IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 11, 2018, accepted December 28, 2018, date of publication January 9, 2019, date of current version January 29, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2891561

An Adaptive Mechanism for Recommendation Algorithm Ensemble

QI HUANG[1], YUAN-YUAN XU[1], YONG CHEN[2], HENG-RU ZHANG[1], AND FAN MIN[1], (Member, IEEE)

[1]School of Computer Science, Southwest Petroleum University, Chengdu 610500, China
[2]School of Mathematics and Computer Science, Shaanxi University of Technology, Hanzhong 723001, China

Corresponding author: Fan Min (minfanphd@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 41604114, in part by the Shaanxi Provincial Department of Education Special Research Project under Grant 18JK0142, and in part by the Department of Science and Technology of Sichuan Province under Grant 2019JY0513.

ABSTRACT Popular recommendation algorithms such as k-nearest neighbors and Slope One are appropriate for different situations. In this paper, we propose an approach, which will be termed as adaptive mechanism for recommendation algorithm ensemble (AMRE). The AMRE consists of three parts, including a set of agents, a reward-function, and a roulette. First, each agent corresponds to a recommendation algorithm. It also contains a reward value to determine whether it should be retained or replaced. Second, a reward-function is designed to update the reward value according to recommendation results. Wrong recommendations bring punishments, while the right ones bring rewards. Finally, the roulette chooses another agent when the reward value is below a given threshold. The experimental results on MovieLens datasets show that the AMRE outperforms 11 well-known algorithms and two classical ensemble methods on Accuracy, Recall, and F1-measure.

INDEX TERMS Adaptive learning, collaborative filtering, ensemble learning, recommender systems.

I. INTRODUCTION

Collaborative filtering (CF) [1] is one of the most successful and popular algorithms in recommendation systems (RSs). It aims at identifying interesting items for users based on available user preferences. Various CF algorithms have been generated for different situations. Matrix factorization (MF) algorithms [2] mine potential factors of users and items to estimate missing values. They perform well when the rating data are relatively sparse. Slope One algorithms [3] consider the difference between ratings of items for users who rated both. They are appropriate for the situation where items are not updated frequently and items are significantly less than users. k-nearest neighbors (k-NN) algorithms [4] take advantage of users with similar purchase histories or items that tend to be rated similar. They are suitable for the situation [2] where most users have rated many items.

Recently, ensemble learning and adaptive learning have attracted much attention in building RSs. Ensemble learning [5], [6] such as Bagging [7], Boosting [8], and Stacking [9] are previously proposed to handle classification problems. Yang and Yao [10] proposed an ensemble selector for attribute reduction and Yu and Wang [11] proposed a general cluster ensemble framework for both two-way decision clustering and three-way decision. In RSs, Jahrer *et al.* [12] improved Accuracy by blending the traditional Bagging with other approaches. Bar *et al.* [13] also proposed an adaptation of several ensemble techniques for CF to decrease the root mean square error (RMSE). Adaptive learning adjusts and optimizes itself with the least manual tuning [14] during data processing and analysis. It has been successfully used in various fields, such as granular computing [15], [16], sensor drift [17], congestion control [18] and fuzzy control [19]. In RSs, Luo and Yang [20] adopted the gradient descent method adapting to the prediction error. Deng *et al.* [21] proposed a dynamic adaptive CF to allow new data to enter the system at a rapid rate. To the best of our knowledge, there are few works on the combination of adaptive learning and ensemble learning in RSs.

In this paper, we propose an approach called adaptive mechanism for recommendation algorithm ensemble (AMRE). The goal is to obtain better predictions for each item through dynamically switching to an appropriate recommendation algorithm. AMRE consists of three parts, including a set of agents, a reward-function and a roulette.

First, each agent corresponds to a recommendation algorithm and a reward value, which determines whether it should be retained or replaced. The candidate recommendation algorithms include k-NN with different similarity/distance measures and weighted Slope One. The reward value is employed to indicate the goodness of the current agent. Second, based on the correctness of the current recommendation, the reward-function feeds back a reward or a punishment to update the reward value. Finally, the roulette is implemented to switch among agents by a randomly selected strategy. Once the reward value is below a given threshold, the roulette will choose another agent. AMRE is similar to a game with players, a referee and a coach corresponding to the agents, reward-function and roulette, respectively.

AMRE is different from existing ensemble learning and adaptive learning methods on RSs. Most ensemble learning methods (see, e.g., [12], [13]) obtain a final prediction through weighting the predictions of several recommendation algorithms. In contrast, AMRE dynamically selects an appropriate recommendation algorithm for rating prediction. Most adaptive learning methods (see, e.g., [21]) adjust parameters within an individual recommendation algorithm. In contrast, AMRE adaptively switches among different recommendation algorithms.

Experiments are undertaken on two MovieLens datasets (943 users $\times$ 1,682 movies and 706 users $\times$ 8,570 movies) with two screening rate. We adopt the "leave-item-out" scenario to compare AMRE with individual recommendation algorithms as well as ensemble methods. Results show that AMRE outperforms eleven well-known individual CF algorithms and two classical ensemble methods in terms of Accuracy, Recall, and F1-measure.

The rest of paper is organized as follows: Section II presents some preliminary knowledge including rating system, CF, ensemble learning and adaptive learning. Section III describes AMRE in detail. Section IV presents the experimental results on two Movielens datasets. Finally, a summary and the future outlook of our work are presented in Section V.

II. RELATED WORK

This section reviews preliminary knowledge such as rating system, CF, ensemble learning and adaptive learning. Table 1 lists the notations used throughout this paper.

A. RATING SYSTEM

In a recommender system, it can be supposed that each user has rated some items. Let $U = \{u_1, u_2, \ldots, u_n\}$ be the set of users and $T = \{t_1, t_2, ..., t_m\}$ be the set of items. The rating function is [22]

$$R : U \times T \to V, \tag{1}$$

where V is the rating level used by users to evaluate items. For convenience, the rating system is represented with an $n \times m$ rating matrix $R = (r_{i,j})_{n\times m}$, where $r_{i,j} = R(u_i, t_j)$, $1 \le i \le n$, and $1 \le j \le m$. $r_{i,j} = 0$ indicates that user u_i has not rated item t_j. $C_{j,q} = \{u_i | 1 \le i \le n, r_{i,j} \ne 0, r_{i,q} \ne 0\}$ is the set of co-rating users who have rated both item t_j and item t_q.

TABLE 1. Notations.

Notation	Meaning
U	The set of all users, $\|U\| = n$
T	The set of all items, $\|T\| = m$
A	The set of all agents, $\|A\| = M$
R	The rating matrix
P	The prediction matrix
u_i	The i-th user
t_j	The j-th item
a_k	The k-th agent
$r_{i,j}$	The rating of u_i to t_j
$p_{i,j}$	The prediction of u_i to t_j
α	The rating threshold
β	The prediction threshold
w^-	The punishment feedback
w^+	The reward feedback
v_0	The initial reward value
$v_{\min}$	The threshold for switching among agent
$v_{\max}$	The upper bound of reward value
γ	The ratio of initial known ratings, $\gamma \in (0, 1)$
ρ	The ratio of consecutive non-recommendation, $\rho \in (0, 1]$
CN	The upper bound of consecutive non-recommendation, $CN = \rho M$
$C_{j,q}$	The set of co-rating users who have rated both t_j and t_q.
U_j	The set of users who have rated t_j, $\|U_j\| = n_j$
U'	The set of users with known ratings to t_j, $U' \subset U_j$

Table 2 lists an example rating system, where $V = \{1, 2, 3, 4, 5\}$. $C_{2,4} = \{u_3, u_6, u_9\}$ is the set of co-rating users who have rated both item t_2 and item t_4.

TABLE 2. A rating matrix (R).

UID \ TID	t_1	t_2	t_3	t_4	t_5
u_1	3	0	1	3	2
u_2	4	3	0	0	0
u_3	3	1	2	1	2
u_4	0	0	1	1	0
u_5	3	2	0	0	1
u_6	5	1	0	1	2
u_7	0	0	2	1	0
u_8	2	2	5	0	4
u_9	2	1	0	2	2
u_{10}	0	0	2	1	0

B. COLLABORATIVE FILTERING

Most CF algorithms calculate similarity/distance among items for rating prediction. Table 3 summarizes some popular similarity/distance measures for CF.

