Knowledge-Based Systems 28 (2012) 1–12

Contents lists available at SciVerse ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Interest-based real-time content recommendation in online social communities

Dongsheng Li [a], Qin Lv [b,*], Xing Xie [a], Li Shang [b], Huanhuan Xia [a], Tun Lu [a], Ning Gu [a,*]

[a] Fudan University, Shanghai 200433, PR China
[b] University of Colorado Boulder, Boulder, CO 80309, USA

ARTICLE INFO

Article history:
Received 8 June 2011
Received in revised form 13 September 2011
Accepted 24 September 2011
Available online 12 October 2011

Keywords:
Content recommendation
Collaborative filtering
Real time
Interest
Online social community

ABSTRACT

The fast-growing popularity of online social communities and the massive amounts of user-generated content pose a critical need for, and new challenges on, content recommender system. The system needs to identify the unique and diverse interests of individual users and deliver content to interested users on a real-time basis. In this work, we propose *Farseer*, a system for personalized real-time content recommendation and delivery in online social communities. The proposed solution consists of a set of integrated offline and online algorithms that identify and utilize unique item-based interest clusters and cluster-based item rating in order to recommend newly-generated content items to individual users in real time. Our main contributions are (1) a detailed analysis of content popularity distribution and user interest distribution in online social communities; (2) a novel interest-based clustering and cluster-based content recommendation solution; and (3) a complete implementation and deployment in an online social community. Evaluation results gathered from real-world user studies demonstrate that the proposed system outperforms three widely-used collaborative filtering algorithms (kNN, PLSA, SVD) in existing recommender systems. It can effectively identify personal interests and improve the quality and efficiency of real-time personalized content recommendation in online social communities.

1. Introduction

Online social communities, also referred to as online social networks or member communities, have enjoyed explosive growth during the recent years and are now among the most visited websites on the Internet. Existing online social communities, such as Facebook, Livejournal, and Twitter, provide rich functionalities for online users to create, explore, and share interested content within various social forums and groups. Everyday, a large number of user-generated content items are posted online on a real-time basis. These data items are highly-dynamic and diverse, directly reflecting the unique interests of individual users. The fast-growing popularity of online social communities and the massive user-generated content pose a critical need for, also a great challenge, on identifying the unique interests of individual users and recommend content in real time.

A number of recommendation algorithms and systems have been developed, targeting diverse recommendation scenarios ranging from movies and products to dynamic news articles. Collaborative filtering (CF) is a class of information filtering techniques that identify and leverage the common knowledge or patterns shared among a group of users or agents [12]. Recent studies have demonstrated the potential of using CF to address the content recommendation challenge in online social communities [21]. Over the years, exemplary systems, such as Google News [6] and Amazon e-commerce [7], have gradually adopted CF techniques into their content recommender systems to help identify user-interested content. However, most existing content recommender systems suffer from a common flaw – they tend to yield biased decisions favoring highly popular content, and have difficulties in judging the content with low popularities. This is mainly due to the challenge of accurately characterizing the highly diverse yet unique interests of online users. Using existing CF techniques, users sharing highly popular content (i.e., common interests) tend to be classified as similar to each other. The unique interests of each individual are thus difficult to capture. For instance, in an online basketball forum, Alice is interested in the top NBA teams, but also supports her own home team. Since most of the online posts are devoted to the top, hence more popular, NBA teams, Alice is considered to be very similar to other people in the forum by CF. As a result, posts of Alice's home team may not be recommended to Alice as others are not interested and thus are less popular. A related problem, focusing on content diversification, was recently studied by Yu et al. [22], and techniques were proposed to compromise accuracy for diversity.

In this work, we propose and develop *Farseer*, a system for personalized real-time content recommendation and delivery in online social communities. Through interest-based content–user clustering and cluster-based content recommendation, plus the

* Corresponding authors. Tel.: +86 13764202435.
E-mail addresses: dongshengli@fudan.edu.cn (D. Li), qin.lv@colorado.edu (Q. Lv), ninggu@fudan.edu.cn (N. Gu).

0950-7051/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.knosys.2011.09.019

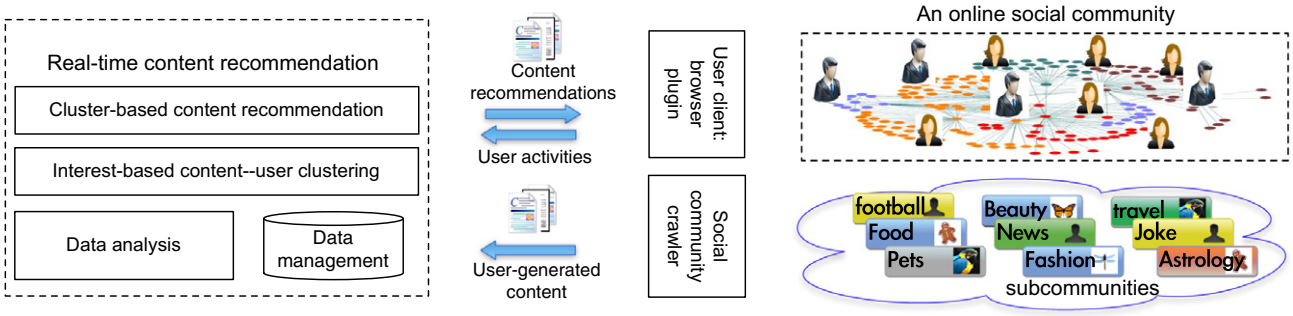

Fig. 1. *Farseer*: Personalized real-time content recommendation and delivery in online social communities.

temporal context of online user activities, *Farseer* can accurately characterize the interests of individual users and deliver content to interested users in real time. As shown in Fig. 1, *Farseer* consists of the following key components: (1) a real-time content recommendation server that supports data management, data analysis, interest-based content–user clustering, and cluster-based content recommendation; (2) a social community crawler that collects content and user activity updates in online social communities; and (3) a user-friendly browser plug-in that supports system–user interactions. Our work makes the following contributions:

- A detailed data analysis to understand the discrepancy between content popularity distribution and user interest distribution in online social communities. This study reveals the limitation of the biased decisions of existing CF methods and their inability to accurately assess and deliver less popular content to interested users.
- A new recommendation solution is proposed, which consists of a novel interest-based content–user clustering algorithm and cluster-based content recommendation algorithm, with the capability of accurately characterizing the diverse yet unique interests of individual online users. The proposed clustering algorithm can efficiently and accurately determine the optimal number of clusters, thus overcoming a key limitation of many other clustering algorithms. The proposed algorithms can be applied to other CF algorithms and improve their recommendation quality and efficiency.
- The proposed recommender system further leverages the temporal context of online user activities, thereby identifying user groups with similar interests and online access patterns. The user context information can be used to overcome the "false negative" problem suffered by many existing recommender systems, and further improve the recommendation quality.
- The proposed system has been fully implemented and deployed in an online social community with over 63,000 users, 2 million posts, and 18 million views. Evaluation and measurement results gathered from real-world user studies demonstrate that the proposed system can effectively identify personal interests and improve the quality and efficiency of real-time personalized content recommendation and delivery in online social communities. It outperforms three widely-used CF algorithms in existing recommender systems, as demonstrated in the experiments.

The rest of this article is organized as follows. Section 2 analyzes content popularity and user interests, and motivates the problem. Section 3 presents in detail interest-based content–user clustering, clustering-based content recommendation, and real-time recommendation strategies. A comprehensive evaluation of the proposed solution is presented in Section 4. Section 5 discusses the related work, and Section 6 concludes the article and discusses future work.

2. Data analysis

This section analyzes the distribution of content popularity and the diversity of user interests in online social communities, and highlights the challenge and importance of accurately identifying users' specific interests in content items with diverse popularities. We have collected a data set from Fudan BBS (http://www.bbs.fudan.edu.cn), one of the most popular online social communities among Chinese universities. It has over 63,000 users, 20,000 daily posts, 180,000 daily views, and in total over 2 million user posts and 18 million user views. Also, this data set contains detailed user view (read) information of specific content items, which is essential in content popularity and user interest analysis.

