to create a model that generates recommendations. Hybrid methods [8–10] combine different types of recommenders to gain better performance [11].

Most existing RSs implicitly assume one particular type of user behavior. These behaviors include browsing, rating, and sequence-independent manner. The browsing behavior [12, 13] is that the user only specifies which items are browsed. The rating behavior [14, 15] is that the user specifies the score to items. The sequence-independent manner [16] is not related to the sequence of recommended items. However, these RSs seldom consider user-recommender interactive scenarios in real-world environments.

In this paper, we propose a hybrid recommender system based on user-recommender interaction. The interaction provides a framework for user-recommender behavior and recommender algorithms. We employ the random algorithm to deal with cold-start problem in the face of data sparsity. The hybrid algorithm is applied to incremental recommendation when the data reaches a certain degree. The sequence of the user's choice affects subsequent recommendation in the process of user-recommender interaction. Consequently, the interaction forms an active learning scenario.

This work has four aspects. First, we define the user-recommender interaction. The recommender behavior has three steps: (1) accept user request, (2) recommend N items to user with certain algorithm, and (3) record user choice for further recommendation. The user behavior has three steps: (1) get recommendations of N items, (2) compare with her interest, and (3) select one to browse. Second, we build an interactive hybrid RS based on active and incremental learning. In the initialization phase, no record of recommendation exists. The items are recommended through a random algorithm. In the transition stage, the data is very sparse.

The items are recommended based on hybrid algorithm of both random and k-nearest neighbors (kNN). In the stable stage, the scale of data reaches a certain degree. The items are recommended mainly through the kNN algorithm. Third, we redefine the recall and diversity metrics [17] under the new scenario to evaluate the quality of the RS. The number of recommendations in each turn essentially serves as a constraint. Fourth, we test the random, kNN, and our hybrid algorithm with the new metrics.

The contribution of the paper is threefold. (1) We define the interactive behavior between user and recommender. The new scenario has practical significance because it is closer to the real world. (2) The common accuracy based on error metrics (such as RMSE, MAE [18]) is not a natural fit for evaluating the interactive RS. We redefine two metrics of the recall and diversity [19] based on the new scenario to evaluate our RS. (3) We test the hybrid algorithm in the process of interactive recommender.

Experiments on the well-known MovieLens dataset (http://www.movielens.org/) show that (1) when the ratio of random recommendations is not too big, it has no great influence on the performance; (2) the RS should employ kNN algorithm to recommender as early as possible; (3) the hybrid algorithm performs better than the nonhybrid one.

The rest of the paper is organized as follows. Section 2 makes the assumptions for interactive recommendation scenario. The user and recommender behaviors are defined in our interactive scenario. Section 3 builds a new hybrid recommender system based on the user-recommender interaction. Section 4 presents experimental results on the MovieLens dataset with two metrics of the recall and diversity. The appropriate parameters setting of our hybrid algorithm is discussed in detail. Section 5 briefly presents some of the research literature related to user-recommender interaction, active learning, kNN, hybrid filtering, and granular computing. Finally, concluding remarks are made in Section 6.

2. The Interactive Scenario

In this section, we make the assumptions for interactive recommendation scenario and define the browse user and recommender.

2.1. Binary Relations for Recommender Systems. In this subsection, we revisit the definition of binary relations in information systems [18].

Definition 1. Let $U = \{u_1, u_2, \ldots, u_n\}$ and $M = \{m_1, m_2, \ldots, m_k\}$ be two sets of objects. Any $R \subseteq U \times M$ is a binary relation from U to M.

An example of binary relation is given in Table 1, where $U = \{u_1, u_2, \ldots, u_n\}$ is the set of users and $M = \{m_1, m_2, \ldots, m_k\}$ is the set of movies. A binary relation can be viewed as an information system. However, in order to save space, it is more often stored in the database as a table with two foreign keys.

Table 1: Binary relation.

UID\MID	m_1	m_2	m_3	m_4
u_1	1	1	0	1
u_2	1	0	0	1
u_3	0	1	0	0
u_4	0	1	0	1

2.2. Assumptions. We make the following assumptions to simplify the scenario.

(1) The user interest is deterministic. Given a set of items, the user is interested in a fixed subset. This may be different from the reality where the user interest changes over time or is influenced by the recommendation sequence.

(2) The items set does not change. In slow evolving applications, this might be true. In rapid changing applications such as e-commerce, this is not the case since items are added and removed frequently.

(3) The users set does not change. This is similar to the case of the items set.

(4) The RS starts from the initial state where there is no browse history. In other words, all values of the browse matrix are 0 at the beginning.

(5) A fixed number of items are recommended to the user in each round. Let N be the number of items recommended to a user each time. For example, a fixed number of website links are shown in one web-search page.

For the first assumption, we can define a user interest matrix as follows.

Definition 2. Let $U = \{u_1, u_2, \ldots, u_n\}$ be the set of all users and $T = \{t_1, t_2, \ldots, t_m\}$ the set of all items. The user interest matrix is

$$um_{n\times m}, \tag{1}$$

where

$$um_{i,j} = \begin{cases} 1, & \text{if } u_i \text{ is interested in } t_j; \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

2.3. User and Recommender Behaviors. The user-recommender interaction is a mutual action between the login user and recommender system. The user first logins on the system. The system then returns one or multiple recommended items. The user selects an item as her choice or terminates the interaction.

Users of an RS can be divided into two types according to their feedback. If the user only specifies which items are browsed, she will be called a browse user. If the user specifies the rating to items, she will be called a rate user. We only consider browse user throughout the paper.

The user behavior is illustrated in Figure 1. A browse user logs on the system and gets N recommendations. The user

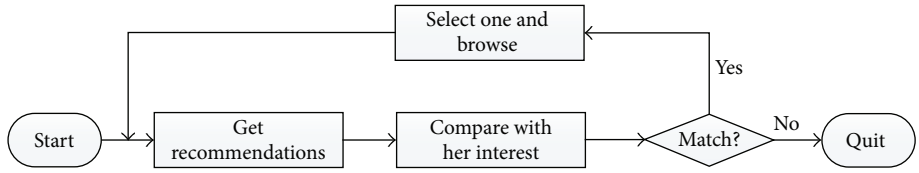


Figure 1: The user behavior.