1) BC [23], [24] is defined to measure similarity between two probability distributions. Let $pd_{j,x}$ and $pd_{q,x}$ be the probability distribution of the rating x in item t_j and item t_q, respectively.
2) Cosine [25] is defined to measure the angle between the rating vectors of two items. Let $\vec{r}_j = (r_{1,j}, r_{2,j}, \ldots, r_{n,j})^T$ and $\vec{r}_q = (r_{1,q}, r_{2,q}, \ldots, r_{n,q})^T$ be the rating vector of item t_j and item t_q, respectively.
3) Pearson [26] is defined to consider the linear correlation between two ratings vectors. Let $\overline{r}_j$ and $\overline{r}_q$ be the average rating of item t_j and item t_q, respectively.
4) CPCC [27] is defined to consider the impact of positive and negative ratings on the basis of Pearson. Let r_{med} be the median rating of V.

TABLE 3. Similarity/distance measures for CF.

Measures	Description
BC [23], [24]	$\mathrm{BC}(t_j, t_q) = \sum_{x \in V} \sqrt{pd_{j,x} \times pd_{q,x}}$
Cosine [25]	$\mathrm{Cosine}(t_j, t_q) = \frac{\vec{r}_j \cdot \vec{r}_q}{\lvert\vec{r}_j\rvert \times \lvert\vec{r}_q\rvert}$
Pearson [26]	$\mathrm{Pearson}(t_j, t_q) = \frac{\sum_{u_i \in C_{j,q}} (r_{i,j} - \overline{r}_j)(r_{i,q} - \overline{r}_q)}{\sqrt{\sum_{u_i \in C_{j,q}} (r_{i,j} - \overline{r}_j)^2} \times \sqrt{\sum_{u_i \in C_{j,q}} (r_{i,q} - \overline{r}_q)^2}}$
CPCC [27]	$\mathrm{CPCC}(t_j, t_q) = \frac{\sum_{u_i \in C_{j,q}} (r_{i,j} - r_{\mathrm{med}})(r_{i,q} - r_{\mathrm{med}})}{\sqrt{\sum_{u_i \in C_{j,q}} (r_{i,j} - r_{\mathrm{med}})^2} \times \sqrt{\sum_{u_i \in C_{j,q}} (r_{i,q} - r_{\mathrm{med}})^2}}$
Jaccard [28]	$\mathrm{Jaccard}(t_j, t_q) = \frac{\lvert U_j \cap U_q\rvert}{\lvert U_j \cup U_q\rvert}$
PIP [29]	$\mathrm{PIP}(t_j, t_q) = \sum_{u_i \in C_{j,q}} \mathrm{Pr}(r_{i,j}, r_{i,q}) \times \mathrm{Im}(r_{i,j}, r_{i,q}) \times \mathrm{Po}(r_{i,j}, r_{i,q})$
MCFV [30]	$\mathrm{MCFV}(t_j, t_q) = \frac{\vec{d}_j \cdot \vec{d}_q}{\lvert\vec{d}_j\rvert \times \lvert\vec{d}_q\rvert}$
TMJ [31]	$\mathrm{TMJ}(t_j, t_q) = (1 - \frac{\sqrt{\sum_{u_i \in C_{j,q}} (r_{i,j} - r_{i,q})^2}}{\sqrt{\sum_{u_i \in C_{j,q}} r_{i,j}^2} + \sqrt{\sum_{u_i \in C_{j,q}} r_{i,q}^2}}) \times \frac{\lvert U_j \cap U_q\rvert}{\lvert U_j \cup U_q\rvert}$
ED [32]	$\mathrm{ED}(t_j, t_q) = \sqrt{\sum_{u_i \in C_{j,q}} (r_{i,j} - r_{i,q})^2}$
MD [34]	$\mathrm{MD}(t_j, t_q) = \lvert\overline{r}_j - \overline{r}_q\rvert$

5) Jaccard [28] is defined as the number of the intersection divided by the number of the union of the rating users. Let $U_j = \{u_i | 1 \le i \le n, r_{i,j} \ne 0\}$ and $U_q = \{u_i | 1 \le i \le n, r_{i,q} \ne 0\}$.
6) PIP [29] consists of three factors, namely Proximity, Impact, and Popularity. Let $\mathrm{Pr}(r_{i,j}, r_{i,q})$, $\mathrm{Im}(r_{i,j}, r_{i,q})$ and $\mathrm{Po}(r_{i,j}, r_{i,q})$ be the value of Proximity, Impact and Popularity between rating $r_{i,j}$ and rating $r_{i,q}$, respectively.
7) MCFV [30] is defined to measure the angle between the multi-channel feature vectors of two items. Let $\vec{d}_j$ and $\vec{d}_q$ be the multi-channel feature vectors of item t_j and item t_q, respectively.
8) TMJ [31] is defined to integrate Triangle [31] and Jaccard similarities. Triangle considers both the length and the angle between the rating vectors of two items, namely the information of co-rating users, while Jaccard considers more information of non co-rating users.
9) ED [32], [33] is defined as the linear distance between two points in Euclidean space.
10) MD [34] is defined as the difference between the average ratings of two item.

To predict the rating of user u_i to item t_j, we first find item t_j's k nearest neighbors, which are already rated by u_i. Denote this neighbor set as $N(t_j)$, the prediction is given by

$$p_{i,j} = \frac{\sum_{t_q \in N(t_j)} r_{i,q}}{k}. \tag{2}$$

Slope One algorithms calculate predictions directly without using distance measures. The weighted Slope One algorithm [3] computes the prediction as

$$p_{i,j} = \frac{\sum_{t_q \in T_i} (\mathrm{dev}_{j,q} + r_{i,q}) \times |C_{j,g}|}{\sum_{t_q \in T_i} |C_{j,q}|}, \tag{3}$$

where $T_i = \{t_j | 1 \le j \le m, r_{i,j} \ne 0\}$, $\mathrm{dev}_{j,q}$ is defined as

$$\mathrm{dev}_{j,q} = \sum_{u_i \in C_{j,q}} \frac{r_{i,j} - r_{i,q}}{|C_{j,q}|}. \tag{4}$$

C. ENSEMBLE LEARNING

Ensemble learning integrates multiple algorithms to achieve better predictive performance than any of them [35], [36]. The current ensemble methods can be broadly divided into two categories, namely serialization methods and parallelization methods. Serialization methods make strong dependencies among algorithms and their representative method is Boosting [8]. Parallelization methods do not have such dependency and their representative methods are Bagging [7] and Random Forest [37].

Bagging combines different predictions that individual algorithms trained in their respective sampling sets. It usually employs the averaging method for regression tasks and the voting method for classification tasks.

Denote the prediction obtained in agent a_k as $p_{i,j}^k$. The simple averaging method for dealing with continuous predictions is represented as

$$p_{i,j} = \frac{\sum_{k=1}^{M} p_{i,j}^k}{M}. \tag{5}$$

Denote the prediction marker set as $\{c_1, c_2, \ldots, c_H\}$. The predictions of agent a_k are represented as an H-dimensional vector $(d_k^1; d_k^2; \ldots; d_k^H)$, where d_k^z is the output of $p_{i,j}^k$ on predictive marker c_z. The plurality voting method for dealing with discrete predictions is represented as

$$p_{i,j} = c_{\arg\max_z \sum_{k=1}^{M} d_k^z}. \tag{6}$$

D. ADAPTIVE LEARNING

Adaptive learning usually optimizes itself by adjusting parameters during data processing and analysis [38].

Its application in RSs has also received widespread attention. Shahabi and Chen [39] structured a model to automatically learn the confidence values by utilizing implicit users' relevance feedback. Deng *et al.* [21] proposed an adaptive personalized recommendation with fast learning ability. It allows new users, items and ratings to enter the system at a rapid rate. Luo and Yang [20] adopted the gradient descent method to minimize the prediction error and set the deviation coefficient to adaptive the prediction error.

III. AMRE MECHANISM

In this section, we describe the adaptive mechanism in the following three aspects. First, we introduce the framework of AMRE. Then, we define the reward-function to update the reward value. Finally, we illustrate the adaptive recommendation process through agent switching and recommendation by one agent.

A. THE PROPOSED FRAMEWORK

Figure 1 shows our AMRE framework. It is a continuous iterative process with four aspects. First, the roulette randomly selects an agent from the agent set $A = \{a_1, a_2, \ldots, a_M\}$. Each agent corresponds to a recommendation algorithm and a reward value v. Second, ratings are predicted by the recommendation algorithm in the current agent. Third, based on the correctness of the current recommendation, the reward-function feeds back a reward or a punishment to update v. Fourth, the roulette is triggered to choose another agent when $v < v_{\min}$.

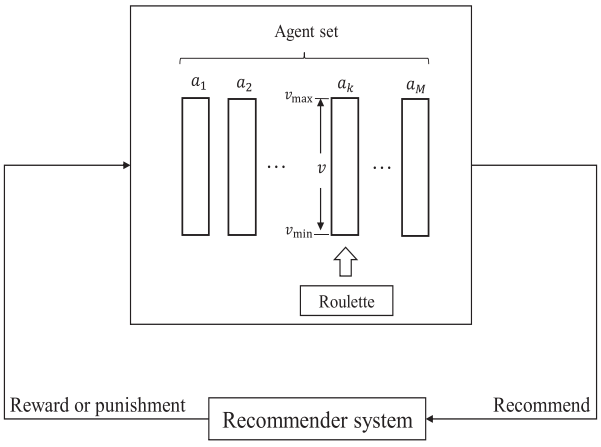


FIGURE 1. The framework of AMRE.