2.1. User interest group

In an online social community, users may view and post content items to specific *subcommunities* (e.g., forums on basketball or movie), which represent high-level interest categories. However, a user may not be interested in all content items posted to a subcommunity. To identify and manage the diverse user interests, within each subcommunity, we propose the notion of *interest group* as the basic unit of user interest. It is defined as follows:

Definition 1. For a given subcommunity SC with k interest groups $\{g_1, g_2, \ldots, g_k\}$, U is the set of users who are interested in SC, and I is the set of items in SC. Each interest group $g \in SC$ is a 3-tuple $g = < I_g, U_g, c_g >$, in which $I_g \in I$ is a non-empty set of items, $U_g \in U$ is a corresponding set of users, and $c_g \in I_g$ is the center of g. Each $g \in SC$ has the following properties: (1) For each user $u \in U_g$, u likes most of the items in I_g; (2) Center c_g is the item with the smallest average distance from other items in I_g, and c_g can be considered as a representative of group g's "interest"; (3) For any two interest groups $g_i, g_j \in SC (i \neq j)$, $I_{g_i} \cap I_{g_j} = \emptyset$ and $U_{g_i} \neq U_{g_j}$; and (4) $I_{g_1} \cup I_{g_2} \cup \cdots \cup I_{g_k} = I$, and $U_{g_1} \cup U_{g_2} \cup \cdots \cup U_{g_k} = U$.

The interest group structure within subcommunities is similar to the subcommunity structure in online social communities. However, the interest group has finer granularity and is more adequate in reflecting users' true interests, as a subcommunity may contain multiple content interests and draws the attention of different sets of users.

An interest group contains a set of users and a set of items – the items are similar to each other, and the users like most of these items. The interest groups in a subcommunity may vary in size, as interest groups representing popular or common interests will have more users than the interest groups that contains less popular items. But inside an interest group, items may not vary dramatically in popularity, as they are all liked by most users of the inter-

est group. Therefore, the interest group structure can effectively divide items into distinct groups of different interests and different popularities.

A user may have interests in several interest groups, but his/her "friends" vary in different interest groups. This is a closer reflection of our daily lives. Here, we present a real-world scenario which we discovered in a subcommunity of Fudan BBS. This example occurred in the subcommunity of "Astrology", where user u_1 is only interested in items that are related to his own constellation – "Scorpio". User u_2 is interested in items about "Scorpio", as well as popular items in the subcommunity. Using CF-based recommendation algorithm, we find that most of u_1's neighbors are also interested in popular items in the subcommunity, like u_2. As a result, items that are recommended to u_1 contain a lot of items that are popular but not related to "Scorpio". Meanwhile, most of the neighbors of u_2 are users who like popular items, not users like u_1. Thus, u_2 is recommended with many popular items and few items that are related to "Scorpio". In this case, the recommendations to u_1 and u_2 are both inaccurate. By adding the interest group structure within the subcommunity, u_1 and u_2 will be in the interest group of "Scorpio", and be recommended with items about "Scorpio". Meanwhile, u_2 will be in the interest group of the popular items, and be recommended with popular items. Thus, each of them can receive recommendations that match *only* and *all* their individual interests – *only* means a user will not be recommended items that do not match his/her interests, and *all* means that the user will get recommendations that cover all his/her interests, be it popular or not.

2.2. Content popularity and user interests

An online social community, typically organized as a number of subcommunities, covers diverse social interests (e.g., over 100 subcommunities in Fudan BBS). Driven by the diverse interests, user activities (e.g., posting an article or viewing a post) directly affect the content of data items and their popularity in an online social community. A highly popular post reflects common interest shared within or even across multiple subcommunities, while a post that only draws attention from a small interest group has low popularity.

Fig. 2(a) plots the content popularity distribution of the Fudan BBS data set, and its Astrology subcommunity. The plot highlights the heterogeneity in content popularity. A small percentage of the online content is highly popular, i.e., 5% of the most popular content is viewed by approximately 30% of the online users. On the other hand, the long tail distribution, i.e., a large percentage of the online content with similar (and lower) popularity, reflects the diverse interests of online users. Consider the Astrology subcommunity, which has 12 implicit interest groups corresponding to the 12 different zodiacs. Fig. 2(b) shows the cumulative distribution of the fraction of users who belong to a certain number of the 12 interest groups. As we can see, most users belong to only one or a few of the interest groups, thus the diversity of user interests within the subcommunity. Similar patterns of content popularity and user interest diversity have been observed in other subcommunities.

2.3. Impact of content popularity & interest diversity on recommendation

Existing CF-based content recommender systems identify the common patterns, e.g., similar interest, among a group of users, and deliver content to an end user based on the opinions of other group members sharing similar interest. Considering Google News personalization, a recent work by Das et al. [6], it assigns each individual user into multiple communities in which users share similar interest, and then makes personalized content recommendations via weighted average of a content item's interest level across all the communities that the user belongs to.

The diverse user interests in online social communities pose serious challenges to personalized content recommendation. Using existing CF methods, user interest grouping is heavily influenced by highly popular content. As illustrated by the basketball example in Section 1, users sharing similar interest on highly popular content (e.g., top NBA teams) are considered very similar. The decision on less popular content, which matches the unique interest of an individual or a small group of users, will be significantly influenced by the irrelevant majorities. Therefore, we argue that existing CF methods tend to make biased decisions favoring highly popular content, and fail to recommend content with less popularity but fits the unique interests of individual users.

Fig. 3(a) shows the recommendation decisions of content items sorted by popularity, using either a MinHash-based CF method (MCF) or our proposed Farseer solution (Section 3). The MCF method uses a state-of-the-art MinHash method [6] for content recommendation within each Fudan BBS subcommunity. Farseer, on the other hand, uses interest-based content–user clustering to effectively identify interest groups and user interests on content with diverse popularities. As shown in Fig. 3(a), compared with Farseer, the MCF method indeed favors popular content – considerably more recommendations were made on the highly-popular content items. Such biased decision significantly affects the recommendation quality. As shown in Fig. 3(b), Farseer outperforms the MCF method by 20% on average.

In summary, content popularity and user interest diversity have a strong impact on recommendation quality. It is thus critical to accurately characterize the diverse user interests to enable high-quality content recommendation in online social communities. To this end, we propose interest-based clustering algorithm and cluster-based content recommendation strategies, which can effectively characterize the internal user interest structures, i.e., interest groups, within social communities and subcommunities, thereby effectively identifying users' personal interests and deliver content with diverse popularities to interested users.

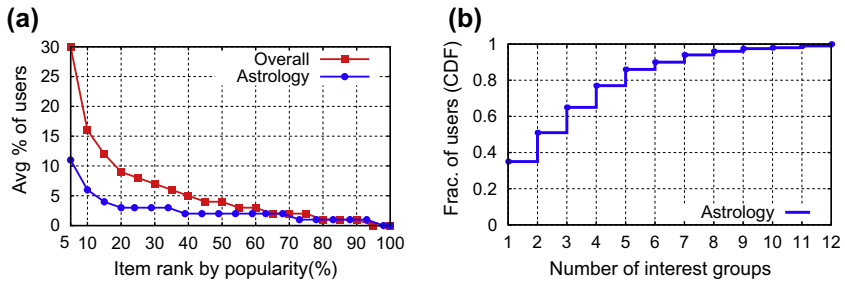

Fig. 2. (a) Diversity of content popularity. (b) Diversity of user interests within a subcommunity.

3. The *Farseer* content recommender system design

In this section, we present the detailed design of Farseer for personalized real-time content recommendation and delivery. Specifically, we focus on two key components in the recommendation process (Fig. 1).