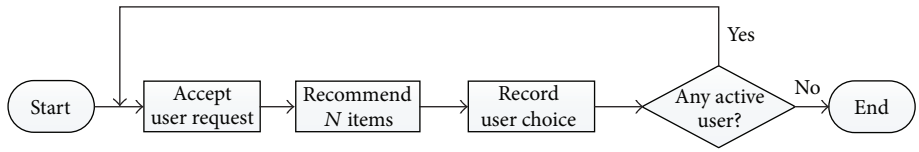


Figure 2: The recommender behavior.

can obtain a candidate browse set according to her interest and the recommendations. We define the candidate browse set as follows.

Definition 3. Let $T_i \subseteq T$ be the interests set of user u_i and $T_r = \{t_{x_1}, t_{x_2}, \ldots, t_{x_N}\}$ the recommended set. $T_c = T_i \cap T_r$ is the candidate browse set.

If $T_c \neq \emptyset$, u_i is interested in some of these items. Let $L_r = (t_{x_1}, t_{x_2}, \ldots, t_{x_N})$ be the recommended list. The user will browse the first match of L_r and continue to use the system. We define the first match as follows.

Definition 4. If $t_{x_j} \in T_c$ and $\forall k \in \{1, 2, \ldots, j-1\}, t_{x_k} \notin T_c, t_{x_j}$ is the first match.

If $T_c = \emptyset$, u_i will quit the system since the recommended items do not match her interest. The user has no patience to continue to use the RS. Since the number of items is limited, the browse user will eventually quit.

We define a *grey list* for improving the effectiveness of recommendations. Based on Definition 4, we consider the list $L_o = (t_{x_1}, t_{x_2}, \ldots, t_{x_{j-1}})$ to be not interesting for u_i. All items of L_o are added into the grey list of u_i and will not be recommended again.

After all users quit the system, we obtain a matrix recording whose items have been browsed by the users. This is called the browse matrix and given by

$$bm_{n\times m}, \tag{3}$$

where

$$bm_{i,j} = \begin{cases} 1, & \text{if } u_i \text{ has browsed } t_j; \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Note that the browse matrix is influenced by not only the recommender system but also the user behavior such as the period between two browse actions.

In most cases, we assume that the user cannot visit the RS again after she quit the system. However, some users may be very patient. They can visit the RS repeatedly. We define the kind of user behavior as *revisit*.

The recommender behavior is illustrated in Figure 2. The RS accepts user login and request and recommends N items to the browse user. If the user has browsed one item, the RS records her choice to the browse matrix. The number of browsed items will increase due to more interactions. This is also related to incremental learning. If the user cannot discover her interests from the N recommendations, she will quit the system. With all users quitting, the system will halt.

3. Hybrid Recommendation for the Interactive Scenario

In this section, we build a new hybrid RS based on the user-recommender interaction. There are three aspects: (1) a new hybrid approach is designed to achieve a tradeoff between the interactive recall and diversity; (2) the model of user-recommender interaction is built based on the new hybrid algorithm; (3) a number of parameters are proposed to adjust our algorithm.

3.1. The Hybrid Recommendation Algorithm. Our hybrid algorithm consists of random and kNN ones. The random algorithm is used to solve cold-start problem and keep the diversity of RS. The kNN algorithm finds k similar users through computing the cosine value. Each neighbor recommends a certain number of items. Let TR be the threshold of kNN recommendations. It determines whether or not to use the kNN algorithm. Based on (4), the number of browsed items by user u_i is $|br(u_i)|$. This is called the kNN switch and given by

$$k\text{NN switch} = \begin{cases} \text{true}, & \text{if } \left|br\left(u_i\right)\right| \geq TR; \\ \text{false}, & \text{otherwise.} \end{cases} \tag{5}$$

The hybrid algorithm is the core of the RS. It has three stages based on the data characteristics of browsed matrix. There are three characteristics in the initialization stage: (1) the browsed matrix is null; (2) the browsed items by the login user are null; (3) the number of items that any user browsed is not more than TR.

There are two characteristics of browsed matrix in the transition stage: (1) browsed matrix is not null and (2) the

number of browsed items over TR is greater than zero, but the number of recommended items is not enough to trigger the kNN algorithm. The kNN algorithm is used to recommend a few items, and the other ones are recommended based on random algorithm.

In the stable stage, the number of browsed items over TR is enough to meet the kNN algorithm to recommend sufficient items. The items are recommended mainly based on kNN algorithm. For the diversity of recommendations, the random algorithm is used to recommend a certain proportion of items.

An incremental hybrid algorithm is given by Algorithm 1. The input includes the user id (u_{id}) and three user-specified parameters. These are the total recommended number (N), the ratio of random recommendations (RT), and TR. It essentially has four steps.

Step 1. In the initialization stage, there is not any history data of the browse matrix. We adopt the random algorithm to deal with the cold-start problem. This step corresponds to Lines 3–6 of the algorithm.

Step 2. TR is used to determine whether or not to use the kNN algorithm. We first count the number N_b of browsed items by u_i. If $N_b \geq TR$, the kNN algorithm is employed. This step corresponds to Lines 8–14 of the algorithm.

Step 3. When the recommender is running smoothly, the random algorithm is used to recommend some new items for exploring new interests. This step corresponds to Lines 15–18 of the algorithm.

Step 4. The kNN algorithm is used to recommend some items. The other items are recommended based on random algorithm in transition and stable stages. This step corresponds to Lines 17–23 of the algorithm.

The output of the algorithm is stored in the memory for Algorithm 2.

3.2. The User-Recommender Interaction. The model of user-recommender interaction is built in our RS. The RS is divided into the user and recommender parts. The user logs in and browses items. The main work is fulfilled by the recommender part.