B. THE REWARD-FUNCTION

To illustrate the reward-function, we introduce two matrices. One is the interest matrix RM, the other is the recommendation matrix PM.

According to the threshold α, the rating matrix $R = (r_{i,j})_{n\times m}$ can be transferred to the interest matrix $RM = (rm_{i,j})_{n\times m}$, where

$$rm_{i,j} = \begin{cases} -1, & \text{if } 0 < r_{i,j} < \alpha; \\ 1, & \text{if } r_{i,j} \geq \alpha; \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

In other words, 1 and -1 indicate that the user likes and dislikes the item, respectively. 0 indicates unknown.

According to the threshold β, the prediction matrix $P = (p_{i,j})_{n\times m}$ can be transferred to the recommendation matrix $PM = (pm_{i,j})_{n\times m}$, where

$$pm_{i,j} = \begin{cases} -1, & \text{if } p_{i,j} < \beta; \\ 1, & \text{if } p_{i,j} \geq \beta. \end{cases} \tag{8}$$

In other words, 1 and -1 indicate the recommend and non-recommend behaviors, respectively.

By comparing PM with RM, the set of user-item pairs $\{\langle i,j\rangle | r_{i,j} \neq 0\}$ are classified into four regions, namely, true recommendation (RY), false recommendation (RN), false non-recommendation (NY) and true non-recommendation (NN). For example, $\alpha = 3.5$ and $\beta = 4$, with $r_{i,j} = 5$ and $p_{i,j} = 2.4$, user-item pair $\langle i,j\rangle$ is classified into NY. The numbers of user-item pairs in the four regions are given by

$$\begin{aligned} RY &= |\{\langle i,j\rangle | rm_{i,j} = 1,\ pm_{i,j} = 1\}|; \\ RN &= |\{\langle i,j\rangle | rm_{i,j} = -1,\ pm_{i,j} = 1\}|; \\ NY &= |\{\langle i,j\rangle | rm_{i,j} = 1,\ pm_{i,j} = -1\}|; \\ NN &= |\{\langle i,j\rangle | rm_{i,j} = -1,\ pm_{i,j} = -1\}|. \end{aligned} \tag{9}$$

Thus, based on the correctness of the recommendation, the reward-function is defined as

$$f_{i,j} = \begin{cases} w^-, & \text{if } \langle i,j\rangle \in RN; \\ w^+, & \text{if } \langle i,j\rangle \in RY; \\ 0, & \text{otherwise,} \end{cases} \tag{10}$$

where w^- and w^+ indicate punishment and reward, respectively.

Then we introduce the update process of the reward value v. For the item t_j, when it is recommended to the user u_i, u_i will be added to the user set U'. For any $i \in U'$, the reward value is updated as

$$v = v_0 + \sum_{u_i \in U'} f_{i,j}. \tag{11}$$

If the reward value v is less than the threshold $v_{\min}$, the roulette will be triggered to choose another agent.

C. ADAPTIVE RECOMMENDATION

There are two processes of adaptive recommendation for each item. One is agent switching and the other is recommendation by one agent.

1) AGENT SWITCHING

Algorithm 1 illustrates the process of agent switching for item t_j. We employ "leave-item-out" scenario which first leaves out all ratings of the current item. Then, ratings of the users in user set U' are given to predict the remaining ratings.

Algorithm 1 Agent Switching

Input: R, A, CN, γ
Output: P
1: $cn = 0$;
2: initialize P;
3: initialize $U' = \{u_i | 1 \leq i \leq \gamma \cdot n_j\}$, $|U'| = \gamma \cdot n_j$;
4: **while** (true) **do**
5: Choose agent $a_k \in A$;
6: Predict and recommend using agent a_k, update cn, P and U' accordingly;
7: **if** ($cn = CN$ or $|U'| = n_j$) **then**
8: break;
9: **end if**
10: **end while**
11: **return** P;

Step 1 (lines 1-3) initialize cn, P and U'. Where cn indicates the actual number of consecutive non-recommendation, the user set U' is initialized as the top γ of the total number of users who have rated item t_j.

Step 2 (line 5) selects an agent $a_k \in A$ by a random selection technique in the roulette.

Step 3 (line 6) performs predictions and recommendations by agent a_k. During the execution of the agent a_k, cn, P and U' will be updated accordingly. Since it is an important part, Section III-C.2 is designed to illustrate it in particular.

Step 4 (lines 7-9) determines whether to terminate the adaptive recommendation of item t_j when agent a_k is replaced. If cn reaches the upper bound CN, or $|U'|$ reaches the upper bound n_j, the adaptive recommendation for item t_j will be terminated directly. Otherwise, the loop continues and the roulette will be triggered to choose another agent.

2) RECOMMENDATION BY ONE AGENT

Algorithm 2 illustrates the recommendation process by one agent for item t_j.

Step 1 (line 3-4) predicts the remaining ratings by the recommendation algorithm in agent a_k. Then item t_j is recommended to the user u_{i^*} with the highest prediction q_{i^*} (line 4).

Step 2 (lines 5-20) discusses two situations of recommended and non-recommended, respectively. If prediction q_{i^*} is no less than the threshold β, item t_j will be recommended to user u_{i^*}. Otherwise, it will not be recommended to user u_{i^*}.

Step 2.1 (lines 5-16) shows the recommended situation:

(1) If user u_{i^*} likes item t_j (lines 7-8), the reward-function will feed back a reward w^+ to update the reward value v of agent a_k. In addition, an upper bound $v_{\max}$ is set for the reward value v.

Algorithm 2 Recommendation by One Agent

Input: R, a_k
Output: cn, U', P
1: $v = v_0$;
2: **while** ($|U'| < n_j$) **do**
3: Predict ratings for all $u_i \in U_j - U'$, and denote the prediction of u_i as q_i;
4: $i^* = \arg\max_{u_i \in U_j - U'} q_i$;
5: **if** (recommend t_j to u_{i^*}) **then**
6: $cn = 0$; //Reset this value
7: **if** (u_{i^*} likes t_j) **then**
8: $v = \min\{v + w^+, v_{\max}\}$; //Reward with an upper bound
9: **else**
10: $v = v + w^-$; //Punishment
11: **end if**
12: $U' = U' \cup \{u_{i^*}\}$;
13: $p_{i^*,j} = q_{i^*}$;
14: **if** ($v < v_{\min}$) **then**
15: break;
16: **end if**
17: **else**
18: cn++;//Not recommend
19: break;
20: **end if**
21: **end while**
22: **return** cn, U', P;

(2) If user u_{i^*} dislikes item t_j (lines 9-11), the reward-function will feed back a punishment w^-.

After obtaining the feedback (lines 12-13), user u_{i^*} is added to the user set U' and the prediction $p_{i^*,j}$ is updated. When the reward value v is lower than the threshold $v_{\min}$ (lines 14-16), the current agent will be replaced directly.

Step 2.2 (lines 17-20) shows the non-recommended situation. If it happens, the current agent will be replaced directly after updating cn.

D. A RUNNING EXAMPLE

We now explain the adaptive recommendation process. Here we set $\alpha = 3$, $\beta = 3.5$, $w^- = -0.75$, $w^+ = 0.5$, $v_0 = 0.1$, $v_{\min} = 0$, $v_{\max} = 1$ and $CN = 3$.

Figure 2 shows a running example for item t_1 in Table 2. At the beginning, the ratings $r_{1,1} = 3$ and $r_{2,1} = 4$ are given. The ratings of the user set $\{u_3, u_5, u_6, u_8, u_9\}$ are predicted based on $r_{1,1}$ and $r_{2,1}$. In this process, the roulette selects agents a_1, a_3 and a_2 in turn.

