Improved Neighborhood Search for Collaborative Filtering

Yeounoh Chung¹, Noo-ri Kim², Chang-yong Park³, and Jee-Hyong Lee²

¹Department of Computer Science, Brown University, Rhode Island, USA

²Department of Electrical and Computer Engineering, Sungkyunkwan University, Suwon, Korea

³LG Electronics, Seoul, Korea

Ijfis

Abstract

k -Nearest Neighbor (k -NN) and other user-based collaborative filtering (CF) algorithms have gained popularity because of the simplicity of their algorithms and performance. As the performance of such algorithms largely depends on neighborhood selection, it is important to select the most suitable neighborhood for each active user. Previous user-based CF simply relies on similar users or common experts in this regard; however, because users have different tastes as well as different expectations for expert advice, similar users or common experts may not always be the best neighborhood for CF. In search of a more suitable neighborhood, so-called personalized experts develop personalized expert features. Through experimentation, we show that personalized experts are different from similar users, common experts, or similar common experts. The personalized, expert-based CF algorithm outperforms k -NN and other user-based CF algorithms.

Keywords: Personalized experts, Recommender system, Collaborative filtering, Support vector machine

1. Introduction

With the success of many e-commerce services (e.g., Amazon, Netflix, Last.fm), recommender systems have gained significant interest and popularity in recent years, and significant effort has been dedicated to researching and building better recommender systems and algorithms [1]. One of the most popular algorithms for recommender systems is collaborative filtering (CF), which simply finds patterns among similar users or items [2]. CF achieved widespread success because of its simplicity and efficiency, despite several drawbacks (e.g., the sparsity problem) [3–7].

Typical CF (neighborhood or user-based CF) develops user profiles based on the item-consumption profiles of those users and provides personalized recommendations to active (or target) users based on a combination of similar user profiles. CF is based on the simple assumption that users with tastes that are similar to the active user may give more useful information, which may lead to a better recommendation. However, in some cases, users with similar tastes may not give useful information because the active user may have already consumed what the similar users have consumed. Ideally, to obtain the best performance, CF algorithms require users who can provide useful information for recommendations, and they may not necessarily be similar users.

Received: Feb. 1, 2018
 Revised : Mar. 10, 2018
 Accepted: Mar. 21, 2018

Correspondence to: Jee-Hyong Lee
 (john@skku.edu)
 ©The Korean Institute of Intelligent Systems

©This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (<http://creativecommons.org/licenses/by-nc/3.0/>) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

In parallel to similar user-based CF, expert-based CF and recommender systems have been proposed. General users who lack domain knowledge often trust more reliable and knowledgeable experts when making decisions to purchase items. A study conducted in the field of retail and marketing shows that consumers regard expert opinions as more reliable [8]. In agreement with this observation, several recent studies have exploited the knowledge of experts [9–18]. Those approaches are based on the assumption that users with more expertise may give more useful information, which will lead to more accurate recommendations. Expert-based CF can be more robust for situations where there are not enough item-consumption histories available from which to draw similarities between users (i.e., the sparsity problem) than similar user-based CF [9, 12]. However, expert-based CF is limited in that the experts can only recommend items that are generally popular. In other words, the recommendations are less customized.

In this work, we seek to find a better neighborhood for user-based CF and to combine the merits of both user-based and expert-based approaches. The notion of personalized experts as the better neighborhood from which to provide useful information was first proposed in our previous works [19, 20]. However, personalized expertise was expressed in crudely developed features for support vector machine (SVM) model training and yielded less accurate recommendations than k -Nearest Neighbor (k -NN). Here, we examine the notion of personalized expertise in various aspects and carefully design new expertise features to identify personalized experts for users with various profiles and preferences. Notably, our new personalized expert-based recommender system outperforms k -NN in terms of prediction accuracy. Furthermore, we present a better learning process for a single global SVM model to find customized expert groups for each user, without any given expert labels or explicit user feedback. The key idea is to train an SVM model to learn the mapping between different user profiles and the most beneficial groups of neighbors. In [19], we proposed to search personalized experts among similar users. This reduced the cost of training, but also bounded the personalized experts to similar users (what if a user does not want any suggestions from similar neighbors?). Instead, we refined the expert pool to be users with any expertise characteristics (e.g., early adopter, heavy access, niche-item access) and select more diversified personalized experts from them.

Our approach is expert-based, but unlike previous expert-based approaches, different experts are chosen for each active user to accommodate different needs. Some users prefer similar

users; others prefer early adopters or even users with very eccentric tastes. Furthermore, this personalized expert identification problem is more thoroughly studied to yield a machine learning solution by using a SVM. The resulting recommendations from personalized expert-based CF prove to be more accurate than k -NN and expert-based CF systems and more customized than expert-based CF.

The rest of this study is organized as follows. In Section 2, we briefly discuss previous user-based CF algorithms. In Section 3, we describe the personalized expert identification problem along with the personalized expertise measures in detail. In Section 4, we present the experimental results and analysis. In Section 5, we further discuss the robustness of the proposed recommender system considering the sparsity problem. Finally, we conclude in Section 6.