For each user, the user-recommender interaction is as follows: (1) the user gets N recommendations from the RS; (2) whether or not the user browses an item is based on her own interest list. If the user browses an item, the process continues. Otherwise, she is not interested in these items recommended by RS; she will quit immediately.

For our hybrid RS, the user-recommender interaction is as follows: (1) the RS accepts user request and recommends N items to the user based on hybrid algorithm; (2) if the user has browsed one of the recommendations, the RS records the choice to the browsed matrix; otherwise, the interaction between this user and the RS terminates; (3) all users visit the RS in a random sequence.

This process of the interactive recommender is described in Algorithm 2. The input includes three user-specified parameters N, RT, and TR. It essentially has three steps for each user.

Table 2: The user interest matrix.

UID\MID	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8
u_1	1	1	0	1	1	1	0	1
u_2	0	0	0	1	0	1	0	0
u_3	0	1	0	1	0	1	0	0
u_4	1	0	1	0	1	0	0	1

Step 1. Get N items of recommendation through calling Algorithm 1. This step corresponds to Line 8 of the algorithm.

Step 2. If the user is not interested in recommendation items, the RS refuses to further recommendation. This step corresponds to Line 11 of the algorithm.

Step 3. If the user is interested in recommendation items, the RS records the user's choice and updates the browsed matrix. This step corresponds to Line 13 of the algorithm.

Since the number of users is limited, the user-recommender interaction will eventually terminate. The interactive recall and diversity are then computed. This step corresponds to Lines 18-19 of the algorithm.

3.3. A Running Example. An example of the interest matrix is given in Table 2. There are four users and eight movies. The browse matrix is null in the initial stage. N, RT, TR, and k are assigned as 3, 0.25, 1, and 1, respectively, in the running example.

A running example of hybrid recommendation for the interactive scenario is illustrated in Figure 3. The users login on the RS in random order. We assume that a login sequence is (u_3, u_2, u_1, u_4).

Figure 3(a) depicts the interaction between u_3 and the recommender. Because she is the first login user with no neighbor, the movies are recommended to her based on the random algorithm. The interactive process consists of three rounds. We assume that the recommended list of the first round is (m_5, m_6, m_2). The $\{m_6, m_2\}$ are in her interests set, while the m_5 is not in. The m_6 is browsed by her because it is the first match. The corresponding element of the browse matrix is set from 0 to 1. The m_5 is added into her grey list. The recommended list of the second round is the (m_7, m_8, m_4). The m_4 is browsed by her and the $\{m_7, m_8\}$ are added into her grey list. The recommended list of the third round is the (m_3, m_1, m_2). The m_2 is browsed by her and the $\{m_3, m_1\}$ are added into her grey list. She will quit the RS because no movie can be recommended to her. Finally, she browsed the $\{m_2, m_4, m_6\}$ and her grey list is $(m_1, m_3, m_5, m_7, m_8)$.

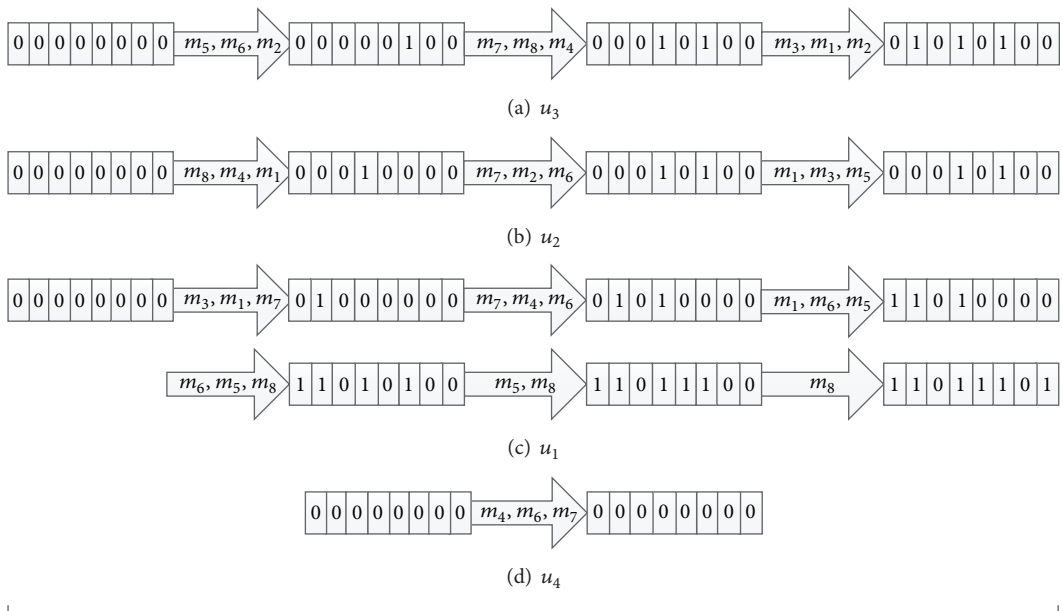
Figure 3(b) depicts the interaction between u_2 and the recommender. The interactive process consists of three rounds. The first round of recommendation is based on random algorithm. The recommended list of the first round is (m_8, m_4, m_1). The m_4 is browsed by her and m_8 is added into her grey list. Because $TR = 1$ and $k = 1$, the u_3 becomes her neighbor. The $\{m_2, m_6\}$ are recommended based on kNN

Input: u_{id}, N, RT, TR
Output: *recommender items*(T_r)
Method: *hybridAlgorithm*
(1) T_b = *the browsed items by* u_{id};
(2) bm = *the browser matrix of all users*;
(3) **if** ($bm = \emptyset$ **or** $T_b = \emptyset$) **then**
(4) T_r = *recommended* N *items by random algorithm;*
(5) **return** T_r;
(6) **end if**
(7) $U = \{u_1, u_2, \ldots, u_n\}$;
(8) $Count = 0$;
(9) **for** each (u_i *in* U) **do**
(10) N_b = *the number of browsed items by* u_i;
(11) **if** ($N_b \geq TR$) **then**
(12) $Count = Count + 1$;
(13) **end if**
(14) **end for**
(15) **if** ($Count \leq 0$) **then**
(16) T_r = *recommended* N *items by random algorithm;*
(17) **return** T_r;
(18) **end if**
(19) $N_k = N * (1 - RT)$;
(20) T_k = *recommended* N_k *items by kNN algorithm;*
(21) $N_{rd} = N - |T_k|$;
(22) T_{rd} = *recommended* N_{rd} *items by random algorithm;*
(23) $T_r = T_k \cup T_{rd}$;
(24) **return** T_r;

Algorithm 1: The hybrid algorithm.