Figure 2(a) depicts the recommendations by agent a_1. In the first round, the recommendation algorithm of agent a_1 is in charge of predicting ratings of users in $\{u_3, u_5, u_6, u_8, u_9\}$. After sorting predictions in descending order, the candidate list of the first round is $(u_6, u_5, u_8, u_3, u_9)$. Because the highest prediction of 4.5 is higher than the threshold $\beta = 3.5$, item t_1 is recommended to

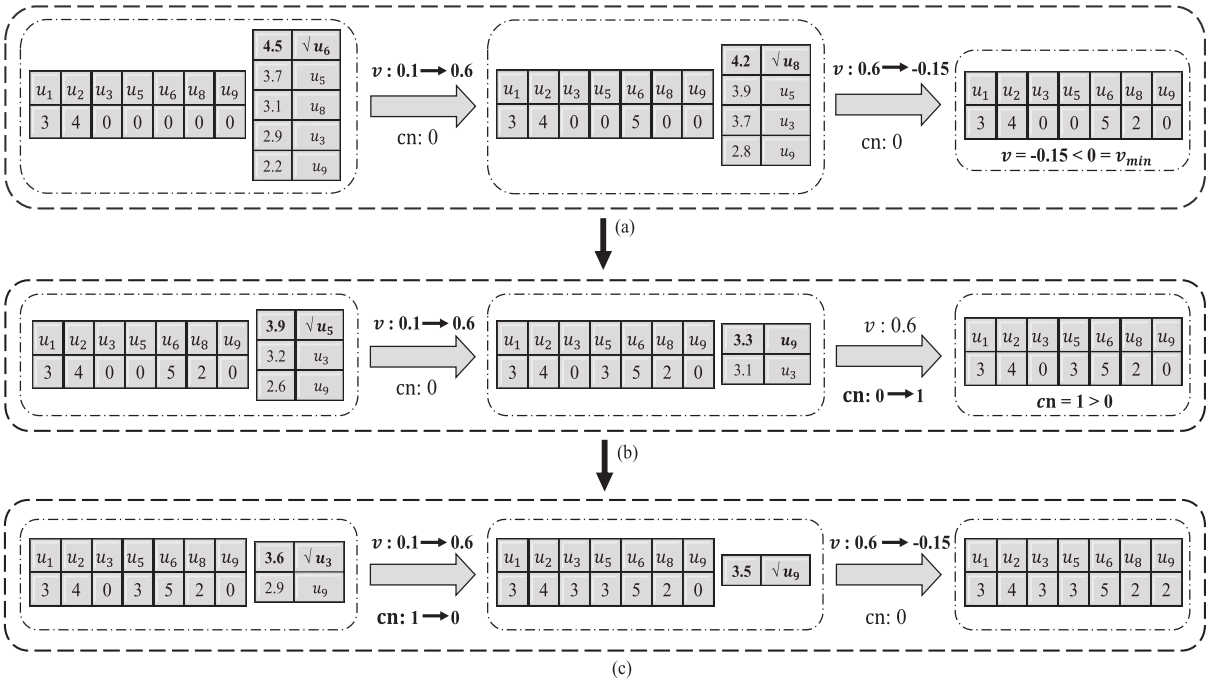


FIGURE 2. A running example for item t_1. (a) Agent a_1. (b) Agent a_3. (c) Agent a_2.

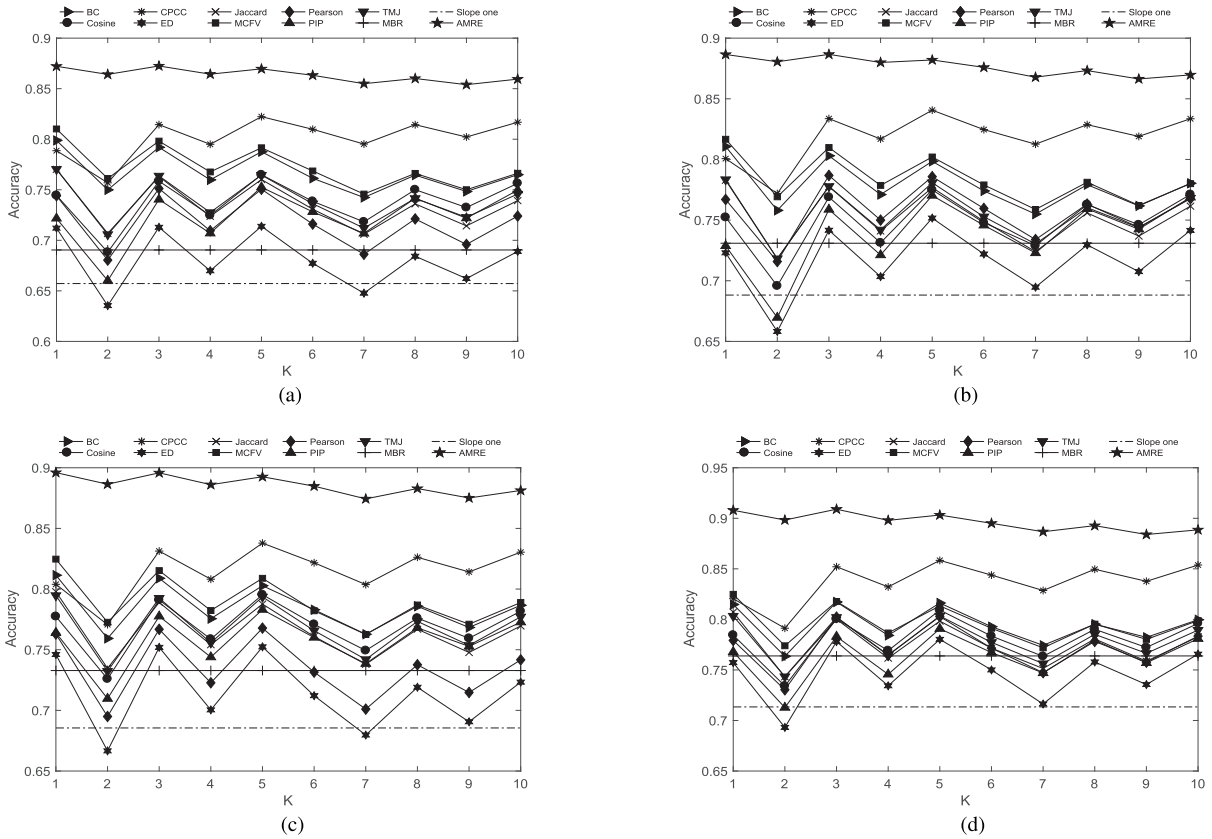


FIGURE 3. The Accuracy of AMRE and other individual recommendation algorithms on four datasets. (a) ML943u (0.2). (a) ML943u (0.3). (a) ML706u (0.2). (a) ML706u (0.3).

user u_6. Table 2 shows the actual rating of user u_6 to item t_1 is 5 which is higher than the threshold $\alpha = 3$, namely the recommendation is correct. Then the reward-function feeds back a reward of $w^+ = 0.5$ to update the reward value v to 0.6, and cn remains at 0. The rating of user u_6 is set to be known accordingly. In the second round, ratings of users

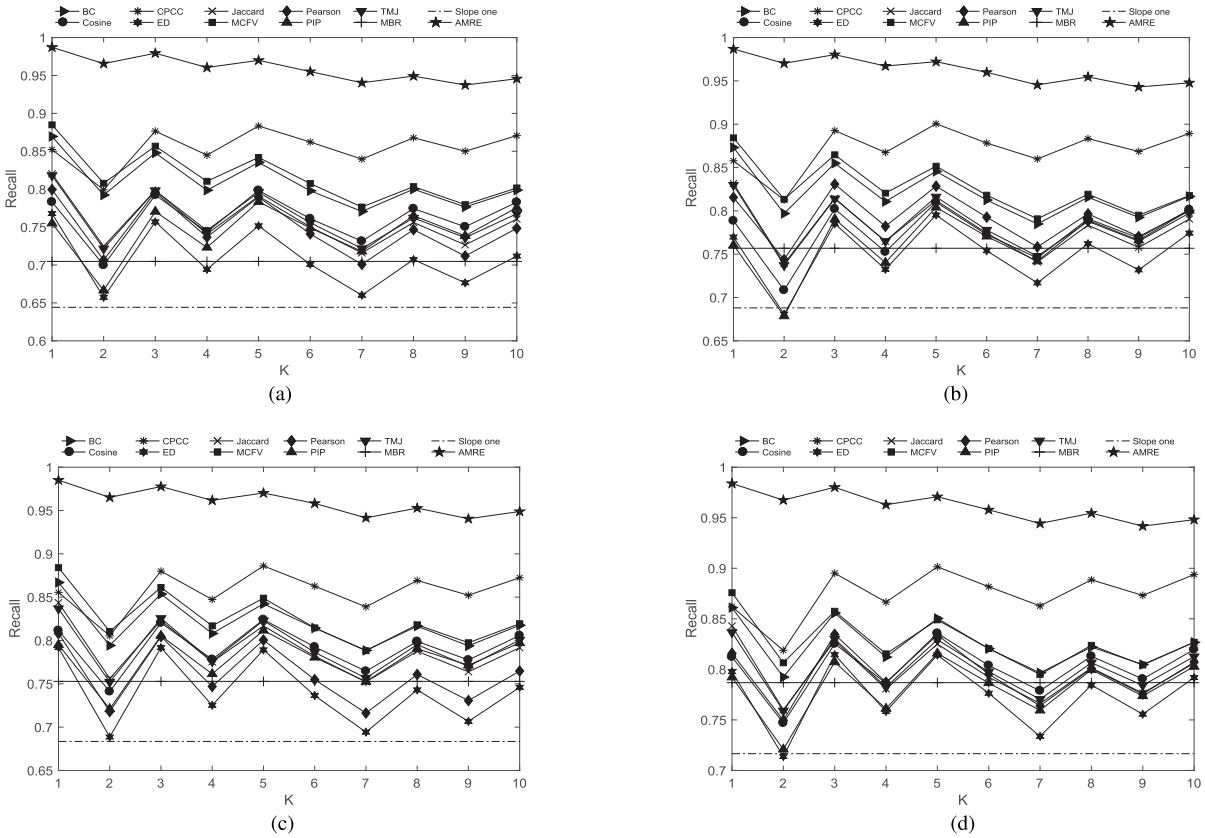


FIGURE 4. The Recall of AMRE and other individual recommendation algorithms on four datasets. (a) ML943u (0.2). (a) ML943u (0.3). (a) ML706u (0.2). (a) ML706u (0.3).