2. Related Work

A recommender system based on a k -NN CF algorithm relies on collaborative opinions of a neighborhood with similar user profiles computed from item consumption histories. Because recommendations are generated based on user profiles alone, similar user-based recommender systems result in accurate recommendations for various users. However, the recommendations may be inaccurate if the item-consumption histories are not sufficient to build rich user profiles [4–6]. This lack of information in item-consumption histories is referred to as the sparsity problem, and it is one of the most limiting factors for its performance in practice. Many techniques, ranging from dimension reduction to sparse data smoothing, were proposed to address this issue [4, 21–24].

To alleviate the sparsity problem and build a better recommender system, several researchers have suggested expert-based CF. Papagelis et al. [6] shows that expert profiles from a movie review website can be used to model user profiles of a much larger user group. By CF of the opinions of similar external experts, the authors were able to produce comparable recommendations to k -NN. Similarly, other external expert-based CF algorithms used external expert knowledge identified from web blogs or real human participants who can provide dynamic feedback for recommendations [11, 16]. This type of external expert-based CF is robust to the sparsity problem; however, it is very expensive to source expert knowledge in most cases, which may limit the scalability of the applications.

Instead of using external expert knowledge, other researchers focused on identifying experts among active users. As the

performance of CF algorithms largely depends on neighbor selection (i.e., the source of collective opinions in CF), defining and identifying appropriate experts is important for successful expert-based recommender systems [10–15, 17, 18]. The expert groups used in those works are early adopters, personal innovators, and users with highly common expert measures.

Song et al. [14] proposes three common expert measures and identifies a set of common experts from an active user group. Because the same common experts are used in CF for all active users, the resulting recommendations are less personalized. Similarly, Lee and Lee [12] identified common experts per similar item group in their recent work. Their approach suggests different expert groups for different item groups, but recommendations are still not personalized with respect to the active users.

3. Personalized Expert Search

Instead of simply choosing similar users, our approach chooses different experts for each active user who can better accommodate various needs and expectations. We define personalized experts per each active user as neighbors who are the most resourceful for CF-based recommendations. To efficiently determine whether a neighboring user is a personalized expert or not, we train a single global SVM model that learns the matching pattern between personalized experts and active users. Because the task is not just finding similar user profiles, the matching pattern can be complicated, and generating an accurate SVM learner to solve this personalized expert identification problem is challenging. In the following subsections, we discuss three challenges and the solutions for them.

3.1 How to Label Training Data?

Training an accurate SVM learner to find personalized experts for active users requires training data with labels—these labels should identify which experts belong to whom. Because such labels are not available (i.e., it is very expensive to receive explicit feedbacks from users), we approximate the labels with a random search.

First, we define a personalized expert group for an active user as a set of users who give the most accurate recommendations. With this definition, and by only using the training data, we select a group of users of a fixed size, called V_{ui} , at random for each active user, u_i , to carry out CF with the group and evaluate performance increases. For each iteration, we randomly switch one user in V_{ui} with one user not in V_{ui} . If the new V_{ui}

yields better recommendation accuracy, the new V_{ui} is accepted. This random search procedure repeats for a fixed number of iterations, and the final V_{ui} is used as an approximated personalized expert group for u_i . However, this technique is too costly from a computational perspective. To reduce the computational complexity, we assume that the personalized experts exhibit several degrees of common expertise that are accepted by the general population; in other words, we reduce the search space to a handful of users with a higher expertise. The expertise measures are defined in the next subsection.

This generic random search algorithm is simple and yet very useful for obtaining a near-optimal solution. In solving ill-structured global optimization problems with many potential stationary points, a random search ensures convergence to a global optimum in terms of probability. Essentially, if the random selection does not ignore any part of the search space, then the algorithm is guaranteed to converge with a probability one [25]. As it follows a geometric distribution, the number of expected iterations until near-optimal convergence (within distance from the optimum) is as follows:

$$E[N(V_{ui}^* + \varepsilon)] = \frac{1}{p(V_{ui}^* + \varepsilon)}. \quad (1)$$

Finding the optimum is still very expensive for a practical recommender system, even with the search space reduction. In this work, we limit the number of iterations for finding personalized experts to 1,000, which is empirically shown to be sufficient.

3.2 How to Describe Personalized Expertise?

To extract a meaningful matching pattern, we carefully develop features to represent the relationship between any pair of users. The personalized expertise feature vector, X_{ij} , indicates how an active user, u_i , views a neighbor, u_j . We measure such a pair-wise view with absolute and relative measures. The absolute expertise measures describe how much a neighbor u_j is generally accepted as an expert, and the relative measures are used to represent the information of u_j with respect to an active user, u_i .

We express absolute expertise measures with four features: *Early Adopter*, *Heavy Access*, *Niche-Item Access* and *Eccentricity*. *Early Adopter*, *Heavy Access* and *Niche-Item Access* are common expertise measures [14] and *Eccentricity* indicates how eccentric and unique a user is. However, relative expertise measures are defined between a pair of users; namely, an ac-

tive user and a neighbor. They are expressed in three features: *Similarity*, *Common-Item Access*, *Unknown-Item Access*.