Input: N, RT, TR
Output: *Recall and diversity*
Method: *interactiveRecommender*
(1) $R_f = true$ //R_f *is the flag of successfully recommender;*
(2) $U = \{u_1, u_2, \ldots, u_n\}$;
(3) U' = *an randomized array of* U;
(4) $T = \{t_1, t_2, \ldots, t_m\}$;
(5) **while** (R_f) **do**
(6) $R_f = false$;
(7) **for** each (u_{id} *in* U') **do**
(8) $T_r = hybridAlgorithm(u_{id}, N, RT, TR)$;
(9) T_i = *the interestarray of* u_{id};
(10) **if** ($T_i \cap T_r = \emptyset$) **then**
(11) *the user quits to the RS*;
(12) **else**
(13) *the browsed item is recorded into* $bm(u_{id})$;
(14) $R_f = true$;
(15) **end if**
(16) **end for**
(17) **end while**
(18) $recall = \dfrac{\sum_{i=1}^{n} \sum_{k=1}^{m} bm_{i,k}}{\sum_{i=1}^{n} \sum_{k=1}^{m} um_{i,k}}$;
(19) $BR(u, T) \subseteq T$;
(20) $diversity = \dfrac{|\cup_{u \in U} BR(u, T)|}{|T|}$;
(21) **return** *recall and diversity*;

Algorithm 2: The interactive recommender.



UID	MID							
	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8
u_1	1	1	0	1	1	1	0	1
u_2	0	0	0	1	0	1	0	0
u_3	0	1	0	1	0	1	0	0
u_4	0	0	0	0	0	0	0	0

(e) The browse matrix

Figure 3: A running example.

algorithm and the m_7 is recommended based on random one in the second round of recommendation. The recommended set of the second round is $\{m_2, m_6, m_7\} = \{m_2, m_6\} \cup \{m_7\}$. The recommended list is (m_7, m_2, m_6). The m_6 is browsed by her and the (m_7, m_2) are added into her grey list. Because no movie can be recommended based on kNN algorithm, the third round of recommendation is based on random algorithm. The recommended list is (m_1, m_3, m_5). She will quit the RS in the third round because the recommended movies are not in her interest list.

Figure 3(c) depicts the interaction between u_1 and the recommender. The interactive process consists of six rounds. The first round of recommendation is based on random algorithm. The recommended list of the first round is (m_3, m_1, m_7). The m_2 is browsed by her and m_3 is added into her grey list. The recommendations of the second round are based on kNN and random algorithms. The number of items recommended by the kNN algorithm is $N_k = \lfloor N*(1-RT) \rfloor = \lfloor 3 * (1 - 0.25) \rfloor = 2$. The random recommended number is $N_{rd} = N - N_k = 1$. The u_3 becomes her neighbor and recommends the $\{m_4, m_6\}$. The two movies and the random recommended m_7 constitute the list (m_7, m_4, m_6). The m_4 is browsed by her and m_7 is added into her grey list. The recommendations of the subsequent rounds are based on the random algorithm because her neighbors u_2 and u_3 cannot recommend new movies.

Figure 3(d) depicts the interaction between u_4 and the recommender. The interactive process only includes one round because the recommended list (m_7, m_6, m_4) is not in her interest list.

When there is no active user in the RS, the browse matrix is shown in Figure 3(e). From the viewpoint of the running result, it is more beneficial to recommend if the user's interest is more extensive. Finally, the interactive recall is $ir(U, T) = 11/15 \approx 0.73$. The number of the successfully recommended items is 6; therefore the interactive diversity is $\mathrm{id}(U, T) = 6/8 = 0.75$. The user-recommender interaction forms an active learning scenario because the sequences of the users' login and choice affect subsequent recommendation.

4. Experiments

We design two kinds of experimental schemes. The first kind includes three experimental approaches to find appropriate

parameters setting of our hybrid algorithm. The second kind includes two experimental approaches for comparison of random and hybrid algorithms. Each approach is repeated 10 times with different recommendation items due to random algorithm.

We try to answer the following questions through experimentation.

(1) What is the appropriate proportion of random recommendations?

(2) What is the reasonable threshold of kNN recommendations?

(3) What is the appropriate k in kNN recommendations?

(4) How does N influence the performance of the algorithm?

(5) Does the hybrid algorithm outperform the random one?

4.1. Dataset. We experimented with a well-known MovieLens dataset which is widely used in recommender systems (see, e.g., [10, 20]). The database schema is as follows:

(i) user (*userID*, age, gender, and occupation),

(ii) movie (*movieID*, release-year, and genre),

(iii) rates (*userID, movieID*).

We use the version with 943 users and 1,682 movies. The original rate relation contains the rating of movies with 5 scales, while we only consider whether a user has rated a movie. All users have watched at least one movie, and the dataset consists of approximately 100,000 movies ratings. But rating matrix is still spare because no one has watched more than 45 percent of the total movies, and only the 20 percent of users have watched more than 10 percent of movies.

4.2. Experimental Design. We know which movies the user watches through the original dataset. These movies are assumed as the interest matrix of all users in our RS. The browsed matrix of all users is empty in the initial stage. In the process of incremental recommendations, the new browsed items are recorded into the browsed matrix. Finally we compute the interactive recall and diversity based on the browsed and interest matrix.