in $\{u_3, u_5, u_8, u_9\}$ are predicted. The sorted candidate list of the second round is (u_8, u_5, u_3, u_9). Then item t_1 is recommended to user u_8. Since this recommendation is incorrect, the reward-function feeds back a punishment $w^- = -0.75$ to update the reward value v to -0.15. Because -0.15 is lower than the threshold $v_{\min} = 0$, the roulette is triggered to choose another agent a_3 after setting the rating of user u_8 to be known.

Figure 2(b) depicts the recommendations by agent a_3. In the first round, ratings of users in $\{u_3, u_5, u_9\}$ are predicted. The sorted candidate list of the first round is (u_5, u_3, u_9). Then item t_1 is recommended to user u_5, and the reward-function feeds back a reward w^+ to update the reward value v to 0.6. The rating of user u_5 is set to be known accordingly. In the second round, ratings of users in $\{u_3, u_9\}$ are predicted. The sorted candidate list of the second round is (u_9, u_3). Since the prediction of item t_1 is below the threshold β, item t_1 is not recommended to user u_9. Therefore, cn is updated to 1 and the roulette is triggered to choose a_2 as the next agent.

Figure 2(c) depicts the recommendations by agent a_2. In the first round, ratings of users in $\{u_3, u_9\}$ are predicted. The sorted candidate list is (u_3, u_9). Then item t_1 is recommended to user u_3 and the reward-function feeds back a reward w^+. The reward value v is updated to 0.6, cn is updated to 0. The rating of user u_3 is set to be known accordingly. In the second round, the rating of user u_9 is predicted and item t_1 is recommended to user u_9. Then the reward-function feeds back a punishment w^- to update the reward value v to -0.15. Since ratings for all users have received feedbacks, the adaptive recommendation process for item t_1 is terminated.

IV. EXPERIMENTS

In this section, we design the sets of experiments to answer the following questions:

i) Does AMRE mechanism outperform individual well-known CF algorithms?
ii) Does AMRE mechanism outperform classical ensemble methods?

A. DATASETS

We use two MovieLens datasets (943 users × 1,682 movies and 706 users × 8,570 movies). AMRE requires an appropriate number of ratings to better implement adaptive recommendations, so we discard unpopular items that only have a small number of ratings. For each dataset, we first calculate the number of ratings for each item. Then the maximum of these numbers is selected in each dataset. Finally, we discard items whose ratings are less than the product of the screening

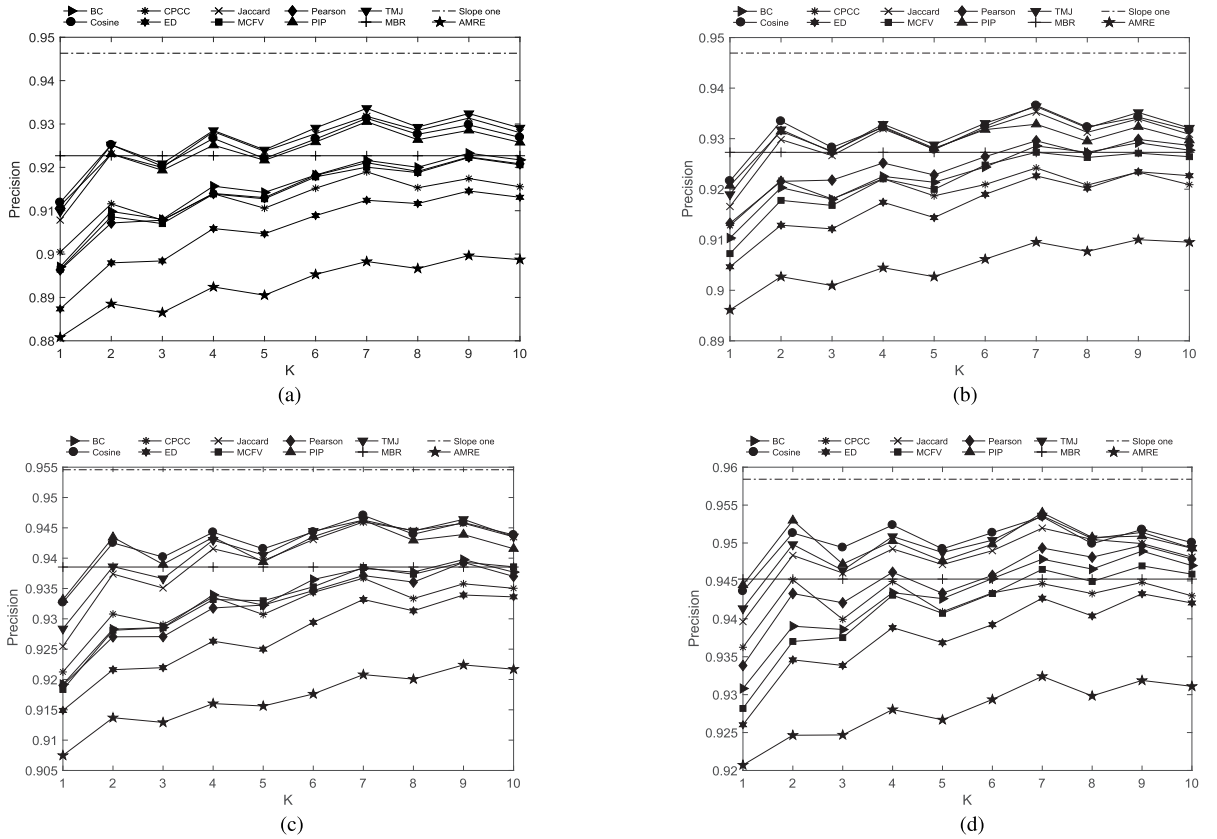


FIGURE 5. The Precision of AMRE and other individual recommendation algorithms on four datasets. (a) ML943u (0.2). (a) ML943u (0.3). (a) ML706u (0.2). (a) ML706u (0.3).

rate θ and the maximum. Table 4 lists the detail of four datasets with different θ.

TABLE 4. Datasets.

Dataset	θ	Users	Items	Ratings
ML943u (0.2)	0.2	943	290	60,133
ML943u (0.3)	0.3	943	151	40,310
ML706u (0.2)	0.2	706	322	38,038
ML706u (0.3)	0.3	706	158	24,842

B. EVALUATION METRICS

Based on the four regions mentioned in Eq. (9), we evaluate performances by adopting four metrics, namely Accuracy, Recall, Precision and F1-measure.

Accuracy is the ratio of the number of correct predictions to the total, which is defined as

$$\text{Accuracy} = \frac{NN + RY}{NN + RY + RN + NY}. \tag{12}$$

Recall is the ratio of the number of correct recommendations to the number of actual likes, which is defined as

$$\text{Recall} = \frac{RY}{RY + NY}. \tag{13}$$

Precision is the ratio of the number of correct recommendations to the total number of recommendations, which is defined as

$$\text{Precision} = \frac{RY}{RY + RN}. \tag{14}$$

F1-measure is the harmonic mean of Recall and Precision, which is defined as

$$\text{F1-measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{15}$$

C. EXPERIMENT DESIGN

We design two sets of experiments to answer the questions raised at the beginning of this section.

Exp1. We add eleven individual well-known CF algorithms to AMRE as agents, including Slope One and 10 k-NN measures: BC, Cosine, Pearson, CPCC, Jaccard, PIP, MCFV, TMJ, ED and MD. Then, we compare the performance of AMRE with the above eleven individual recommendation algorithms on different datasets in four metrics.