In the following expression, we define the expertise measures used to express different notions of neighborhood expertise:

$$\vec{X}_{ij} = \langle EA(u_j), HA(u_j), NA(u_j), EC(u_j), PR(u_i, u_j), CA(u_i, u_j), UA(u_i, u_j) \rangle. \quad (2)$$

Early Adopter ($EA(u_i)$) uses new items before others and their opinion can have influence. It measures how long it takes for u_i to access newly released items on average. Given reference time (T_R), item released time of m (T_m), item rated time of u_i ($T_{u_i, m}$), the list of items that u_i accessed ($I(u_i)$), we compute $EA(u_i)$ as follows:

$$EA(u_i) = \sum_{m \in I(u_i)} \frac{T_R - T_{u_i, m}}{|I(u_i)|}. \quad (3)$$

Heavy Access ($HA(u_i)$) measures how many items a user accessed. In general, more experience means more expertise:

$$HA(u_i) = \log(|I(u_i)| + 1). \quad (4)$$

Niche-Item Access ($NA(u_i)$) measures the average unpopularity of accessed items. In a sense, users who find hidden items that are not popular are ad hoc experts. Given the list of users who accessed item m , $U(m)$, we compute $NA(u_i)$ as follows:

$$NA(u_i) = \sum_{m \in I(u_i)} \frac{\log 2}{\log(|U(m)| + 1)} / |I(u_i)|. \quad (5)$$

Eccentricity ($EC(u_i)$) measures the average item preference deviation from the popular beliefs or the population mean. Some believe that experts must have different and more eccentric views on matters than the rest of the world. Given the average rating on m ($\bar{R}_m$), the actual rating of u_i on m ($R_{u_i, m}$), the upper bound of rating values ($R_{\max}$), the lower bound of rating values ($R_{\min}$), we compute $EC(u_i)$ as follows:

$$EC(u_i) = \sum_{m \in I(u_i)} \frac{\log(|\bar{R}_m - R_{u_i, m}| + 1)}{\log(R_{\max} - R_{\min})} / |I(u_i)|. \quad (6)$$

Similarity ($Sim(u_i, u_j)$) measures the similarity between two user profiles. It is measured with the Pearson correlation coefficient of the ratings of the two users. Users with similar item preferences may be more helpful; but some users may prefer users with very different item preferences, so we consider

$PR(u_i, u_j)$ in our neighbor search:

$$Sim(u_i, u_j) = \frac{\sum_{m \in I(u_i) \cap I(u_j)} (R_{u_i, m} - \bar{R}_{u_i})(R_{u_j, m} - \bar{R}_{u_j})}{\sqrt{\sum_{m \in I(u_i) \cap I(u_j)} (R_{u_i, m} - \bar{R}_{u_i})^2 \times \sum_{m \in I(u_i) \cap I(u_j)} (R_{u_j, m} - \bar{R}_{u_j})^2}}. \quad (7)$$

Common-Item Access ($CA(u_i, u_j)$) is different from similarity. A user may trust other users with the same item experiences. If two users consume exactly the same set of items and both users like or dislike the same item, the similarity will be high. However, $CA(u_i, u_j)$ will be high if the number of commonly accessed items is large:

$$CA(u_i, u_j) = \log(|I(u_i) \cap I(u_j)| + 1). \quad (8)$$

Unknown-Item Access ($UA(u_i, u_j)$) measures how many new items u_j has accessed, of which u_i has no prior knowledge. u_i may prefer neighbors with more experience with new items:

$$UA(u_i, u_j) = \log(|I(u_i) - I(u_j)| + 1). \quad (9)$$

3.3 How to Train SVM on Class-Imbalanced Data?

The performance of personalized expert-based CF largely depends on the qualifications of personalized experts; thus, the classification accuracy of the SVM learner is very important. One of the biggest concerns in approximating expert labels is that the number of personalized experts for u_i is very small compared to the number of the entire user group. As a result, the accuracy of an SVM learner trained on such an imbalanced training data is degraded [26]. To cope with this, we use the cost sensitive support vector machine (C-SVM) learner [20], which assigns different training error penalties to different classes to effectively learn from imbalanced data [27]. The personalized expert identification problem transformed into an SVM optimization problem is as follows:

$$\begin{aligned} & \underset{\vec{W}}{\text{minimize}} \frac{1}{2} \vec{W} \cdot \vec{W} + (C_+ + C_-) \cdot \sum_i \sum_j \varepsilon_{ij} \\ & \text{subject to } y_{ij}(\vec{W} \cdot \vec{X}_{ij} + b) \geq 1 - \varepsilon_{ij}, \quad \varepsilon_{ij} \geq 0, \end{aligned} \quad (10)$$

where C_+ and C_- control the trade-off between training errors and margin maximization for positive and negative examples, respectively. By tuning the cost factor, C_+/C_- , one can more effectively learn from class imbalanced data.