We redefine the recall and diversity metrics for measuring the interactive recommendation performance. The ratio of browsed items with respect to items potentially interesting to her will be called the *interactive recall*.

Definition 5. The interactive recall of user u_i on a set of items T through an RS is

$$ir\left(u_i, T\right) = ir\left(bm, um, u_i\right) = \frac{\left|\left\{t_j \in T \mid bm_{i,j} = 1\right\}\right|}{\left|\left\{t_j \in T \mid um_{i,j} = 1\right\}\right|} = \frac{\sum_{k=1}^{m} bm_{i,k}}{\sum_{k=1}^{m} um_{i,k}}. \quad (6)$$

Now we study the interactive recall of the recommender.

Definition 6. The interactive recall of an RS is

$$ir\left(U, T\right) = ir\left(bm, um\right) = \frac{\sum_{i=1}^{n}\sum_{k=1}^{m} bm_{i,k}}{\sum_{i=1}^{n}\sum_{k=1}^{m} um_{i,k}}. \quad (7)$$

Here we observe that the interactive recall is related to the user interest matrix.

Next we study the *interactive diversity* of the recommender. Let $BR(u, T) \subseteq T$ be the set of items browsed by user u in the user-recommender interaction. The set of all items successfully recommended to at least one user is given by

$$BR\left(U, T\right) = \cup_{u \in U} BR\left(u, T\right). \quad (8)$$

The interactive diversity is

$$\mathrm{id}\left(U, T\right) = \frac{\left|BR\left(U, T\right)\right|}{\left|T\right|}. \quad (9)$$

Naturally, higher $\mathrm{id}(U, T)$ indicates more diverse recommendation since more items are recommended and browsed. Our goal is to maximize the interactive recall and diversity. In other words, we expect each user to browse as many items as possible and the recommender to discover as many user interests as possible.

We design two kinds of experiments. The first kind of experiments is to determine appropriate setting of the RT, TR, and k to hybrid algorithm. The second kind of experiments is to compare the interactive recall and diversity of random and hybrid algorithms.

In the first kind of experiments, we assign $N = 20$. Let $RT \in \{0, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$, $TR \in \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 40, 80\}$, and $k \in \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 25, 30, 35, 40, 45, 50, 55, 60\}$. Six experiments are undertaken to answer the questions raised at the beginning of the section one by one. A parameter is allowed to make a change, and the other parameters are set to a fixed value in each experiment. Each experiment is repeated 10 times, and the average recall and diversity are computed.

4.3. Results. In Figure 4(a), we assign $TR = 3$, $k = 45$ and make RT change. When RT changes from 0 to 0.4, the recall basically keeps stable. When $RT \geq 0.4$, the recall begins to decline rapidly.

The parameter settings of Figure 4(b) are the same as Figure 4(a). From the overall trend, the diversity increases with the increasing ratio of random recommendations.

In Figure 4(c), we assign $RT = 0.25$, $k = 45$ and make TR change. The overall trend of the recall is downward with the increase of TR. When TR changes from 1 to 20, the recall decreases rapidly. When $TR \geq 20$, the recall decreases slowly.

The parameter settings of Figure 4(d) are the same as Figure 4(c). The diversity fluctuates between 0.6 and 0.85, and the overall trend remains consistent.

In Figure 4(e), we assign $RT = 0.25$, $TR = 3$ and make k change. When k changes from 1 to 25, the recall is basically linear increase. When k changes from 25 to 45, the recall increases slowly. When $k \geq 45$, the recall keeps stable.