- **Interest-based content–user clustering.** Our first step is to identify inherent structures, e.g., interest groups, within subcommunities, such that content and users can be quickly mapped to diverse yet specific interest clusters. The key challenges of this step are (1) how to measure the distance of items, and (2) how to determine the optimal number of clusters for a given subcommunity.
- **Real-time interest group based content recommendation.** Given the interest group clusters that users belong to, along with the temporal context of different users, we propose personalized recommendation algorithms that utilize users' explicit interests and level of interest in different clusters. At runtime, further techniques are proposed to extract user temporal context and select appropriate neighbors for real-time recommendation, as well as dynamic personalization and adaptation of user temporal weight information.

3.1. Interest-based content–user clustering

The first algorithm we propose is an interest-based content–user clustering algorithm that characterizes the internal structure, i.e., interest groups, within individual social subcommunities, thereby helping identify the diverse interests of individual users. Specifically, we aim to create a clustering structure that identifies the interest groups (of items) and users' membership in these interest groups. As mentioned earlier, an online social community is organized as a set of subcommunities (e.g., "automobile"). Each subcommunity covers a variety of user interests and distinct interest groups may exist (e.g., "car purchase" or "car maintenance"), albeit the management structure inside each subcommunity is typically flat. Characterizing such implicit interest group structures is essential, as it help understand and quantify the diverse interests of individual users and then deliver content to interested users. Concerns may arise that the interest group structure may suffer from the two main common flaws of CF: cold start and data sparsity [19]. Since the amounts of users and items are large, there are still sufficient data to make accurate recommendations in interest groups. Meanwhile, as all interested users are contained in the interest groups, this will not make the cold start problem worse. Furthermore, the interest group structures can help solve the data sparsity problem, which is suffered by many recommendation algorithms, because the user item rating matrices in interest groups are much denser than that in subcommunities. Besides the above concerns, determining the optimal number of clusters, i.e., interest groups, has been a challenge in existing clustering algorithms [16,26,3].

3.1.1. Objective function

In interest group clustering, our first task is to determine the objective function for measuring the goodness of the clustered interest groups. Intuitively, a better clustering should have a higher probability of assigning users into the right interest groups. Therefore, we can define an objective function that is optimized when the probability of clustering error is minimal. Specifically, for a given cluster c and a given user $u \in U$, let A be the event that items belong to cluster c and B be the event that items belong to the viewed list of user u. Then, the joint probability $P(A,B)$ represents the probability that user u belongs to cluster c, since the items viewed (liked) by u belong to c. We refer to this joint probability as $Support(u,c)$. For a clustering of k clusters C, the event of classifying an item into a cluster $c \in C$ is exclusive, so is the event of a clustering error. The probability that one event in k exclusive events happens is the sum of the probabilities of all k events. Thus, for user u, the probability of clustering error can be calculated as $PE_u = 1 - \sum_{c \in C} w_{u,c} \times Support(u,c)$, where $w_{u,c}$ is the fraction of items viewed by user u that belong to cluster c. Our objective of clustering is to maximize:

$$Objective = \sum_{u \in U} \sum_{c \in C} w_{u,c} \times Support(u,c) \quad (1)$$

which is the sum of the probabilities of clustering users into the "right" clusters. Therefore, maximizing the *Objective* function corresponds to the smallest probability of clustering error.

3.1.2. Interest-based clustering algorithm

Using the *Objective* function defined above, we propose an interest-based clustering algorithm that not only determines the proper number of clusters k, but also adjusts a given clustering for the optimal *Objective* value. As a result, our algorithm finds the optimal clustering that represents the internal structure of interest groups and users' diverse interests in these groups.

Finding the optimal clustering of the interest groups is challenging. The size of candidate solution space is $O(n^n)$, as each item has the possibility of being clustered into any interest groups. The maximal number of interest groups is n, where n is the number of items. Also, there is no clear relationship among possible solutions, so we cannot adopt the deterministic optimization algorithms to find the global optimum. Instead, in Farseer, we leverage a genetic algorithm [30] based method to find the near optimal solution.

The proposed interest-based clustering algorithm adopts the idea of hierarchical agglomerative clustering. We first assign each item into a single cluster. Then, in each round of iteration, we combine the small clusters which can produce the biggest increase of the *Objective* value. After some iterations, the clustering algorithm will stop when the *Objective* value does not increase. Since the number of items in online social communities is huge, this method is not efficient. To address this issue, we add an initialization step, which generates "micro clusters" by grouping similar items into small groups using the classical k-means clustering algorithm [18], and treat each"micro cluster" as a single item in the hierarchical agglomerative clustering process. This approach significantly improves clustering efficiency without much loss in clustering accuracy.

However, this hierarchical clustering process may not converge to the global optimum, so we need to further optimize the interest group clustering to find the near optimal and even optimal interest group clustering. The basic idea of the optimization process is to introduce "Mutation" and "Crossover" into interest groups. Also, to obtain the optimal number of clusters, we require that the optimization algorithm to automatically "Merge" and "Divide" interest groups, thus the number of clusters can be changed. In Farseer, these operations are defined as following:

- **Mutation.** Randomly select one interest group g and select 10% of items in g with the largest distances from the center. Assign each item to another interest group with the smallest distance.
- **Crossover.** Randomly select two interest groups g_1 and g_2, select 10% of items that have the largest distances from the center in g_1 and g_2, then swap the two sets of items.
- **Merge.** Randomly select two interest groups g_1 and g_2, combine these two groups (both item sets and user sets) into a new interest group.

- **Divide.** Randomly select one interest group g, divide it into two interest groups g_1 and g_2. The center of g_1 is the center of g, and the center of g_2 is the item in g that has the biggest distance from g_1. Then, items that are closer to the center of $g_1(g_2)$ are assigned to $g_1(g_2)$.

To calculate the distance between two items, we leverage Jaccard similarity. Specifically,

$$Distance(i,j) = 1 - JaccardSimilarity(i,j) \quad (2)$$

$$= 1 - |U_i \cap U_j| / |U_i \cup U_j| \quad (3)$$

where i and j are items, U_i and U_j are the sets of users who like item i and j, respectively. The details of the proposed interest group clustering algorithm are described in Algorithm 1. Steps 1 to 14 in the algorithm are the cluster-merging steps. Steps 15 to 30 are used to optimize the clustering, we call them the optimization steps.

Algorithm 1: InterestGroupClustering(U,X)

Require: U is the set of users, X is the set of items, and $Obj(U,X,C)$ is the *Objective* value of clustering U and X as C. $Move(C,x,c_1,c_2)$ returns the clustering of moving x from cluster c_1 to cluster c_2.

1: Initialize $C = \{c_1, c_2, \ldots, c_k\}$ using standard k-means clustering. $delta = 1$
2: **while** $delta > 0$ **do**
3: $delta = 0$, $C^* = \emptyset$
4: **for** each $c_i, c_j \in C$, and $i \neq j$ **do**
5: $C' = C - \{c_i\} - \{c_j\} + \{c_i \cup c_j\}$, where $\{c_i \cup c_j\}$ is the merged cluster of c_i and c_j
6: $delta(i,j) = Obj(U,X,C') - Obj(U,X,C)$
7: **if** $delta(i,j) > delta$ **then**
8: $delta = delta(i,j)$, $C^* = C'$
9: **end if**
10: **end for**
11: **if** $delta > 0$ **then**
12: $C = C^*$, where C^* is the clustering with the biggest $delta$.
13: **end if**
14: **end while**
15: $round = 0$
16: **while** $round < CONSTANT$ **do**
17: Select an operation $o \in \{Crossover, Merge, Divide\}$ and two interest groups $g_1, g_2 \in C$ randomly, and operate o on g_1 and g_2 to generate new clustering C'
18: **if** $Obj(U,X,C') > Obj(U,X,C)$ **then**
19: $C = C'$
20: **end if**
21: **if** C does not change for 10 rounds **then**
22: Select an interest group $g \in C$ randomly, and do *Mutation* on g
23: **end if**
24: $round++$
25: **end while**
26: Return C as the clustering with the biggest *Objective* value in all rounds

For a subcommunity consisting of n users and m items, *InterestGroupClustering*(U,X) has a complexity of $O(knm)$. Calculating *Objective* is $O(nm)$. Thus, the cluster-merging steps have a complexity of $O(k^2nm)$. As for the optimization steps, we perform at least one operation in each round, the complexity of which is $O(nm)$. Also, the calculation of the *Objective* value is $O(nm)$. The optimization only takes a constant number of steps, so the complexity is $O(nm)$. Overall, the complexity of *InterestGroupClustering*(U,X) is $O(nm)$, which is acceptable for offline processing.