Exp2. We generate eleven data subsets (with replacement) containing 80% of the original dataset, and each of them is used for one CF algorithm, respectively. Then, we adopt the voting method and the averaging method to combine predictions obtained in different subsets. By integrating the predictions of the above eleven CF algorithms, we compare the performance of AMRE with these two ensemble methods on dataset ML706u (0.2).

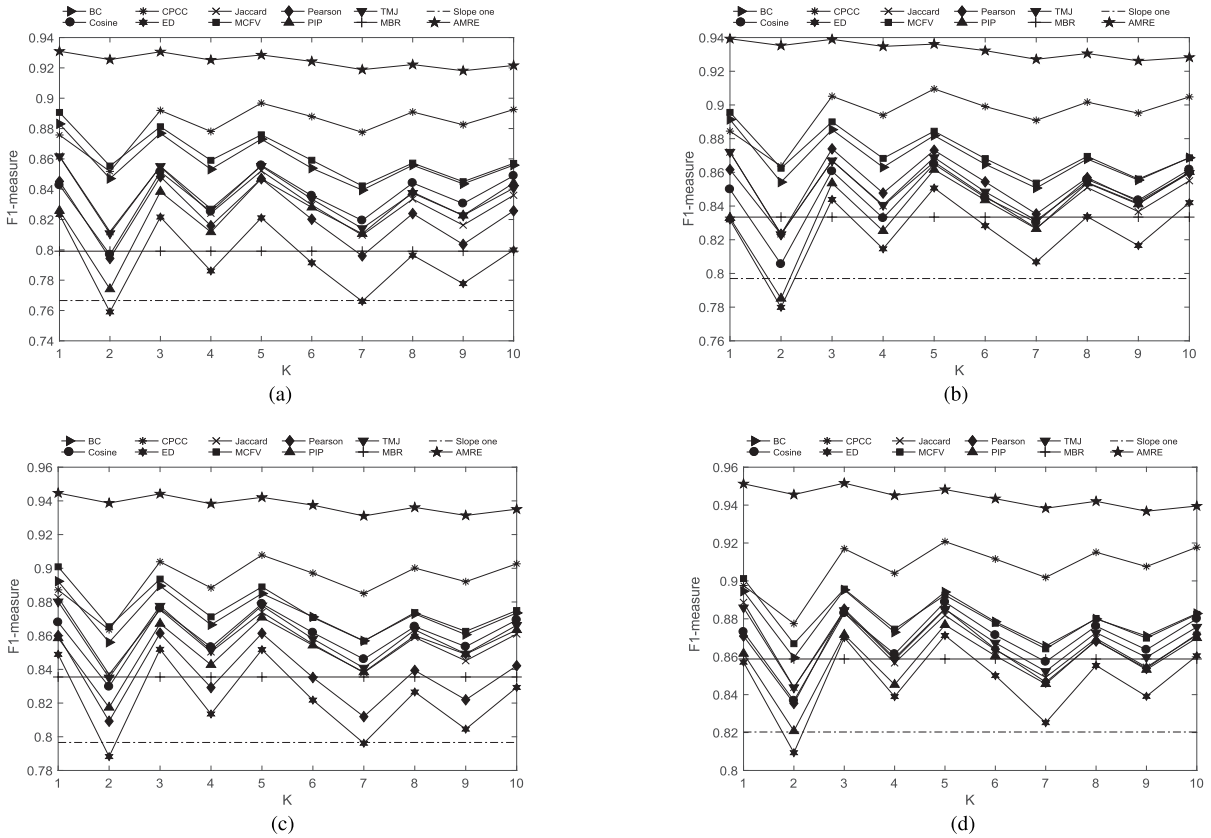


FIGURE 6. The F1-measure of AMRE and other individual recommendation algorithms on four datasets. (a) ML943u (0.2). (a) ML943u (0.3). (a) ML706u (0.2). (a) ML706u (0.3).

We employ ''leave-item-out'' scenario which first leaves out all ratings of the current item. Then the initial known ratio γ is given, which indicates that the top γ of the total ratings for the current item are used as the initial known rating.

Parameters are set as follows. $\gamma = 20\%$, $\alpha = 3$, $\beta = 3.6$, $w^- = -0.7$, $w^+ = 0.5$, $v_0 = 1$, $v_{\min} = 0$, $v_{\max} = 5$ and $\rho = 1$.

D. RESULTS

Two sets of experiments are undertaken to answer each of the questions raised at the beginning of the section.

1) COMPARISON WITH INDIVIDUAL CF ALGORITHMS

Tables 5, 6, 7 and 8 compare the four metrics obtained by AMRE and other individual recommendation algorithms. The AMRE mechanism achieves the best performance in terms of Accuracy, Recall, and F1-measure, but achieves a lower in Precision.

Compared with the best individual algorithms on four datasets, the Accuracy of AMRE is higher by 6.2%, 7.0%, 7.1% and 8.3%, respectively. The Recall is higher by 10.1%, 10.3%, 10.1% and 10.8%, respectively. The F1-measure is higher by 4.0%, 4.4%, 4.4% and 5.0%, respectively. The Precision is lower by 6.6%, 5.1%, 4.7% and 3.8%, respectively.

TABLE 5. The accuracy comparison.

Algorithm	ML943u (0.2)	ML943u (0.3)	ML706u (0.2)	ML706u (0.3)
BC	0.7988 (03)	0.8107 (03)	0.8114 (03)	0.8148 (04)
Cosine	0.7445 (07)	0.7523 (08)	0.7774 (07)	0.7845 (07)
CPCC	0.7887 (04)	0.8007 (04)	0.8039 (04)	0.8205 (03)
ED	0.7121 (10)	0.7230 (11)	0.7460 (10)	0.7572 (11)
Jaccard	0.7702 (06)	0.7828 (06)	0.7979 (05)	0.8072 (05)
MCFV	*0.8102* (02)	*0.8166* (02)	*0.8247* (02)	*0.8248* (02)
Pearson	0.7440 (08)	0.7670 (07)	0.7629 (09)	0.7792 (08)
PIP	0.7217 (09)	0.7287 (10)	0.7640 (08)	0.7678 (09)
TMJ	0.7703 (05)	0.7834 (05)	0.7947 (06)	0.8033 (06)
MBR	0.6904 (11)	0.7309 (09)	0.7328 (11)	0.7639 (10)
Slope One	0.6572 (12)	0.6882 (12)	0.6855 (12)	0.7134 (12)
AMRE	**0.8721** (01)	**0.8865** (01)	**0.8960** (01)	**0.9079** (01)

TABLE 6. The recall comparison.

Algorithm	ML943u (0.2)	ML943u (0.3)	ML706u (0.2)	ML706u (0.3)
BC	0.8695 (03)	0.8731 (03)	0.8669 (03)	0.8610 (04)
Cosine	0.7832 (08)	0.7886 (08)	0.8116 (07)	0.8124 (08)
CPCC	0.8523 (04)	0.8578 (04)	0.8555 (04)	0.8620 (03)
ED	0.7679 (09)	0.7698 (09)	0.7918 (10)	0.7978 (09)
Jaccard	0.8202 (05)	0.8315 (05)	0.8436 (05)	0.8430 (05)
MCFV	*0.8849* (02)	*0.8843* (02)	*0.8840* (02)	*0.8758* (02)
Pearson	0.7995 (07)	0.8155 (07)	0.8081 (08)	0.8159 (07)
PIP	0.7551 (10)	0.7605 (10)	0.7951 (09)	0.7922 (10)
TMJ	0.8182 (06)	0.8297 (06)	0.8368 (06)	0.8366 (06)
MBR	0.7048 (11)	0.7568 (11)	0.7528 (11)	0.7869 (11)
Slope One	0.6442 (12)	0.6880 (12)	0.6835 (12)	0.7166 (12)
AMRE	**0.9872** (01)	**0.9868** (01)	**0.9850** (01)	**0.9838** (01)

Figures 3, 4, 5 and 6 compare AMRE and other individual recommendation algorithms by setting different k values (i.e, number of the nearest neighbors) on different datasets with