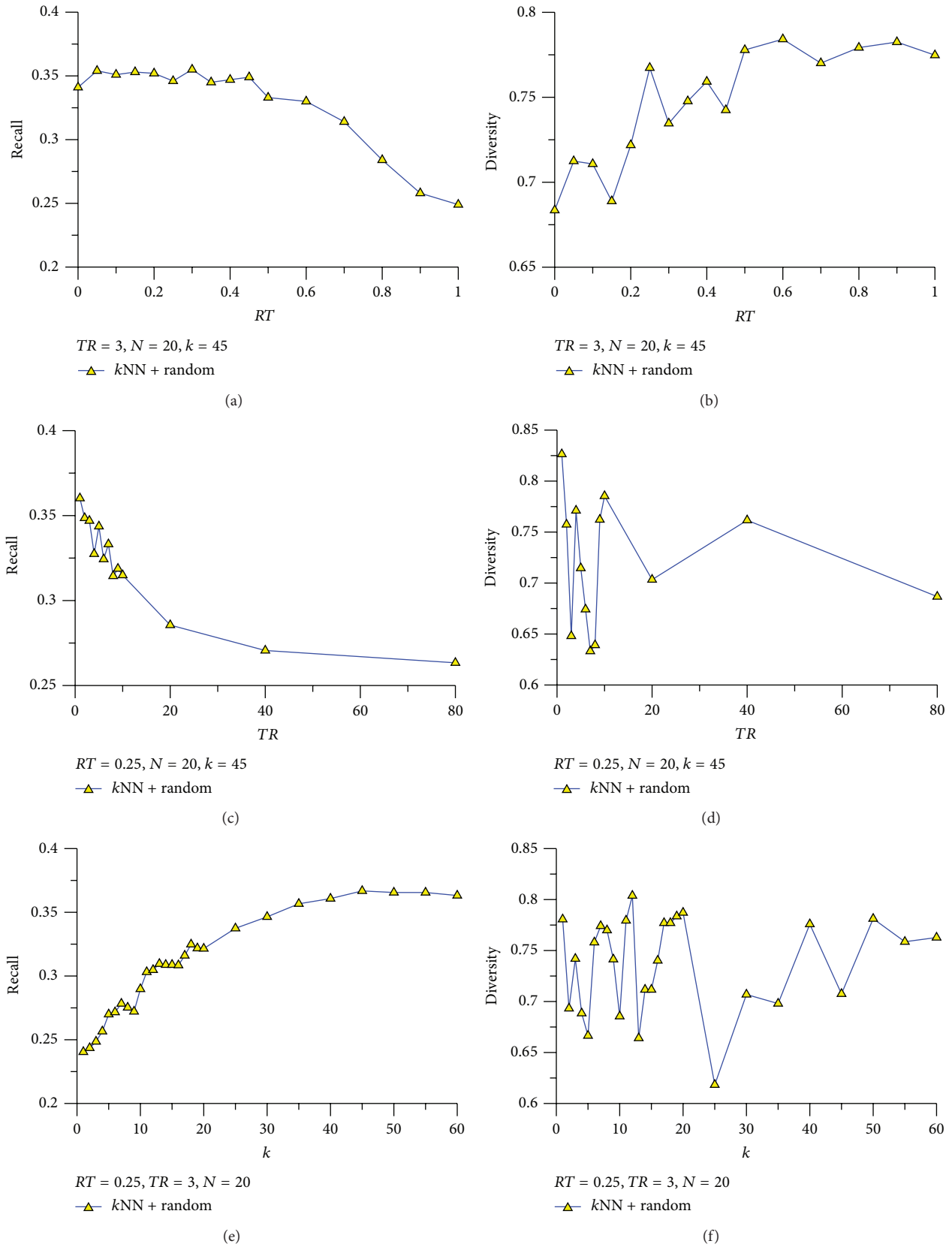


Figure 4: The influence of different parameters.

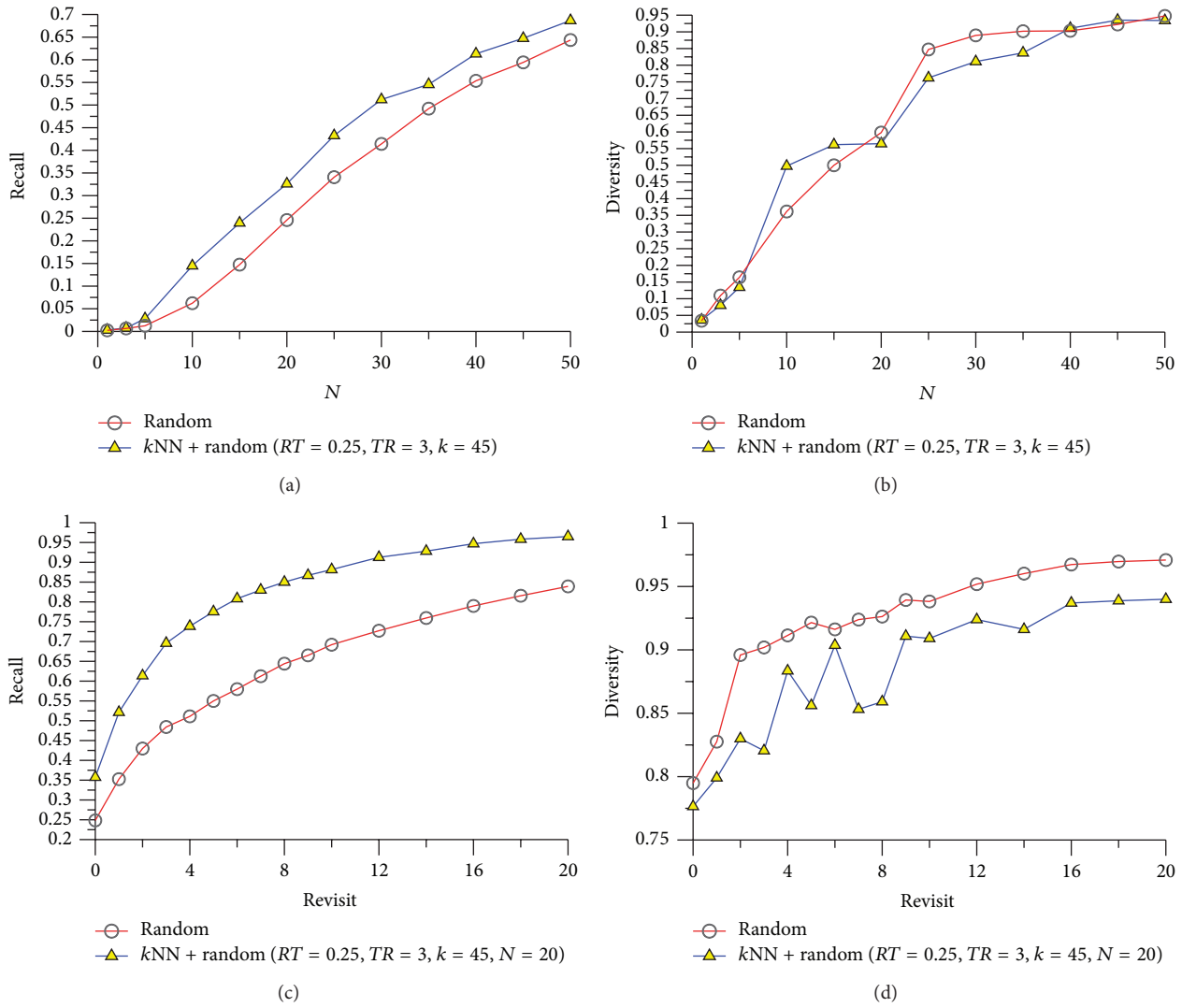


Figure 5: Comparison between random and hybrid recommendation.

The parameter settings of Figure 4(f) are the same as Figure 4(e). The diversity fluctuates between 0.6 and 0.8, and the overall trend remains consistent.

Based on the first kind of experiments, our hybrid algorithm has a better performance when the RT, TR, and k are assigned as 0.25, 3, and 45, respectively.

In the second kind of experiments, we assign $RT = 0.25$, $TR = 3$, and $k = 45$ and make N and *revisit* change.

In Figure 5(a), we assign $N \in \{1, 3, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50\}$. The overall trend of the recall is basically linear increase with the increase of total number. The recall of hybrid algorithm is better than the random one.