4. Experiment

In this section, we present experimental results to show that personalized expert-based CF can produce better recommendations than similar user- or common expert-based CF recommender systems. We use MovieLens data sets to accomplish this. The data sets are widely used in recommender systems and CF studies, and they are compiled and collected over various periods of time [4]. Specifically, we use MovieLens 100k data set (ML-100k), which contains 100,000 ratings from 943 users and 1,682 items. We divide the data set into five folds for cross-validation.

4.1 Evaluation Metrics

Different CF algorithms and recommender systems exhibit different performance characteristics, and several properties of recommender systems are traded-off at the expense of the other properties. Therefore, various performance metrics must be used to evaluate CF algorithms [6]. In this work, we consider both prediction accuracy evaluation and recommendation list evaluation.

The prediction accuracy is by far the most common and important performance evaluation metric in recommender system evaluation. To evaluate any CF based recommender systems in prediction accuracy, we use the *Mean Absolute Error*.

Mean Absolute Error (*MAE*) measures the average difference between the predicted ratings and the actual ratings. *MAE* for u_i is calculated as follows:

$$MAE(u_i) = \sqrt{\frac{\sum_{m \in I(u_i)^{test}} |\hat{R}_{u_i, m} - R_{u_i, m}|}{|I(u_i)^{test}|}}. \quad (11)$$

Here, $\hat{R}_{u_i, m}$ is the predicted rating of u_i on m , and $I(u_i)^{test}$ is the accessed item lists of u_i for the items in the test data. *MAE*(u_i) of all users are then averaged to evaluate the *MAE* of recommender systems.

Recommendation list evaluation is important for studying various properties of recommender systems. In this domain, we consider *Item Coverage*, *User Coverage*, *Diversity*, *Precision* and *Recall* of returned recommendations.

Item Coverage (*Cov_{item}*) measures the proportion of items that a recommender system can recommend from the entire item space:

$$Cov_{item} = \frac{\sum_{m \in Item^{test}} |U(m)| \cdot \delta(m, Rec(User^{test}))}{\sum_{m \in Item^{test}} |U(m)|}. \quad (12)$$

Table 1. Recommendation results classification

	Recommended	Not recommended
Linked	True-Positive (tp)	False-Positive (fp)
Not linked	False-Positive (fp)	True-Negative (tn)

$\delta(m, Rec(User^{test})) = 1$, only if item m appears in any recommendation lists for a given test data; $U(m)$ is the list of users who accessed item. A list of a fixed number of recommendations, *Rec*, is produced for each active user u_i , and we define recommendable items as items with predicted ratings greater than the average rating of u_i . *Rec* contains items with the highest predicted ratings.

Diversity (*Div*) measures how diverse recommendation lists are. The pairwise diversity for two users are computed by the following formula:

$$Div(u_i, u_j) = \frac{|Rec(u_i) \cap Rec(u_j)|}{|Rec|}. \quad (13)$$

Div(u_i, u_j) for all pairs of users are then averaged to evaluate the *Diversity* of the recommender systems. This measure is of particular interest, if one is interested in the customization of recommendations given to each individual.

Precision (*Prec*) measures the proportion of the successful recommendations among all recommendations. *Precision* indicates the quality of the produced recommendations with an emphasis on recommendation successes, rather than recommendation failures. *Precision* is calculated by the following formula:

$$Prec = \frac{|tp|}{|tp| + |fp|}. \quad (14)$$

tp and *fp* are the numbers of true-positive and false-positive recommendation results, respectively. All possible recommendation results are shown in Table 1.

Recall (*Rec*) measures the proportion of the successful recommendations with the respect to the items that users actually liked. *Recall* indicates the quality of the produced recommendations with an emphasis on recommendation failures, rather than recommendation successes. *Recall* is calculated by the following formula:

$$Rec = \frac{|tp|}{|tp| + |fn|}. \quad (15)$$

Because each active user has a different watch history and access counts for the items in the testing data, it is impossible to

generate the same fixed size recommendation lists for all users. Therefore, *Precision* and *Recall* are measured on all recommendations that can be validated with true ratings in the testing data; both metrics measure the quality of recommendations in different aspects. *Precision* increases with more recommendation successes, while *Recall* increases with less missed successful recommendation opportunities.

4.2 Baseline

We compare the proposed recommender system with three different types of CF recommender systems: similar user-based recommender system (SU), common expert-based recommender system (CE) and similar common expert-based recommender system (SCE). SU computes pairwise similarities for every pair of users based on their previous rating histories; then, a number of similar neighboring users are selected. Finally, CF is used to predict the ratings or produce recommendations for each user. CE chooses a fixed number of experts considering three absolute expertise measures (*Early Adopter*, *Heavy Access*, *Niche-Item Access*) and then uses the chosen experts as the neighbors for all the users. The last baseline is SCE. It first creates a pool of common experts by considering three and chooses neighbors for each active user by similarity. Thus, it is also expected to strike a good balance between recommendation accuracy and customization.