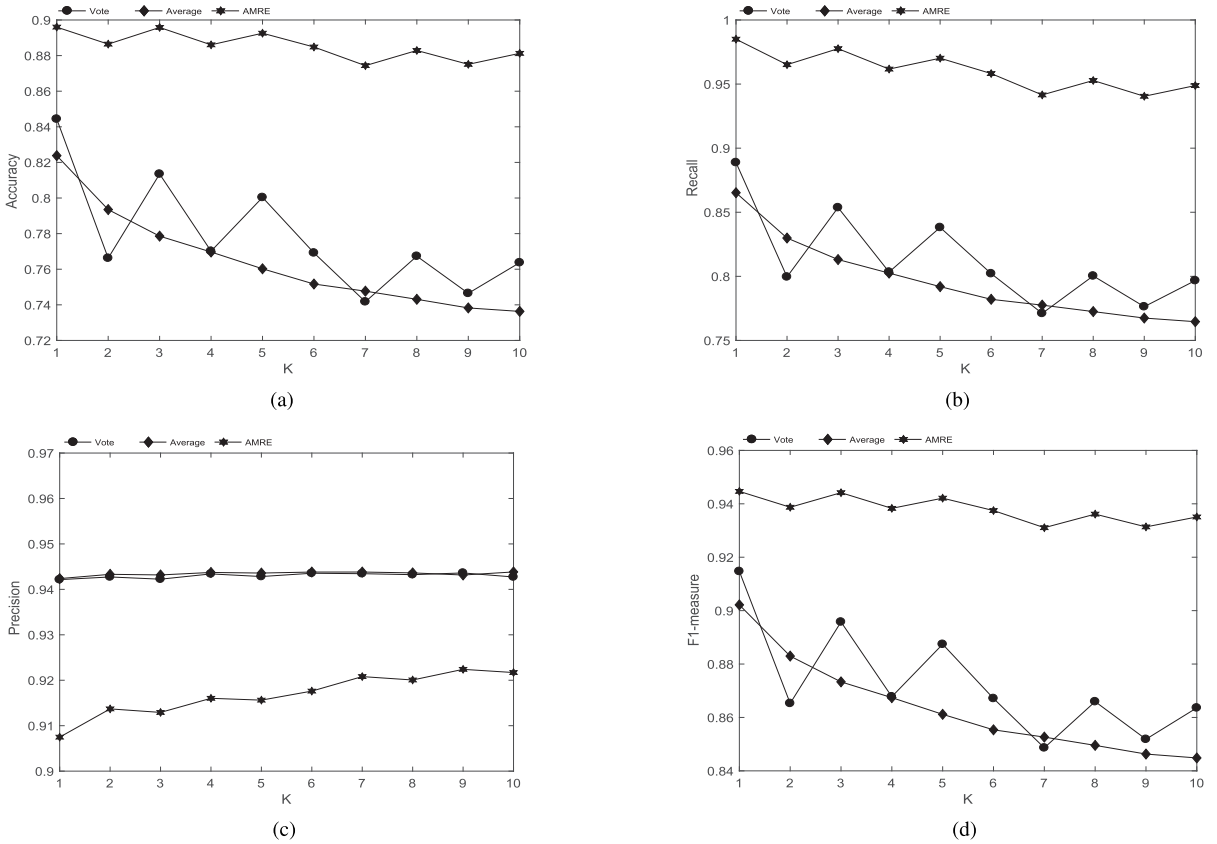


FIGURE 7. The four metrics on AMRE and ensemble methods. (a) Accuracy. (a) Recall. (a) Precision. (a) F1-measure.

TABLE 7. The precision comparison.

Algorithm	ML943u (0.2)	ML943u (0.3)	ML706u (0.2)	ML706u (0.3)
BC	0.8971 (08)	0.9103 (09)	0.9192 (08)	0.9308 (09)
Cosine	0.9119 (03)	0.9216 (03)	0.9327 (04)	0.9436 (04)
CPCC	0.9005 (07)	0.9128 (08)	0.9212 (07)	0.9362 (07)
ED	0.8874 (11)	0.9046 (11)	0.9149 (11)	0.9260 (11)
Jaccard	0.9078 (06)	0.9166 (06)	0.9255 (06)	0.9396 (06)
MCFV	0.8965 (09)	0.9073 (10)	0.9184 (10)	0.9282 (10)
Pearson	0.8963 (10)	0.9132 (07)	0.9190 (09)	0.9338 (08)
PIP	0.9112 (04)	0.9207 (04)	0.9332 (03)	0.9444 (03)
TMJ	0.9098 (05)	0.9190 (05)	0.9284 (05)	0.9414 (05)
MBR	*0.9227* (02)	*0.9273* (02)	*0.9385* (02)	*0.9452* (02)
Slope One	**0.9463** (01)	**0.9469** (01)	**0.9546** (01)	**0.9591** (01)
AMRE	0.8808 (12)	0.8961 (12)	0.9075 (12)	0.9207 (12)

TABLE 8. The F1-measure comparison.

Algorithm	ML943u (0.2)	ML943u (0.3)	ML706u (0.2)	ML706u (0.3)
BC	0.8831 (03)	0.8914 (03)	0.8923 (03)	0.8946 (04)
Cosine	0.8427 (08)	0.8499 (08)	0.8679 (07)	0.8731 (07)
CPCC	0.8757 (04)	0.8845 (04)	0.8872 (04)	0.8976 (03)
ED	0.8233 (10)	0.8318 (11)	0.8489 (10)	0.8571 (11)
Jaccard	0.8618 (05)	0.8720 (05)	0.8826 (05)	0.8887 (05)
MCFV	*0.8907* (02)	*0.8956* (02)	*0.9009* (02)	*0.9012* (02)
Pearson	0.8452 (07)	0.8616 (07)	0.8600 (08)	0.8709 (08)
PIP	0.8258 (09)	0.8330 (10)	0.8586 (09)	0.8617 (09)
TMJ	0.8616 (06)	0.8720 (05)	0.8802 (06)	0.8859 (06)
MBR	0.7991 (11)	0.8334 (09)	0.8355 (11)	0.8588 (10)
Slope One	0.7666 (12)	0.7970 (12)	0.7966 (12)	0.8203 (12)
AMRE	**0.9310** (01)	**0.9393** (01)	**0.9447** (01)	**0.9512** (01)

four metrics. As is seen, AMRE mechanism always performs the best in terms of Accuracy, Recall and F1-measure regardless of the k value, but not Precision.

2) COMPARISON WITH ENSEMBLE METHODS

Figure 7 compares AMRE with other two ensemble methods on dataset ML706u (0.2) by setting different k values. Similar to the last set of experiment, AMRE always obtains the best in Accuracy, Recall and F1-measure regardless of the k value, but obtains a lower Precision. Compared with two ensemble methods, Accuracy of AMRE is higher by 7.2% - 14.5% and 5.2% - 13.3%, Recall is higher by 12.0% - 18.4% and 9.6% - 17.0%, Precision is lower by 2.1% - 3.5% and 2.1% - 3.5%, F1-measure is higher by 4.3% - 9.0% and 3.0% - 8.2%, respectively.

E. DISCUSSIONS

AMRE obtains the best in Recall indicates that there are many recommended behaviors in mechanism. With the increase of recommended behaviors, the Precision will decrease accordingly. In other words, Recall and Precision are contradictory in the current situation. Therefore, we introduce F1-measure to consider them comprehensively. Experimental results show that AMRE is superior to the above CF algorithms and ensemble methods in general.

The two comparative experiments illustrate that different users are suitable for different recommendation algorithms (i.e., to obtain better predictions) for each item. Therefore, AMRE dynamically switches to an appropriate recommendation algorithm to achieve higher recommendation quality than individual algorithms and ensemble methods.

According to the above analysis, we can answer the questions raised at the beginning of this section.

1) AMRE outperforms eleven individual well-known CF algorithms in terms of Accuracy, Recall and F1-measure on four datasets.
2) AMRE outperforms two classical ensemble methods in terms of Accuracy, Recall and F1-measure on four datasets.

V. CONCLUSION AND FURTHER WORK

In this paper, we proposed AMRE with three parts, namely a set of agents, a reward-function and a roulette. AMRE obtains appropriate predictions by adaptively switching among recommendation algorithms. Experimental results show that AMRE is superior to well-known CF algorithms and classical ensemble methods in terms of Accuracy, Recall and F1-measure. Since AMRE is based on real-time feedback from users, it is especially suitable for online recommendation.

The following research topics deserve further investigation.

1) New agents with diverse base recommendation algorithms. For better recommendation, we will integrate more types of recommendation algorithms into AMRE.
2) New selection strategies for the roulette. One way of improving the effectiveness of AMRE might be weighted selection strategies.

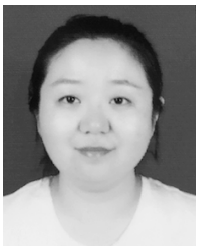
We hope that this work opens a new door to adaptive learning for recommendation problems.