The parameter settings of Figure 5(b) are the same as Figure 5(a). When N changes from 1 to 25, the overall trend of the diversity is basically linear increase with the increase of total number. When N changes from 25 to 50, the overall trend of the diversity keeps stable. The diversity of hybrid algorithm is almost all the same as the random one.

In Figure 5(c), we assign $revisit \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 14, 16, 18, 20\}$. The overall trend of the recall is basically increased with the increase of revisit number.

The parameter settings of Figure 5(d) are the same as Figure 5(c). The overall trend of the diversity is basically increased with the increase of revisit number.

The recall of hybrid algorithm is better than the random one; however the diversity is converse.

5. Related Works

In this section we briefly present some of the research literature related to user-recommender interaction, active learning, kNN, hybrid filtering, and granular computing.

User-recommender interaction (URI) [21] is a framework and methodology for analyzing user tasks and recommender algorithms to generate useful recommendation lists. There

are three pillars to URI: (1) the user-recommender interactive interface: receiving user request and getting one recommendation list from a recommender, (2) the recommender algorithm: generating one recommendation list based on user information, and (3) the recommender personality: the user's perception of the recommender over a period of time.

Our interactive recommendation forms an active learning scenario. Active learning guides the acquisition of new knowledge suitable to update timely rated or browsed information [22]. The similar methods have been applied to the active learning [23–25]. The most widely similar method is kNN [26–29]. The kNN is a method for classifying objects based on closest training instances. In the user to user version, the similarity approaches typically compute the similarity based on item ratings by users. The item to item kNN version computes the similarity between two items. The traditional metrics to compute similarity have the cosine, Pearson correlation, mean squared differences, or Euclidean distance [17, 30].

Our hybrid algorithm consists of random and kNN ones. Hybrid filtering [9, 31] combines recommendation components of different types to achieve improved performance and reduce the cold-start problem. There are four main hybridization techniques: weighted, mixed, switching, and feature combination [9, 32]. Hybrid filtering is usually based on bioinspired or probabilistic methods [4] such as random algorithms [20], kNN algorithm [33], and Bayesian networks [34, 35].

The k and N are two kinds of different granules selection. The k is used to select different number of neighbors to participate in the kNN recommendation. The N denotes the number of items recommended to a user each time. We study the impact of different granules on the performance of our algorithm. Granular association rule [36, 37] is an intersection of relational data mining, granular computing, and recommender system. The description of information granules with different attribute-value pairs and different size embodies the essences of granular computing [37]. Granular computing [38–40] is an emerging conceptual and computing paradigm of information processing. It delivers a cohesive framework supporting a formation of information granules and facilitating their processing and has recently been considerable development [37, 41–43].

6. Conclusions

In this paper, we have proposed a hybrid recommender system for the interactive scenario. Through adjusting the recommended parameters and comparing the random and hybrid algorithms, we may draw the following conclusions: (1) the ratio of random recommendations has no great influence on the performance as long as it is not too big (e.g., not more than 0.25), (2) one should employ kNN as early as possible, (3) the neighbors number should be big enough (e.g., 45), (4) the recall is nearly linear increase with respect to the number of recommendations in each round, and (5) the hybrid algorithm is better than the random one.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is in part supported by National Science Foundation of China under Grant nos. 61379089 and 61379049, Seedling Project of Sichuan Province in China under Grant no. 2014-056, Scientific Research Starting Project of SWPU no. 2014QHZ025, and the Natural Science Foundation of Department of Education of Sichuan Province under Grant no. 13ZA0136.

References

[1] L. Duan, W. N. Street, and E. Xu, "Healthcare information systems: data mining methods in the creation of a clinical recommender system," *Enterprise Information Systems*, vol. 5, no. 2, pp. 169–181, 2011.

[2] F. Min and W. Zhu, "Mining top-k granular association rules for recommendation," in *Proceedings of the IEEE IFSA World Congress and NAFIPS Annual Meeting (IFSA/NAFIPS '13)*, pp. 1372–1376, 2013.

[3] X. He, F. Min, and W. Zhu, "A comparative study of discretization approaches for granular association rule mining," in *Proceedings of the 26th IEEE Canadian Conference on Electrical and Computer Engineering*, pp. 725–729, May 2013.

[4] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowledge-Based Systems*, vol. 46, pp. 109–132, 2013.

[5] J. Delgado and N. Ishii, "Memory-based weighted majority prediction," in *Proceedings of the SIGIR Workshop Recommender Systems*, Citeseer, 1999.

[6] H.-R. Zhang, F. Min, and X. He, "Aggregated recommendation through random forests," *The Scientific World Journal*, vol. 2014, Article ID 649596, 11 pages, 2014.

[7] H. R. Zhang, F. Min, and S. S. Wang, "A random forest approach to model-based recommendation," *Journal of Information and Computational Science*, vol. 11, no. 15, pp. 5341–5348, 2014.

[8] D. M. Pennock, E. Horvitz, S. Lawrence, and C. L. Giles, "Collaborative filtering by personality diagnosis: a hybrid memory-and model-based approach," in *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pp. 473–480, 2000.

[9] R. Burke, "Hybrid recommender systems: survey and experiments," *User Modelling and User-Adapted Interaction*, vol. 12, no. 4, pp. 331–370, 2002.

[10] P. Cremonesi, R. Turrin, and F. Airoldi, "Hybrid algorithms for recommending new items," in *Proceedings of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, pp. 33–40, 2011.

[11] T. Tran and R. Cohen, "Hybrid recommender systems for electronic commerce," in *Knowledge-Based Electronic Markets*, Papers from the AAAI Workshop, AAAI Technical Report WS-00-04, pp. 78–83, AAAI Press, 2000.

[12] V. Moscato, A. Picariello, and A. M. Rinaldi, "Towards a user based recommendation strategy for digital ecosystems," *Knowledge-Based Systems*, vol. 37, pp. 165–175, 2013.