3.2. Real-time recommendation

One important design goal of Farseer is real-time content recommendation. Besides leveraging interest groups of both users and items, we also consider real-time context information of online users, which helps address the following issues in real-time content recommendation.

- *False negatives:* In online social communities, most users only browse a small number of posts. In particular, online posts are typically organized in descending order of posting time, i.e., newly posted articles are placed on the first page. Therefore, users tend to ignore articles that are posted when they are offline, even though some of those posts may be of interest to them. Marking such "ignored" content as uninterested leads to incorrect user interest estimation. Leveraging users' time context can help solve this problem.
- *Real-time neighbor selection:* CF methods make content recommendations based on the opinion of others, also called neighbor group. Run-time neighbor selection thus has critical impact on recommendation quality. Leveraging users' time context information can help to accurately identify the group of online users (neighbors) who have already examined the targeted content item.
- *Different levels of interests:* In online social communities, users will like some items and show their interests by reading them. Such "read" information is binary and may not reflect the different levels of user interests. Users tend to read items for longer time if they find them more interesting. We adopt a time-based weighting method to help address this problem.

In this section, we propose a novel interest group based real-time recommendation algorithm, which can compute item ratings incrementally and deliver content in real-time.

3.2.1. User time context extraction

We propose and develop a method to extract users' real-time context information, more specifically, users' online sessions. A user online session is defined as the time interval when a user is online, and the set of content items that the user has examined. To identify the interests of an online user, only the content items belonging to the user's online sessions are considered. We define a user context matrix $CM \in \{0,1\}^{n \times m}$ to represent the user context information, where n is the number of users and m is the number of items. $CM_{u,i} = 1$ means that user u's context contains item i, and $CM_{u,i} = 0$ otherwise. Next, we determine the values in CM as follows:

$$CM_{u,i} = \begin{cases} 1 & \text{if } u \text{ posts/comments on/clicks at } i \\ \Gamma(t,\tau) \cdot \delta(t_p,t) & \text{otherwise} \end{cases}$$

where t is the time when Farseer detects an online action (i.e., clicks, comments or posts) of u, t_p is the time when item i is posted, and τ is the time of user u's last online action. The $\Gamma(t,\tau)$ function determines if this new action means a new session of user u, and it is defined as follows:

$$\Gamma(t,\tau) = \begin{cases} 1 & \text{if } t - \tau < \theta \\ 0 & \text{otherwise} \end{cases}$$

where θ is a predefined threshold to judge if this new online action is late enough from the last online action. We adopt $\theta = 30$ *minutes* based on a study by Cooley et al. [17]. And $\delta(t_p,t)$ is used to judge

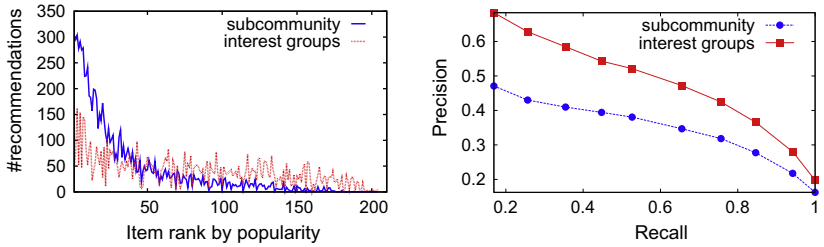

Fig. 3. Comparison of subcommunity-based recommendation (MCF) and interest group-based recommendation (Farseer): (Left) CF's bias for popular content; (Right) Impact of user interest diversity on recommendation quality.

whether item i is posted before t. If i is posted before t, i should have the possibility of being "ignored" by u. $\delta(t_p,t)$ is defined as follows:

$$\delta(t_p,t)=\begin{cases}1 & \text{if } t_p<t\\ 0 & \text{otherwise}\end{cases}$$

The above calculation is incremental as the user performs online actions, and the calculation is triggered only when a new online user action is detected.

3.2.2. Real-time neighbor selection

We next describe the real-time neighbor selection method for online content recommendation. After interest group clustering, users and items are clustered into interest groups, we define a user interest group matrix $GM\in\{0,1\}^{k,n}$, where k is the number of interest groups and n is the number of users. $GM_{c,u}=1$ means the user u has interest in interest group c. Similarly, after the user context extraction, we will obtain the user context matrix CM in real-time. Then, the neighbor user vector of item i, $NB_i\in\{0,1\}^{n\times 1}$, can be calculated as follows:

$$NB_i=CM_i\cdot GM_c^T \tag{4}$$

where CM_i is the context vector of item i, c is the interest group that contains i, and GM_c is the user vector of interest group c. User u will be a neighbor to calculate the rating of i if $NB_{i,u}=1$.

CM is updated incrementally as new user actions are detected. Based on the update of CM, which is recorded in $\Delta CM\in\{0,1\}^{n\times m}$. Since $\Delta CM\in\{0,1\}^{n\times m}$ is a very sparse matrix, we store it into a Hash_Map, which is organized as $<c_{id}, list<r_{id}\gg$. Its key is the column id which means the c_{id}-th column of ΔCM, and $list<r_{id}>$ contains the row index of all non-zero elements in the c_{id}-th column of ΔCM. We design an incremental neighbor calculation method which can avoid the duplicate computation of neighbors (Algorithm 2).

Algorithm 2: *NeighborSelection*$(i,c,GM,\Delta CM,NB_i)$

Require: i is the item that we need to determine its neighbor set of recommendation, c
is the index of the interest group that contains i, GM is the user interest group matrix, ΔCM is the set
of updated user context matrix, and NB_i is the neighbor vector of the last round of calculation
1: Store ΔCM into Hash_Map *C_Map*
2: l_c is the list corresponding to c in *C_Map*
3: **if** $l_c\neq\emptyset$ **then**
4: **for** each $u\in l_c$ **do**
5: **if** $NB_{i,u}\neq 1$ **then**
6: $NB_{i,u}=GM_{c,u}$
7: **end if**
8: **end for**
9: **end if**
10: return NB_i

3.2.3. User time weight adaptation

It is common that users will spend more time on items that are of more interest to them, and change to another item if he/she finds the current item not interesting. Such time information can help recommender systems to understand how a user likes an item. Sugiyama et al. proposed some time weighting approaches to adapt search results from search engine [29]. Ding et al. proposed a time weighting approach in collaborative filtering by giving more recent data higher value in the time weighting [28]. But in Farseer, all data items are considered "recent", as all the recommendations we make are within 24 hr after the post of a new item. Thus, a new time weighting method is proposed in Farseer to help improve the recommendation quality.