In tuning the recommender systems, the neighborhood size, k , can be chosen using a validation data set; however, previous works using the MovieLens data set [28, 29] reported the same result when using a fixed size k for recommendations. In this work, we set k to be 50 to compare the performance of different neighborhoods.

To predict user preference (i.e., ratings), we use the following CF algorithm:

- 1) Select k users as a neighborhood for the given active user.
- 2) Assign a user weight to the selected users.
- 3) Compute a rating prediction of the active user u_i on an item as weighted average rating of the neighborhood.

In SU, the Pearson correlation (i.e., *Similarity*) is used not only as the similarity measure between users but also as the weights of the selected users ($w(u_i, u_j)$). CE uses the expertise of users to choose a neighborhood in step 1 and the Pearson correlation to determine user weights in step 2. In step 3, the weighted average of the ratings of the selected neighborhood is

Table 2. Prediction accuracy of recommender systems (MAE)

	SU	CE	SCE	PR
MAE	0.8709	0.9466	0.8111	0.7723

computed using the following formula:

$$\hat{R}_{u_i, m} = \bar{R}_{u_i} + \frac{\sum_{u_j \in N(u_i)} w(u_i, u_j) \cdot (R_{u_i, m} - \bar{R}(u_j))}{\sum_{u_j \in N(u_i)} w(u_i, u_j)}. \quad (16)$$

We strictly follow the traditional CF algorithm to compare and focus on the qualities of different types of neighborhoods, and if none of the selected neighborhood has used the item, then the system predicts the mean user rating ($\bar{R}(u_i)$).

4.3 Results

We first compare the prediction accuracy in the MAE of different recommender systems. Table 2 shows comparison results and the proposed approach (PE) yields more accurate results than the baselines. It shows an 11.9% improvement over SU, 18.4% over CE, and 4.8% over SCE.

CE yields the least accurate results. It is interesting that SCE is the second best. Both PE and SCE are basically personalized expert-based approaches. SCE first identifies common experts and simply chooses neighbors from the common experts based on similarity to the active user; however, PE first learns the patterns of neighbor selection of each user by SVM considering absolute and relative expertise. Thus, PE can identify more personalized neighbors who can better serve users with different needs and expectations.

To examine various properties of the proposed recommender system, we evaluate recommendations produced by the system. Table 3 shows *Item Coverage* of recommendation lists produced by different recommender systems. *Item Coverage* measures the proportion of items that a recommender system can recommend, and the measure increases as the size of the recommendation list increases. In this respect, SU with similar movie tastes generates recommendation lists with higher *Item Coverage*, while both PE and CE give recommendations that are more widely acceptable, based on their expert knowledge. PE covers slightly more items than CE (2% increase from 0.3837 to 0.3917 at $|Rec| = 20$), and SCE sits in between SU and PE.

For some applications, it is more important to recommend a variety of items. The seller also needs to sell unknown and unpopular items in stock, in addition to the popular items. Table

Table 3. Item coverage of different recommender systems

Rec	SU	CE	SCE	PR
10	0.8608	0.2973	0.3620	0.2986
20	0.9202	0.3837	0.5133	0.3917
30	0.9409	0.4602	0.6103	0.4803
40	0.9531	0.5100	0.6720	0.5611
50	0.9595	0.5614	0.7216	0.6368

Table 4. Diversity of different recommender systems

Rec	SU	CE	SCE	PR
10	0.9290	0.6405	0.8832	0.6914
20	0.9165	0.6393	0.8496	0.6726
30	0.9017	0.6333	0.8201	0.6663
40	0.8862	0.6317	0.7940	0.6672
50	0.8701	0.6287	0.7699	0.6669

4 shows the *Diversity* of the recommendation lists produced by different recommender systems. *Diversity* decreases as the recommendation list size increases, as more common items are included to recommendation lists to active users. Higher *Diversity* means that the more diverse recommendation lists are given to different active users. Similar to *Item Coverage*, SU yields the most diverse recommendation lists, and then SCE, PE and CE follow. The results indicate that SU provides more diverse recommendations that possibly better serve diverse preferences of users; however, recommendation lists with high *Item Coverage* and *Diversity* are not necessarily accurate, as shown in Table 2. In this work, we define personalized experts as neighbors who can help to generate the more accurate recommendations for an active user; hence, PE puts more importance on accuracy over recommendation list customization. If we want PE to generate more customized and diverse recommendation lists, we can accomplish that by searching for personalized experts who can provide diverse recommendations.