[13] S. E. Middleton, N. R. Shadbolt, and D. C. De Roure, "Ontological user profiling in recommender systems," *ACM Transactions on Information Systems*, vol. 22, no. 1, pp. 54–88, 2004.

[14] G. Adomavicius and Y. Kwon, "New recommendation techniques for multicriteria rating systems," *IEEE Intelligent Systems*, vol. 22, no. 3, pp. 48–55, 2007.

[15] T. T. Nguyen, D. Kluver, T.-Y. Wang et al., "Rating support interfaces to improve user experience and recommender accuracy," in *Proceedings of the 7th ACM Conference on Recommender Systems (RecSys '13)*, pp. 149–156, ACM, October 2013.

[16] X. Luo, Y. Xia, and Q. Zhu, "Incremental collaborative filtering recommender based on regularized matrix factorization," *Knowledge-Based Systems*, vol. 27, pp. 271–280, 2012.

[17] G. Adomavicius and 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*, vol. 17, no. 6, pp. 734–749, 2005.

[18] J. S. Armstrong and T. S. Overton, "Estimating nonresponse bias in mail surveys," *Journal of Marketing Research*, vol. 14, no. 3, pp. 396–402, 1977.

[19] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Transactions on Information Systems*, vol. 22, no. 1, pp. 5–53, 2004.

[20] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Methods and metrics for cold-start recommendations," in *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 253–260, 2002.

[21] S. M. McNee, J. Riedl, and J. A. Konstan, "Making recommendations better: an analytic model for human-recommender interaction," in *Proceedings of the Extended Abstracts on Human Factors in Computing (CHI '06)*, pp. 1103–1108, ACM, April 2006.

[22] B. Settles, *Active Learning Literature Survey*, vol. 52, University of Wisconsin, Madison, Wis, USA, 2010.

[23] K. Yu, A. Schwaighofer, V. Tresp, X. Xu, and H.-P. Kriegel, "Probabilistic Memory-Based Collaborative Filtering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 1, pp. 56–69, 2004.

[24] N. Rubens, D. Kaplan, and M. Sugiyama, "Active learning in recommender systems," in *Recommender Systems Handbook*, pp. 735–767, Springer, 2011.

[25] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Advances in Artificial Intelligence*, vol. 2009, Article ID 421425, 19 pages, 2009.

[26] J. Bobadilla, A. Hernando, F. Ortega, and J. Bernal, "A framework for collaborative filtering recommender systems," *Expert Systems with Applications*, vol. 38, no. 12, pp. 14609–14623, 2011.

[27] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th International Conference on World Wide Web*, pp. 285–295, May 2001.

[28] C. H. Lee, Y. H. Kim, and P. K. Rhee, "Web personalization expert with combining collaborative filtering and association rule mining technique," *Expert Systems with Applications*, vol. 21, no. 3, pp. 131–137, 2001.

[29] J. Gemmell, T. Schimoler, M. Ramezani, and B. Mobasher, "Adapting k-nearest neighbor for tag recommendation in folksonomies," in *Proceedings of the 7th Workshop on Intelligent Techniques for Web Personalization and Recommender Systems (ITWP '09)*, p. 528, 2009.

[30] P. B. Kantor, L. Rokach, F. Ricci, and B. Shapira, *Recommender Systems Handbook*, Springer, 2011.

[31] C. Porcel, A. Tejeda-Lorente, M. A. Martínez, and E. Herrera-Viedma, "A hybrid recommender system for the selective dissemination of research resources in a technology transfer office," *Information Sciences*, vol. 184, no. 1, pp. 1–19, 2012.

[32] F. Ullah, G. Sarwar, and S. Lee, "N-screen aware multicriteria hybrid recommender system using weight based subspace clustering," *The Scientific World Journal*, vol. 2014, Article ID 679849, 11 pages, 2014.

[33] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, "Collaborative filtering recommender systems," in *The Adaptive Web*, vol. 4321 of *Lecture Notes in Computer Science*, pp. 291–324, Springer, Berlin, Germany, 2007.

[34] L. M. de Campos, J. M. Fernández-Luna, J. F. Huete, and M. A. Rueda-Morales, "Combining content-based and collaborative recommendations: a hybrid approach based on Bayesian networks," *International Journal of Approximate Reasoning*, vol. 51, no. 7, pp. 785–799, 2010.

[35] Z. Xia, S. Xu, N. Liu, and Z. Zhao, "Hot news recommendation system from heterogeneous websites based on bayesian model," *The Scientific World Journal*, vol. 2014, Article ID 734351, 8 pages, 2014.

[36] X. He, F. Min, and W. Zhu, "Parametric rough sets with application to granular association rule mining," *Mathematical Problems in Engineering*, vol. 2013, Article ID 461363, 13 pages, 2013.

[37] F. Min, Q. Hu, and W. Zhu, "Granular association rules with four subtypes," in *Proceedings of the IEEE International Conference on Granular Computing (GrC '12)*, pp. 353–358, August 2012.

[38] W. Zhu and F.-Y. Wang, "Reduction and axiomization of covering generalized rough sets," *Information Sciences*, vol. 152, no. 1, pp. 217–230, 2003.

[39] D. Liu, T. Li, and D. Ruan, "Probabilistic model criteria with decision-theoretic rough sets," *Information Sciences*, vol. 181, no. 17, pp. 3709–3722, 2011.

[40] Y. Yao and X. Deng, "Quantitative rough sets based on subsethood measures," *Information Sciences*, vol. 267, pp. 306–322, 2014.

[41] Q. Hu, L. Zhang, S. An, D. Zhang, and D. Yu, "On robust fuzzy rough set models," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 4, pp. 636–651, 2012.

[42] W.-Z. Wu and Y. Leung, "Optimal scale selection for multi-scale decision tables," *International Journal of Approximate Reasoning*, vol. 54, no. 8, pp. 1107–1129, 2013.

[43] J. Dai and H. Tian, "Entropy measures and granularity measures for set-valued information systems," *Information Sciences*, vol. 240, pp. 72–82, 2013.