In Farseer, we compare the time that a user spends on an item with the average time he/she spends on all recent items. This average time of user u is called $t_{avg}(u)$. Users will sometimes be "cheated" by the title of an article in online social communities, if they open an link and find it is not interesting at all, they will return as soon as possible. Thus, we consider the access of an item for less than 5 s as no interest at all. And if the time that user u spends on an item is around $t_{avg}(u)$, we consider it as normal interest, which will be weighted as 1. If the time that user u spends on an item is much more than $t_{avg}(u)$, we should weigh it more. At last, if a user posts or comments on an item, we will weigh it as the most important. Based on our empirical study and detailed cross validation, we define the time weights as follows, where t (seconds) is the time that user u spends on item i.

$$time\_weight(u,i)=\begin{cases}0 & \text{if } t\leqslant 5\\ 1+\Phi(t)_u & \text{if } t>5\\ 2 & \text{if } u \text{ posts/comments on } i\end{cases}$$

where $\Phi(t)_u$ is defined as following:

$$\Phi(t)_u=\begin{cases}0 & \text{if } t\in[0.5*t_{avg}(u),2*t_{avg}(u)]\\ \frac{t-t_{avg}(u)}{2|t-t_{avg}(u)|} & \text{otherwise}\end{cases}$$

3.2.4. Real-time interest group based recommendation

To recommend online users with high-quality items in real-time basis, we need to integrate the user time context extraction and real-time neighbor selection methods into our interest group based recommendation algorithm. Also, as the size of the neighbors increases, we need to make incremental updates to the recommendation results to avoid unnecessary duplicate computation. Our real-time interest group based recommendation algorithm contains the following key steps:

1. *Real-time interest group identification of new items:* In interest group based recommendation, items are recommended within interest groups. Given a newly created data item, we first need to determine the interest group that the item belongs to. This can be determined by calculating its distances to the centers of existing interest groups, using Eq. (3). At this step, the

number of neighbors can greatly influence the accuracy of interest group identification of new items. In our experiments, we find that using 30 neighbors can achieve an accuracy of more than 90%. Usually, after an item is posted, a 3 to 15-min waiting period is sufficient to get 30 neighbors, which is tolerable for most users.

2. *Weighted item rating within interest group:* After an item is assigned to an interest group, we need to determine the goodness of the item inside the interest group, i.e., how people in this interest group like this item. In Farseer, for an item i in interest group g, only the users in g and their contexts containing i can judge the goodness of i inside g. Since users may have varying levels of interests in a specific group, users' ratings of the item should be weighed by their interest levels in the group. Intuitively, for an interest group g, users who like most items in g should have higher weights, and users whose interested items are mostly in g should also have higher weights. The former measure can be defined as $Precision(u,g)$, which is the fraction of items in g that are liked by u. The latter measure can be defined as $Recall(u,g)$, which is the fraction of items liked by u that are in g. The two notions are defined as following:

$$Precision(u,g) = \frac{|I_u \cap I_g|}{|I_g|}, \quad Recall(u,g) = \frac{|I_u \cap I_g|}{|I_u|} \tag{5}$$

As both $Precision(u,g)$ and $Recall(u,g)$ are necessary and important for measuring a user's importance in an interest group, we adopt their combination to weigh the importance of user u in interest group g as follows:

$$Weight(u,g) = \frac{(1+\alpha)Precision(u,g) \times Recall(u,g)}{\alpha \times Precision(u,g) + Recall(u,g)} \tag{6}$$

This combination is an "F-Measure" of the two measures, and we choose $\alpha = 1$ as $Precision(u,c)$ and $Recall(u,c)$ are equally important in our problem. Then, this *weight* should be further combined with the time weight to generate the final weight. We use a linear combination as follows:

$$Final\_weight(u,i) = time\_weight(u,i) * weight(u,g) \tag{7}$$

$Final\_weight(u,i)$ is the user's final weight to item i, g is the interest group that i belongs to. After defining the weight of a user for an item, we can calculate the rating of the item inside the interest group:

$$Rating(i) = \frac{\sum_{u \in U_i} Final\_weight(u,i) \times Y(u,i)}{\sum_{u \in U_i} Final\_weight(u,i)} \tag{8}$$

where U_i is whose contexts contain i, and $Y(u,i)$ is u's judgment of i. $Y(u,i) = 1$ if u likes i, otherwise $Y(u,i) = 0$. Please note that, this rating is the aggregated rating of item i in interest group g, thus, is suitable for all users in g.

3. *Global item ranking:* After the computation of the aggregated local rating for each item within the interest group it belongs to, a global ranking method is needed to rank the items in each subcommunity. The global ranking combines the ratings of items inside their specific interest groups and users' interests in different interest groups. Such global ranking is difficult, as interest groups vary significantly in user/item size, and users may have different levels of interest in different groups. Thus, the local ratings inside interest groups are not directly comparable. To address this issue, we adopt a weighted global ranking method, which weighs the ratings of items from different interest groups by the user's interest levels in the interest groups that contain the items. The final score of an item i for a user u is calculated as following:

$$Score(u,i) = Rating(i) \times Recall(u,g) \tag{9}$$

where g is the interest group that i belongs to. After we obtain the *Score* of item i for user u, we rank the *Score* values across all items, and recommend the top-ranked items to user u.

Note that the local item ratings depend on the selection of neighbors, and the set of neighbors change over time. We would like to incrementally update the local item ratings, users' interest distributions, and the recommendation results. Based on our observation of Fudan BBS subcommunities, most clicks (82–92%) of a new item occur within 24 h after the item is posted. Therefore, we only need to update new item ratings within the first 24 h. Algorithm 3 presents the complete real-time interest group based recommendation algorithm.

Algorithm 3: *RealTimeRecommendation*(U,x,C,τ)

Require: U is the user set, x is a new item, and C is the interest group clustering of users and items, τ is the threshold of neighbor size

```
neighborList = ∅
Run the real-time neighbor selection algorithm for x, and add neighbors to neighborList
while neighborList.size < τ and currentTime < x · postTime + 24 h do
    Wait for 30 s, then update the neighborList
    while currentTime < x · postTime + 24 h and neighborList ≠ ∅ do
        for each c_i ∈ C do
            Calculate the distance between x and the center of c_i
        end for
        Let c be the interest group with the smallest distance from x, and x · Y = 0, x · N = 0
        for each n_i ∈ neighborList do
            if n_i likes x then
                x · Y += Final_weight(n_i, x)
            end if
            x · N += Final_weight(n_i, x); Remove n_i from neighborList
        end for
        x · rating = x · Y / x · N
        for each u ∈ U do
            x · rating(u) = x · rating · w_{u,c}, where w_{u,c} is the fraction of items in c that are liked by u
            Update the global ranking of x for u
        end for
    end while
end while
```

4. Evaluations

In this section, we evaluate Farseer using both offline studies and real-time measurements in Fudan BBS. We consider the eight most active subcommunities, which account for 12.2% post activities and 27.8% view activities of Fudan BBS, and cover a variety of social interests. More importantly, these eight subcommunities have diverse internal structures. Implicit interest groups may or may not exist in each subcommunity. Together, these eight subcommunities allow for a comprehensive evaluation of the proposed content recommender system. The overall characteristics of the eight subcommunities are listed in Table 1.

We first analyze the proposed interest-based clustering algorithm, which is then used to study the internal structures, i.e., interest groups, of different subcommunities. Next, we evaluate the online recommendation performance of Farseer using

real-time studies. The experimental results demonstrate that the proposed system can effectively identify personal interests and improve the quality and efficiency of real-time personalized content recommendation in online social communities, compared with other CF algorithms.

4.1. Evaluation of interest group based clustering

Number of Clusters. As described in Section 3.1, a key challenge of existing clustering methods is determining the number of clusters. Many clustering based applications rely on an empirical cluster number [16,26,3]. In our work, the proposed interest-based content–user clustering algorithm can efficiently determine the appropriate number of clusters using the *Objective* function (see Section 3.1.1). Our study also shows that, for a subcommunity with k internal interest groups, even if the number of clusters exceeds the optimal number k, the recommendation quality remains comparable to the optimal case. This implies that the proposed clustering approach is resilient to the noise of subcommunity structures, and the recommender system is able to provide robust and high-quality recommendation results. This is demonstrated in Fig. 4. When applying the clustering algorithm to a subcommunity with 16 interest groups, Fig. 4 (1) shows that, as the number of clusters k increases from 1 to 16 (the optimal clustering), the recommendation quality, i.e., precision and recall, improves consistently. Meanwhile, the *Objective* value (shown in parenthesis) also increases and reaches its maximum (0.261) when $k = 16$. Fig. 4 (2) shows that, starting at $k = 16$, further increase of k only results in slight degradation of recommendation quality, so does the *Objective* value. This is because that the increase of k will break large interest groups into smaller ones, but the smaller interest groups are still accurate in capturing user interest and make accurate recommendations to users. A significant quality degradation is observed only when k dramatically deviates from the optimal setting, e.g., $k = 400 \gg 16$. When k is dramatically large, there are only a few users and items in most of the interest groups, and insufficient size of neighbors will lead to bad recommendations. Overall, this study demonstrates that *Objective* provides an accurate estimation for optimal clustering, and the proposed clustering algorithm can consistently achieve high-quality results.