REFERENCES

[1] J. Wang, A. P. de Vries, and M. J. T. Reinders, "Unified relevance models for rating prediction in collaborative filtering," *ACM Trans. Inf. Syst.*, vol. 26, no. 3, pp. 1–42, 2008.
[2] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Adv. Artif. Intell.*, vol. 2009, pp. 1–19, Jan. 2009.
[3] D. Lemire and A. Maclachlan, "Slope one predictors for online rating-based collaborative filtering," in *Proc. SIAM Int. Conf. Data Mining*, 2005, pp. 471–475.
[4] R. M. Bell and Y. Koren, "Scalable collaborative filtering with jointly derived neighborhood interpolation weights," in *Proc. 7th IEEE Int. Conf. Data Mining*, Oct. 2007, pp. 43–52.
[5] T. G. Dietterich, "Ensemble methods in machine learning," in *Proc. 1st Int. Workshop Multiple Classifier Syst.* Berlin, Germany: Springer, 2000, pp. 1–15.
[6] T. Sun and Z.-H. Zhou, "Structural diversity for decision tree ensemble learning," *Frontiers Comput. Sci.*, vol. 12, no. 3, pp. 560–570, 2018.
[7] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.
[8] R. E. Schapire, "The strength of weak learnability," *Mach. Learn.*, vol. 5, no. 2, pp. 197–227, 1990.
[9] D. H. Wolpert, "Stacked generalization," *Neural Netw.*, vol. 5, no. 2, pp. 241–259, 1992.
[10] X.-B. Yang and Y. Y. Yao, "Ensemble selector for attribute reduction," *Appl. Soft Comput.*, vol. 70, pp. 1–11, Sep. 2018.
[11] H. Yu and G. Wang, "An efficient gradual three-way decision cluster ensemble approach," in *Proc. 17th Int. Conf. Inf. Process. Manage. Uncertainty Knowl.-Based Syst.* Cham, Switzerland: Springer, 2018, pp. 711–723.
[12] M. Jahrer, A. Töscher, and R. Legenstein, "Combining predictions for accurate recommender systems," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2010, pp. 693–702.
[13] A. Bar, L. Rokach, G. Shani, B. Shapira, and A. Schclar, "Improving simple collaborative filtering models using ensemble methods," in *Proc. 11th Int. Workshop Multiple Classifier Syst.* Springer, 2013, pp. 1–12.
[14] M. D. Zeiler. (2012). "ADADELTA: An adaptive learning rate method." [Online]. Available: https://arxiv.org/abs/1212.5701
[15] H. Zhao, P. Wang, and Q. Hu, "Cost-sensitive feature selection based on adaptive neighborhood granularity with multi-level confidence," *Inf. Sci.*, vol. 366, pp. 134–149, Oct. 2016.
[16] A. Fan, H. Zhao, and W. Zhu, "Test-cost-sensitive attribute reduction on heterogeneous data for adaptive neighborhood model," *Soft Comput.*, vol. 20, no. 12, pp. 4813–4824, 2016.
[17] Q. Liu, X. Li, M. Ye, S. S. Ge, and X. Du, "Drift compensation for electronic nose by semi-supervised domain adaption," *IEEE Sensors J.*, vol. 14, no. 3, pp. 657–665, Mar. 2014.
[18] Q. Wu, R. Zheng, J. Pu, and S. Sun, "An adaptive control mechanism for mitigating DDoS attacks," in *Proc. IEEE Int. Conf. Autom. Logistics*, Aug. 2009, pp. 1760–1764.
[19] J.-S. R. Jang, "Self-learning fuzzy controllers based on temporal backpropagation," *IEEE Trans. Neural Netw.*, vol. 3, no. 5, pp. 714–723, Sep. 1992.
[20] Z. H. Luo and Y. Yang, "Adaptive ensemble learning recommendation algorithm," *Adv. Mater. Res.*, vols. 1044–1045, pp. 1433–1436, Oct. 2014.
[21] W.-Y. Deng, Q.-H. Zheng, S. Lian, and L. Chen, "Adaptive personalized recommendation based on adaptive learning," *Neurocomputing*, vol. 74, no. 11, pp. 1848–1858, 2011.
[22] H.-R. Zhang, F. Min, and X. He, "Aggregated recommendation through random forests," *Sci. World J.*, vol. 2014, Aug. 2014, Art. no. 649596.
[23] A. Bhattacharyya, "On a measure of divergence between two statistical populations defined by their probability distributions," *Bull. Calcutta Math. Soc.*, vol. 35, no. 1, pp. 99–109, 1943.
[24] B. K. Patra, R. Launonen, V. Ollikainen, and S. Nandi, "A new similarity measure using Bhattacharyya coefficient for collaborative filtering in sparse data," *Knowl.-Based Syst.*, vol. 82, pp. 163–177, Jul. 2015.
[25] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*. New York, NY, USA: McGraw-Hill, 1983.
[26] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: An open architecture for collaborative filtering of netnews," in *Proc. 5th ACM Conf. Comput. Supported Cooperat. Work*, 1994, pp. 175–186.
[27] U. Shardanand and P. Maes, "Social information filtering: Algorithms for automating 'word of mouth,'" in *Proc. 13th SIGCHI Conf. Hum. Factors Comput. Syst.*, vol. 44, 1995, pp. 210–217.
[28] P. Jaccard, "Nouvelles recherches sur la distribution florale," *Bull. Soc. Vaudoise Sci. Naturelles*, vol. 44, pp. 223–270, 1908.
[29] H. J. Ahn, "A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem," *Inf. Sci.*, vol. 178, no. 1, pp. 37–51, Jan. 2008.
[30] H.-R. Zhang, F. Min, Z.-H. Zhang, and S. Wang, "Efficient collaborative filtering recommendations with multi-channel feature vectors," *Int. J. Mach. Learn. Cybern.*, to be published, doi: 10.1007/s13042-018-0795-8.
[31] S.-B. Sun *et al.*, "Integrating triangle and Jaccard similarities for recommendation," *PLoS ONE*, vol. 12, no. 8, p. e0183570, 2017.
[32] K. L. Elmore and M. B. Richman, "Euclidean distance as a similarity metric for principal component analysis," *Monthly Weather Rev.*, vol. 129, no. 3, pp. 540–549, 2001.
[33] Z.-A. Xue, F. Cen, and L.-P. Wei, "A weighting fuzzy clustering algorithm based on Euclidean distance," in *Proc. 5th Int. Conf. Fuzzy Syst. Knowl. Discovery*, vol. 1, 2008, pp. 172–175.
[34] M. Zheng, F. Min, H. R. Zhang, and W. B. Chen, "Fast recommendations with the m-distance," *IEEE Access*, vol. 4, pp. 1464–1468, 2016.
[35] R. Polikar, "Ensemble based systems in decision making," *IEEE Circuits Syst. Mag.*, vol. 6, no. 3, pp. 21–45, Sep. 2006.
[36] L. Rokach, "Ensemble-based classifiers," *Artif. Intell. Rev.*, vol. 33, nos. 1–2, pp. 1–39, 2010.
[37] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[38] X.-Y. Jia, W.-W. Li, L. Shang, and J.-J. Chen, "An adaptive learning parameters algorithm in three-way decision-theoretic rough set model," *Acta Electron. Sinica*, vol. 39, no. 11, pp. 2520–2525, 2011.
[39] C. Shahabi and Y.-S. Chen, "An adaptive recommendation system without explicit acquisition of user relevance feedback," *Distrib. Parallel Databases*, vol. 14, no. 2, pp. 173–192, 2003.

QI HUANG is currently pursuing the Ph.D. degree with Southwest Petroleum University, Chengdu, China.



YUAN-YUAN XU received the M.S. degree from the University of Electronics Science and Technology of China, Chengdu, China, in 2007. She is currently a Lecturer with Southwest Petroleum University, Chengdu. Her current research interests include signal processing and recommender systems.

YONG CHEN received the M.S. degrees from the School of Computer Science and Engineering, University of Electronics Science and Technology of China, Chengdu, China, in 2008. He is currently an Associate Professor with the Shannxi University of Technology, Hanzhong. He has published over 10 refereed papers in various journals and conferences. His current research interests include data mining, recommender systems, and granular computing.

HENG-RU ZHANG received the M.S. degree from the School of Mechatronics Engineering, University of Electronics Science and Technology of China, Chengdu, China, in 2002. He is currently an Associate Professor with Southwest Petroleum University, Chengdu. He has authored over ten refereed papers in various journals and conferences, including *Information Sciences* and *Knowledge-Based Systems*. His current research interests include data mining, recommender systems, and granular computing.

FAN MIN (M'09) received the M.S. and Ph.D. degrees from the School of Computer Science and Engineering, University of Electronics Science and Technology of China, Chengdu, China, in 2000 and 2003, respectively. He visited the University of Vermont, Burlington, Vermont, from 2008 to 2009. He is currently a Professor with Southwest Petroleum University, Chengdu, and an Associate Editor of the *International Journal of Approximate Reasoning*. He has published over 120 refereed papers in various journals and conferences, including the *Information Sciences*, *International Journal of Approximate Reasoning*, *Knowledge-Based Systems*, and *Expert Systems with Applications*. His current research interests include granular computing, recommender systems, and active learning.