Table 5 shows the *Precision* and *Recall* of recommendations. The high *Precision* and low *Recall* of SU indicates that SU provides only a few recommendations, but with high confidence. However, CE recommends more items with fewer successes, resulting in low *Precision* and high *Recall*. PE and SCE achieve both high *Precision* and high *Recall*, which implies good recommendation quality. PE yields better recommendations than SCE because there is no significant difference in *Precision* and the *Recall* of PE is higher at 0.7357 (2.6% improvement over SCE at 0.7171). Taking the opinions of similar experts with simply

Table 5. Precision and recall of recommendations of recommender systems

	SU	CE	SCE	PR
Precision	0.6533	0.5985	0.6485	0.6433
Recall	0.3412	0.6328	0.7171	0.7357

Table 6. Training data sparsity levels

	All data	−1 month	−2 month
ML-100K	94.9%	95.8%	96.9%

high similarity and high common expertise results in good quality recommendations; however, because users need different levels of expert assistance (high or low measures), customizing the neighborhood in terms of various expertise measures including similarity further improves recommendation quality.

The results indicate that PE generates recommendations that are more accurate in terms of lower MAE and higher *Precision* and *Recall* than other recommender systems. In this work, we define personalized experts as neighbors who can help generate more accurate recommendations for an active user; hence, PE places more importance on accuracy over recommendation list customization and selects neighbors who can give the most accurate recommendations to each active user. If we want PE to generate more customized and diverse recommendation lists, we can facilitate that by searching for personalized experts who can give diverse recommendations, as opposed to the accurate recommendations discussed in this work.

5. Discussion

5.1 The Sparsity Problem

CF performance suffers when there is insufficient information, which is also known as the sparsity problem. A typical SU can generate accurate recommendations, but it is not robust to the *sparsity problem*. In this section, we compare the performance of different recommender systems with varying *sparsity* levels.

Table 6 shows different sparsity levels as we introduce more sparseness into the training data. The original data set is not very sparse ($1 - 100000 / (943 \cdot 1682) = 0.9369$) before splitting into training and testing data. To introduce more sparseness into the training data, we removed the rating information received during the last 1-month or 2-month period.

Table 7 illustrates the sensitivity of different recommender systems in relation to varying sparsity levels. The prediction

Table 7. Prediction accuracy by different sparsity levels (MAE)

Sparsity level (%)	SU	CE	SCE	PE
94.9	0.8803	0.9500	0.8111	0.7762
95.8	1.2829	1.3710	1.3383	1.2000
96.9	1.4502	1.7260	1.6733	1.3803

Table 8. Precision by different sparsity levels

Sparsity level (%)	SU	CE	SCE	PE
94.9	0.6533	0.5985	0.6485	0.6433
95.8	0.6521	0.5291	0.5473	0.6521
96.9	0.6490	0.5682	0.5834	0.6490

Table 9. Recall by different sparsity levels

Sparsity level (%)	SU	CE	SCE	PE
94.9	0.3413	0.6329	0.7171	0.7358
95.8	0.2949	0.1375	0.1143	0.5490
96.9	0.2782	0.2818	0.2554	0.4790

accuracy of the user-based CF algorithm decreases as the training data sparseness increases. Among the four neighborhoods in comparison, PE yields the best prediction accuracy with the lowest MAE at all sparsity levels. The prediction accuracy of CE drops 25.9% from 1.3710–1.7260, the accuracy of SCE drops 25.0% from 1.3383–1.6733, the accuracy of k -NN drops 13.0% from 1.2829–1.4502, and the accuracy of PE drops 15.0% from 1.2000–1.3803 as the sparsity level increases from 95.8%–96.9%. At all sparsity levels, PE yields the most accurate prediction results.

The quality of the recommendation also degrades with increasing data sparseness. The *Precision* and *Recall* values from Tables 8 and 9 indicate that, with sparser data (95.8% and 96.9%), SU yields high *Precision* and low *Recall* recommendations, CE and SCE yield low *Precision* and low *Recall* recommendations, and PE yields high *Precision* and high *Recall* recommendations. At all sparsity levels, PE yields the best quality recommendations.

Neighborhoods are selected with the sparse training data, hence the lack of accurate information results in an inaccurate neighborhood selection; furthermore, it is more likely that none of the selected neighborhoods has watched the item in question. In such a case, a recommendation opportunity is missed as the

Table 10. Recommendation miss rate by different sparsity levels

Sparsity level (%)	SU	CE	SCE	PE
94.9	0.5052	0.0261	0.0439	0.0072
95.8	0.5249	0.6998	0.7856	0.1578
96.9	0.5287	0.4230	0.5333	0.2248

CF algorithm in (16) predicts the active user average rating and the item is not recommended. To provide more accurate recommendations, it may be beneficial to only recommend a few items with confidence; however, many opportunities are missed using this method, as evidenced by the passive recommender system result in low *Recall*. Table 10 shows the recommendation miss rate of different neighborhoods.

The recommendation *miss rate* increases as training data sparsity level increases for all types of neighborhoods. An appropriate neighborhood should be able to provide answers to the request of each active user. Although, personalized experts are selected to maximize the prediction accuracy of the CF algorithm, PE can provide recommendations in most opportunities at a sparsity level of 94.9% (original training data), and even at 96.9% with sparser data. As seen in Table 10, SU provides more accurate predictions and recommendations than CE, while the recommendation miss rates of SU are higher than those of CE in most cases (at the sparsity level of 94.9% and 96.9%): CE recommends items more carelessly than SU, and it yields more recommendation failures than recommendation misses.