Comparison with manual clustering. To further evaluate the proposed interest group clustering method, we combine multiple similar subcommunities which are manually clustered/created by users, and apply the proposed method to the aggregated subcommunity. We can then compare the recommendation quality of our automatic clustering method to that of manual clustering (using individual subcommunities). In Fig. 5(a), the aggregated subcommunity consists of four game-related subcommunities, including *Online game*, *PC game*, *RTS game* and *Sports game*. In Fig. 5(b), the aggregated subcommunity consists of four entertainment-related subcommunities, including *Movie*, *Music*, *TV*, and *Cartoon*. As shown in the figures, our proposed method can effectively distinguish different interest groups within the aggregated subcommunity, thereby achieving comparable performance to that of manual clustering. Please note that manual clustering reflects real-world user interest categories and is the optimal case that clustering algorithms can achieve. Therefore, we can conclude that the proposed interest group identification algorithm is comparable to the best clustering in terms of recommendation quality.

4.2. Online recommendation performance

Next, we evaluate the online recommendation performance of Farseer using a 30-day data set collected from the eight subcommunities in Fudan BBS as presented at the beginning of this section. We use the first 20-day data as the training set and the last 10-day data as the test set. We compare Farseer against three well-known and state-of-the-art CF algorithms.

- **kNN** (k nearest neighbor [9]) is a well-known memory-based collaborative filtering algorithm. Given a target user, the server first identifies the k most similar users/items as the neighbors. It then calculates the recommendations for the user based on the weighted majority vote of the neighbors. This method is simple, but suffers from the data sparsity problem.
- **PLSA** (probabilistic latent semantic analysis based collaborative filtering) is a model-based CF algorithm first proposed by Hofmann [20], and later adopted by Google News [6]. In this method, the server can determine the ratings of items for a target user with the maximum likelihood. This method is accurate, but building the models can be time-consuming.
- **SVD** (singular value decomposition based collaborative filtering method with factor analysis [27]) is an optimized SVD-based CF algorithm. It adopts an expectation maximization (EM) recurrence to achieve the factor analysis. After factor analysis and singular value decomposition, the server can calculate the missing values in the user-item rating matrix based on the user's feature vector. This method is fast, accurate, and robust against the data sparsity problem.

Our comparison focuses on recommendation quality and efficiency in online real-time content recommendation. We also study the run time latency-quality tradeoff of Farseer. We would like to point out that the proposed clustering method and real-time recommendation strategies are mostly orthogonal to, and can thus be adopted by, existing CF methods to improve the performance of personalized real-time content recommendation.

Online Recommendation Quality. Fig. 6 compares the online recommendation quality of Farseer, kNN, PLSA, and SVD. As shown in the figure, kNN, PLSA and SVD achieve similar but lower quality than that of Farseer. Farseer can outperform the three algorithms mainly due to: (1) Farseer adopts interest based recommendation, which means recommendations are only made by "expert" users. Users who are not interested in that kind of items will not introduce any noise to the item ratings; (2) Farseer adopts real-time user context extraction, which can reduce irrelevant items that are not contained in user sessions; (3) in real-time recommendation, the three algorithms suffer from the "false negative" problem that we discussed in Section 3.2, and the inaccurate user ratings lead to bad recommendation accuracy; and (4) the time weight strategy can better understand user interest and help improve recommendation accuracy, which will be shown later. This experi-

Table 1
Characteristics of Eight Subcommunities in Fudan BBS.

Subcommunity	Astrology	Auto	Joke	Movie	Music	TV	Football_WD	TVEntZ
ID	1	2	3	4	5	6	7	8
# of users	3,621	1,514	5,768	597	223	303	1,641	450
# of posts	678	565	561	119	139	265	1,064	263
# of views	58,726	21,413	146,933	4,067	4,528	3,041	78,012	7,259

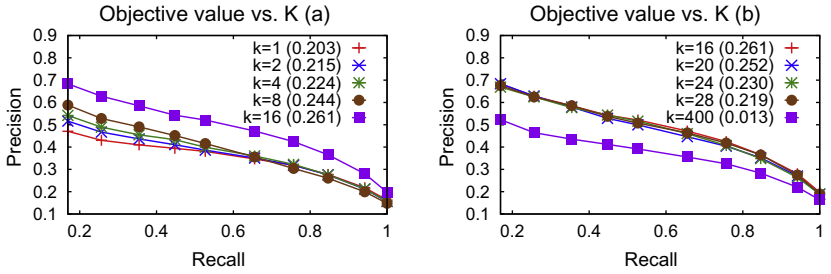

Fig. 4. Accuracy of interest-based clustering in a subcommunity with 16 interest groups. *Objective* values of different k are shown in parenthesis.

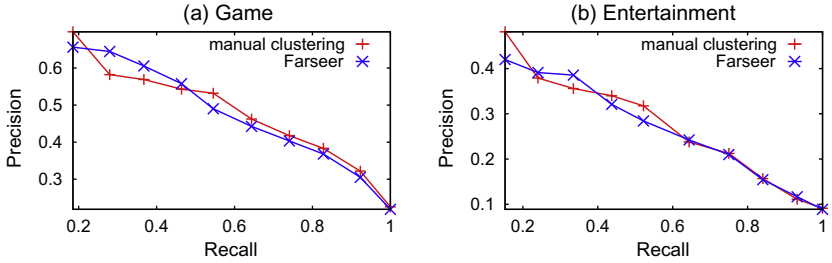

Fig. 5. Comparison of Farseer with manual clustering of combined subcommunities: (a) Game – Online game, PC game, RTS game, and Sports game; (b) Entertainment – Movie, Music, TV, and Cartoon.

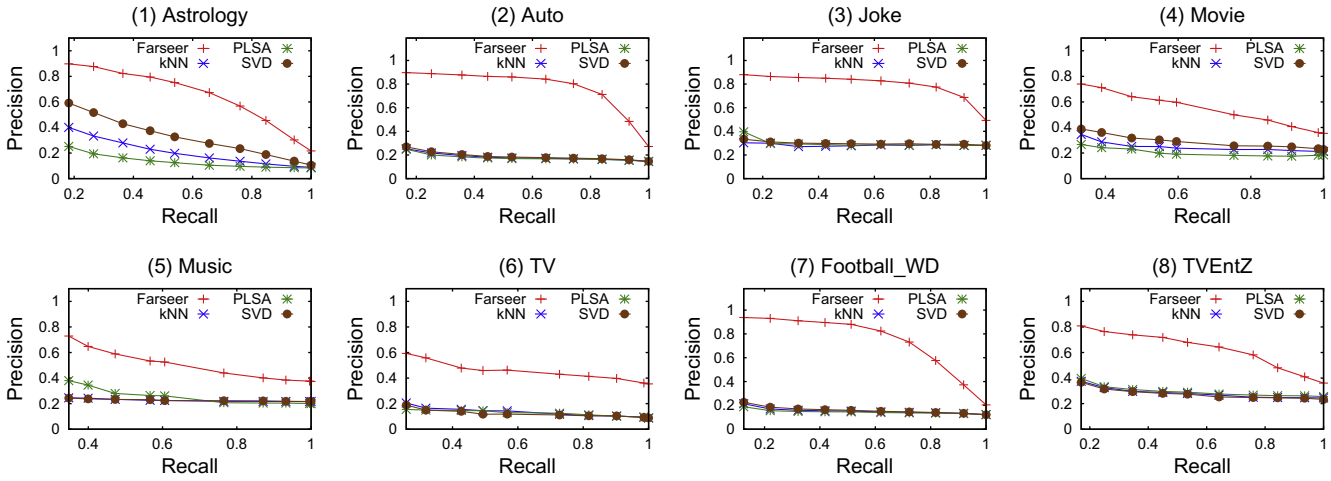

Fig. 6. Recommendation quality comparison of Farseer, kNN, PLSA, and SVD in 8 subcommunities.

ment demonstrates that our interest-based real-time content recommendation algorithm indeed captures user online activity/interest and can improve the recommendation quality with "accurate" neighbors and ratings. As shown in Fig. 6, Farseer consistently outperforms kNN, PLSA and SVD in all eight subcommunities, with an average improvement of 185%, 213%, and 167%, respectively. We also tested with other subcommunities in Fudan BBS and achieved very similar results.

Run-time recommendation latency. During online content recommendation, for each newly posted item, Farseer leverages the opinions of recent online users which have examined the item, also called the neighbor group, for clustering and recommendation decisions. As the number of neighbors increases, the recommendation quality improves, but the recommendation latency also increases, as the system needs to wait longer for more neighbors to become available. Therefore, a tradeoff between recommendation quality and latency needs to be made. In Fig. 7, we can see that as the number of neighbors increases from 10 to 30, the recommendation quality, i.e., precision-recall curve, increases. And the quality of 30-neighbor is very close to the optimal case which uses all neighbors. For most items in the different subcommunities, after an item is posted, a 3- to 15-min waiting period is sufficient

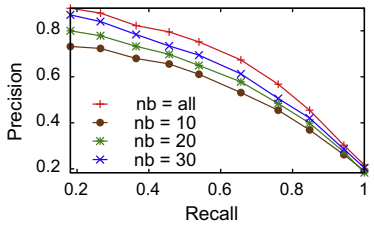

Fig. 7. Recommendation quality vs. number of neighbors (nb).

to obtain 30 neighbors. Thus using 30 neighbors can achieve a good tradeoff between recommendation quality and online recommendation latency.

Effect of time weight. In Farseer, through user time context analysis, we can determine the exact amount of time a user spends on an item. Compared with binary like/dislike, such time information enables more precise characterization of a user's interest in an item (Section 3.2.3). To evaluate the effect of time weight, we compare the recommendation quality in two representative subcommunities, *Astrology* and *Football_WD*, in Fig. 8. In *Football_WD*, users are usually active during game time. Thus, time information is important in measuring their interests in items. As shown in Fig. 8 (b), recommendation with time weight consistently outperforms recommendation without time weight, with an improvement of 7% on average. But in *Astrology*, users only read a few items every day, and the amount of time they spend on different items do not vary much. As a result, recommendation with time weight has limited improvement over recommendation without time weight (2% on average). In summary, using time weight can help improve the recommendation quality, especially in subcommunities with diverse user time contexts.

Efficiency of online content recommendation. Fig. 9 compares the computation time between our method and the three other CF algorithms for online content recommendation. As Farseer adopts interest-based recommendation, fewer neighbors are considered compared with the three CF algorithms. Also, Farseer adopts real-time user context selection and neighbor selection, which further reduces the number of items to be considered. Thus, Farseer can reduce the computation complexity compared with the three other CF algorithms. As shown in the figure, our method consistently outperforms the three CF algorithms in all the eight subcommunities. On average, Farseer requires 35% less computation than kNN, 16% less computation than SVD, and only about 1/5 computation of PLSA. In other words, Farseer provides not only better-quality recommendations, but also much faster recommendations than existing CF algorithms, thus is more suitable for real-time content recommendation in online social communities.

5. Related work

In this work, we propose a personalized real-time recommender system for online social communities. Our work builds upon existing collaborative filtering (CF) techniques. Since Goldberg et al. [12] first introduced CF as an alternative to content-based filtering [14,15], CF has been widely used in recommender systems because of its high prediction quality.

Existing CF techniques can be classified as memory-based [9,19,11] or model-based [6,20,13], depending on whether a model is constructed from raw user ratings. Different from the work of Das et al. [6], which targets the most popular news items using weighted majority voting among multiple user communities, our work aims to identify items which may have low popularity but match the unique interests of individual users. Different from that work, our weighted majority voting are only conducted among interested users, thus reducing the "noisy" voting from uninterested or non-expert users. CF techniques can also be classified as user-based [9,10,25,24,23] or item-based [7,8], depending on whether the similarity is based on user or item. Our work unifies both item-based and user-based methods, namely interest group based user-item clustering and cluster-based content recommendation, which classifies items and users into interest groups. This approach provides consistent performance improvement in online social communities with dynamic content and user interests.

Clustering algorithms have been used in both user-based and item-based recommender systems [3,5,1,2,4]. Our work shares similar motivation with recent researches on cluster-based recommender systems [16,26]. We stress that user similarity should not be computed on the complete item set, as users may share similar interests, or item subsets, with different users. Our work differs from those works in both clustering algorithm and recommendation strategy design. Our clustering method is able to determine the optimal number of clusters, which was not addressed in those works. In addition, those works did not consider the real-time recommendation problem, while our work leverages user time context information and targets the real-time recommendation scenario.

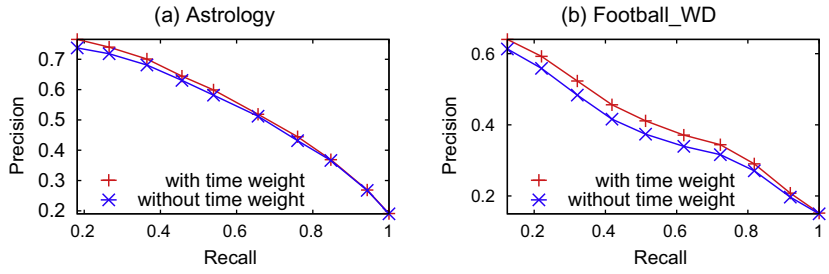

Fig. 8. Comparison of recommendation quality with and without time weight in two subcommunities.

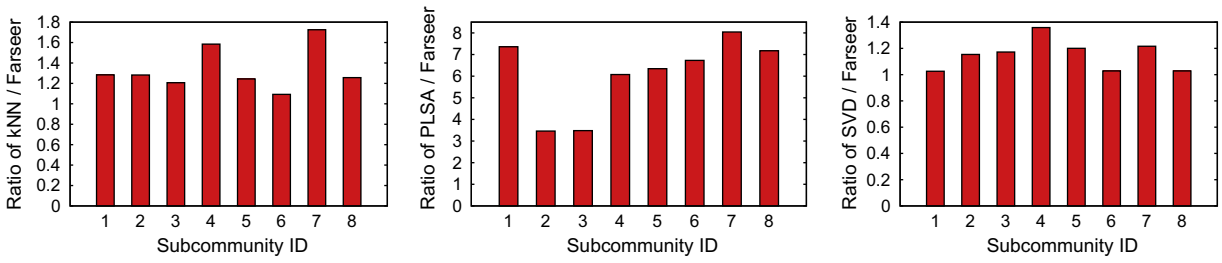

Fig. 9. Comparison of computation time between Farseer and other CF algorithms in 8 subcommunities.

The user interest identification algorithm proposed in this work shares similar essences with existing clustering algorithms, such as *Y*-Means [31] and Genetic *K*-Means [32]. But our algorithm is different mainly in two ways: (1) the goal of our algorithm is to find the best interest groups for content recommendation, and a good clustering with minimum squared errors in other algorithms may not be the best clustering for content recommendation; and (2) since we cluster items and users simultaneously, the global *Objective* function that we optimize is related to both items and users, which is different from the optimization process in other clustering algorithms.

User interest analysis has also been studied in both recommender systems and online social communities. Yang et al. [33] proposed a similarity measure based on rational inference of users, which can measure user similarity on different interests. But they did not address the question of how to find user interests, which is one of the main contributions of this work. Zeng et al. [34] proposed a user interest model, which is built by content analysis, to analyze user activity on the Web. Different from their work, our user interest analysis focuses on popularity distribution, which is another aspect of user interest but very critical to recommendation algorithm design in online social communities.