5.2 Neighborhood Study

In this subsection, we discuss how different neighborhood characteristics result in performance differences in different recommender systems. As seen in Section 4, different neighborhood-based CF algorithms exhibit different performance characteristics. For instance, SU results in recommendations with higher *Diversity* than other recommender systems. This is because SU recommends items that each active user likes; consequently, the overall recommendations for all users are more diverse. However, CE recommends items that the common experts like; this results in the overall recommendations with lower *Diversity*. We want to customize the neighborhood for each user to obtain the best recommendation result; we argue that neighborhoods for users should be different in terms of the degrees of various expertise measures from Section 3.2.

Figure 1 shows different neighborhood characteristics for three different users (User ID: 123, 456). The neighborhood

size is 50, and the standardized expertise measures for all members within each neighborhood are averaged to define the characteristics of the neighborhood. SU, CE, SIMCE and PE show very different characteristics, but the patterns are similar among different users. The SU neighborhood consists of neighbors with the highest similarity only, and its radial graphs peak toward *Sim* measure. The CE neighborhood consists of neighbors with the highest common expertise measure ($\|\langle EA, NA, HA \rangle\|$), and its radial graphs expand toward EA, NA, and EC, with a peak at EA. Note that CE consists of the same common experts for all users and the absolute measures (EA, NA, HA, EC) are constant, whereas relative measures vary by active user. SIMCE stands in-between SU and CE as its neighborhood consists of neighbors with high similarity and high common expertise. Lastly, the PE neighborhood consists of personalized experts; the neighborhood characteristic of PE is very different from the others and expands toward CA, UA, NA and HA. This confirms that personalized experts are not just similar users or common experts; PE provides more accurate recommendations to users as seen in Section 4.

Having shown that a personalized expert is a better alternative to similar users, we now examine how well personalized each expert group is for each active user. By using the Jaccard Index, we measure group similarity among different personalized expert groups. The Jaccard Index is one if two clusters are identical and it is zero if two clusters have no common elements. Given two groups, N_1 and N_2 , the Jaccard Index is defined as follows:

$$J(N_1, N_2) = \frac{|W_1 \cap W_2|}{|W_1 \cup W_2|}. \quad (17)$$

Table 11 shows the neighborhood similarity averages for different types of neighborhoods. Given three different users (User ID: 15, 123, 456), we measure the Jaccard Index for every pair and average the pairwise values for each neighborhood type. As expected CE has neighborhood similarity of one, as the same common experts are suggested for all users; the neighborhoods for SU and SIMCE tend to be more diverse because they are more likely to select neighbors based on *Sim* and users have diverse preferences. We originally expected personalized expert groups for users to be more diverse than what we see here; however, the personalized expert groups overlap significantly and exhibit very similar neighborhood characteristics (high CA, UA, NA, HA), which are also obvious characteristics of heavy access users who access most of the items. In fact, 42 of the most heavy access users (top 5% in

Figure 1. Expertise measures (standardized) of different neighborhood types: similar users (k -NN), common experts (CE), similar common experts (SIMCE), personalized experts (PE). (a) User ID: 123, (b) User ID: 456.

Table 11. Jaccard index of different recommender systems

	SU	CE	SCE	PE
Jaccard	0.0792	1.0000	0.2200	0.8363

HA) are included in each personalized expert group. From this finding and our analysis of personalized expert groups, we conclude that our personalized expert search correctly identifies the most effective neighborhood for a given data set.

k -NN and other user-based CF algorithms gained much popularity for the simplicity of the algorithms and their performance. As the performance of such algorithms largely depends on the neighborhood selection, it is important to select the

most suitable neighborhood for each active user. In this work, we customize the neighborhood for each active user and call such neighborhoods personalized experts; the proposed personalized expert-based recommender system serves users with more accurate recommendations. Furthermore, the proposed neighborhood-based recommender system is more robust to sparse data.

In the neighborhood study, we show that personalized experts are significantly different from similar users, common experts, or similar common experts, and the novel neighborhood (PE) is customized for each active user. We have shown a way to build a global model to find a personalized neighborhood for each active user, but building such a global model can be impractically costly (see Section 3.1) and limits the scalability of the system. In this regard, we plan to explore unsupervised or reinforcement learning algorithms in the future.

Conflict of Interest

No potential conflict of interest relevant to this article was reported.

Acknowledgements

This research was supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2014M3C4A7030503). Also, this work was supported by the NRF grant funded by the Korea government (MSIP) (No. NRF-2016R1A2B4015820).