6. Conclusions and future work

This article addresses the challenge of personalized real-time content recommendation in online social communities. It overcomes a key limitation of existing collaborative filtering techniques – favoring highly popular content over less popular ones. The proposed solution unifies item-based and user-based collaborative filtering techniques, which improves the accuracy of user interest characterization, hence content recommendation quality. In addition, the proposed system aims for real-time content recommendation. It leverages the time context information of online user activities, and dynamically adjusts the user interest levels to optimize the recommendation quality for real-time content recommendation. The proposed system is fully implemented and evaluated via real-world system deployment in an online social community. Experimental studies demonstrate consistent performance and efficiency improvement over existing collaborative filtering techniques. Our future work includes an ongoing study targeting larger-scale and more diverse online social communities. Another ongoing work focuses on leveraging explicit user social relationships and social interactions to facilitate content recommendation.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China under Grant Nos. 60736020 and 60803118, the Shanghai Leading Academic Discipline Project under Grant No. B114, and the National Science Foundation of USA under Grant No. CNS – C0910995.

References

[1] Mark O'Connor, Jon Herlocker, Clustering items for collaborative filtering, in: ACM SIGIR Workshop on Recommender Systems, ACM press, 1999.
[2] L. Ungar, D. Foster, Clustering methods for collaborative filtering, in: Proceedings of the Workshop on Recommendation Systems. ACM, AAAI Press, Menlo Park California, 1998.
[3] Gui R. Xue, Chenxi Lin, Qiang Yang, WenSi Xi, Hua J. Zeng, Yong Yu, Zheng Chen. Scalable collaborative filtering using cluster-based smoothing, in: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '05, ACM, 2005 pp. 114–121.
[4] Junjie Wu, Hui Xiong, Jian Chen, Adapting the right measures for *k*-means clustering, in: KDD '09: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2009 pp. 877–886.
[5] David Arthur, Sergei Vassilvitskii, *k*-means++: the advantages of careful seeding, in: SODA '07: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, 2007 pp. 1027–1035.
[6] Abhinandan S. Das, Mayur Datar, Ashutosh Garg, Shyam Rajaram, Google news personalization: scalable online collaborative filtering, in: Proceedings of the 16th International Conference on World Wide Web, WWW '07, ACM, 2007 pp. 271–280.
[7] G. Linden, B. Smith, J. York, Amazon.com recommendations: item-to-item collaborative filtering, IEEE Internet Computing 7 (1) (2003) 76–80.
[8] Badrul M. Sarwar, George Karypis, Joseph A. Konstan, John Reidl, Item-based collaborative filtering recommendation algorithms, in: World Wide Web, ACM, 2001 pp. 285–295.
[9] Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, John Riedl, An algorithmic framework for performing collaborative filtering, in: SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1999 pp. 230–237.
[10] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, John Riedl, GroupLens: an open architecture for collaborative filtering of netnews, in: Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work, CSCW '94, ACM, 1994 pp. 175–186.
[11] Joseph A. Konstan, Bradley N. Miller, David Maltz, Jonathan L. Herlocker, Lee R. Gordon, John Riedl, Grouplens: Applying collaborative filtering to Usenet news, Communications of the ACM 40 (3) (1997) 77–87.
[12] David Goldberg, David Nichols, Brian M. Oki, Douglas Terry, Using collaborative filtering to weave an information tapestry, Communications of the ACM 35 (1992) 61–70.
[13] John S. Breese, David Heckerman, Carl Kadie, Empirical analysis of predictive algorithms for collaborative filtering, in: Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence (UAI-98), July 1998 pp. 43–52.
[14] Balabanović, Marko, Shoham, Yoav, Fab: content-based, collaborative recommendation. Communications of the ACM vol. 40, 1997 pp. 66–72.
[15] Michael Pazzani, Daniel Billsus, Learning and revising user profiles: the identification of interesting web sites, Machine Learning 27 (3) (1997) 313–331.
[16] Truong Khanh Quan, Fuyuki Ishikawa , Shinichi Honiden, Improving accuracy of recommender system by clustering items based on stability of user similarity, in: Proceedings of 2006 International Conference on Computational Intelligence for Modelling Control and Automation, CIMCA '06, IEEE Computer Society, 2006 pp. 61–68.
[17] Robert Cooley, Bamshad Mobasher, Jaideep Srivastava, Data preparation for mining world wide web browsing patterns, Knowledge and Information Systems 1 (1999) 5–32.
[18] J.B. MacQueen, Some methods for classification and analysis of MultiVariate observations, in: Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability, University of California Press, 1967, pp. 281–297.
[19] Gediminas Adomavicius, Alexander Tuzhilin, Toward the next generation of recommender systems: a survey of the State-of-the-Art and possible extensions, IEEE Transactions on Knowledge and Data Engineering 17 (6) (2005) 734–749.
[20] Thomas Hofmann, Latent semantic models for collaborative filtering, ACM Transaction on Information Systems 22 (1) (2004) 89–115.
[21] Wen Y. Chen, Jon C. Chu, Junyi Luan, Hongjie Bai, Yi Wang, Edward Y. Chang, Collaborative filtering for orkut communities: discovery of user latent behavior, in: Proceedings of the 18th International Conference on World Wide Web, WWW '09, ACM, 2009 pp. 681–690.
[22] Cong Yu, Laks Lakshmanan, Sihem A. Yahia, It takes variety to make a world: diversification in recommender systems, in: Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology, EDBT '09, ACM, 2009 pp. 368–378.
[23] J. Bobadilla, F. Ortega, A. Hernando, J. Alcalá, Improving collaborative filtering recommender system results and performance using genetic algorithms, Knowledge-Based Systems 24 (8) (2011) 1310–1316.
[24] J. Bobadilla, F. Serradilla, J. Bernal, A new collaborative filtering metric that improves the behavior of recommender systems, Knowledge-Based Systems 23 (6) (2010) 520–528.
[25] J. Bobadilla, F. Serradilla, A. Hernando, Collaborative filtering adapted to recommender systems of e-learning, Knowledge-Based Systems 22 (4) (2009) 261–265.
[26] Panagiotis Symeonidis, Alexandros Nanopoulos, Apostolos Papadopoulos, Yannis Manolopoulos, Nearest-Biclusters collaborative filtering, in: Proceedings of KDD Workshop on Web Mining and Web Usage Analysis, WebKDD '06, ACM, August 2006.
[27] John Canny, Collaborative filtering with privacy via factor analysis, in: SIGIR '02: Proceedings of the 25th annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2002 pp. 238–245.
[28] Yi Ding, Xue Li, Time weight collaborative filtering, in: Proceedings of the 14th ACM International Conference on Information and Knowledge Management, CIKM '05, ACM, 2005 pp. 485–492.
[29] Kazunari Sugiyama, Kenji Hatano, Masatoshi Yoshikawa, Adaptive web search based on user profile constructed without any effort from users, in: Proceedings of the 13th International Conference on World Wide Web, WWW '04, ACM, 2004 pp. 675–684.

[30] John H. Holland, Adaptation in Natural and Artificial Systems, The University of Michigan Press, 1975. p. 183.
[31] Ali A. Ghorbani, Iosif-Viorel Onut, *Y*-means: an autonomous clustering algorithm, Hybrid Artificial Intelligence Systems 6076 (2010) 1–13.
[32] K. Krishna, M. Narasimha Murty, Genetic *K*-means algorithm, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 29 (3) (1999) 433–439.
[33] J.-M. Yang, K.F. Li, D.-F. Zhang, Recommendation based on rational inferences in collaborative filtering, Knowledge-Based Systems 22 (1) (2009) 105–114.
[34] J. Zeng, S. Zhang, C. Wu, A framework for WWW user activity analysis based on user interest, Knowledge-Based Systems 21 (8) (2008) 905–910.