References

- [1] B. M. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Recommender systems for large-scale e-commerce: scalable neighborhood formation using clustering," in *Proceedings of the 5th International Conference on Computer and Information Technology*, Dhaka, Bangladesh, pp. 291-324, 2002.
- [2] P. Deivendran, T. Mala, and B. Shanmugasundaram, "Content based recommender systems," *International Journal of Computer Science & Emerging Technologies*, vol. 2, no. 1, pp. 148-152, 2011.
- [3] V. Formoso, F. Cacheda, and V. Carneiro, "Algorithms for efficient collaborative filtering," in *Efficiency Issues in Information Retrieval Workshop*. Heidelberg: Springer, pp. 17-28, 2008.
- [4] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," in *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Berkeley, CA, pp. 230-237, 1999. <https://doi.org/10.1145/312624.312682>
- [5] N. Kim and J. H. Lee, "Performance analysis of group recommendation systems in TV domains," *International Journal of Fuzzy Logic and Intelligent Systems*, vol. 15, no. 1, pp. 45-52, 2015. <https://doi.org/10.5391/IJFIS.2015.15.1.45>
- [6] M. Papagelis, D. Plexousakis, and T. Kutsuras, "Alleviating the sparsity problem of collaborative filtering using trust inferences," in *Trust Management*. Heidelberg: Springer, 2005, pp. 224-239. https://doi.org/10.1007/11429760_16
- [7] Q. Shambour and J. Lu, "An effective recommender system by unifying user and item trust information for B2B applications," *Journal of Computer and System Sciences*, vol. 81 no. 7, pp. 1110-1126, 2015. <https://doi.org/10.1016/j.jcss.2014.12.029>
- [8] S. Senecal and J. Nantel, "The influence of online product recommendations on consumers online choices," *Journal of Retailing*, vol. 80, no. 2, pp. 159-169, 2004. <https://doi.org/10.1016/j.jretai.2004.04.001>
- [9] X. Amatriain, N. Lathia, J. M. Pujol, H. Kwak, and N. Oliver, "The wisdom of the few: a collaborative filtering approach based on expert opinions from the web," in *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, Boston, MA, 2009, pp. 532-539. <https://doi.org/10.1145/1571941.1572033>
- [10] N. Kawamae, "Serendipitous recommendations via innovators," in *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, Geneva, Switzerland, 2010, pp. 218-225. <https://doi.org/10.1145/1835449.1835487>
- [11] A. Kumar and M. Bhatia, "Community expert based recommendation for solving first rater problem," *International Journal of Fuzzy Logic and Intelligent Systems*, vol. 15, no. 1, pp. 53-62, 2015. <https://doi.org/10.5391/IJFIS.2015.15.1.53>

- tional Journal of Computer Applications*, vol. 37, no. 10, pp. 7-13, 2012.
- [12] K. Lee and K. Lee, "Using experts among users for novel movie recommendations," *Journal of Computing Science and Engineering*, vol. 7, no. 1, pp. 21-29, 2013. <https://doi.org/10.5626/JCSE.2013.7.1.21>
- [13] P. Rusmevichientong, S. Zhu, and D. Selinger, "Identifying early buyers from purchase data," in *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Seattle, WA, 2004, pp. 671-677. <https://doi.org/10.1145/1014052.1014138>
- [14] S. I. Song, S. Lee, S. Park, and S. G. Lee, "Determining user expertise for improving recommendation performance," in *Proceedings of the 6th International Conference on Ubiquitous Information Management and Communication*, Kuala Lumpur, Malaysia, 2012. <https://doi.org/10.1145/2184751.2184833>
- [15] S. K. Tyler, S. Zhu, Y. Chi, and Y. Zhang, "Ordering innovators and laggards for product categorization and recommendation," in *Proceedings of the 3rd ACM Conference on Recommender Systems*, New York, NY, 2009, pp. 29-36. <https://doi.org/10.1145/1639714.1639721>
- [16] S. W. Kim, C. W. Chung, and D. Kim, "An opinion-based decision model for recommender systems," *Online Information Review*, vol. 33, no. 3, pp. 584-602, 2009. <https://doi.org/10.1108/14684520910969970>
- [17] L. Cheng, Y. Fan, C. Yu, and Y. Du, "An improved trust-aware recommender system for personalized user recommendation in Tmall," in *Proceedings of the 2nd International Conference on Mechanical, electronic and Information Technology Engineering*, Chongqing, China, 2016, pp. 60-63. <https://doi.org/10.12783/dtetr/ICMITE20162016/4573>
- [18] J. Huang, K. Zhu, and N. Zhong, "A probabilistic inference model for recommender systems," *Applied Intelligence*, vol. 45, no. 3, pp. 686-694, 2016. <https://doi.org/10.1007/s10489-016-0783-1>
- [19] Y. Chung, H. W. Jung, J. Kim, and J. H. Lee, "Personalized expert-based recommender system: training C-SVM for personalized expert identification," in *Machine Learning and Data Mining in Pattern Recognition*. Heidelberg: Springer, 2013, pp. 434-441. https://doi.org/10.1007/978-3-642-39712-7_33
- [20] Y. Chung, S. W. Lee, and J. H. Lee, "Personalized expert-based recommendation," *Journal of Korean Institute of Intelligent Systems*, vol. 23, no. 1, pp. 7-11, 2013. <https://doi.org/10.5391/JKIS.2013.23.1.7>
- [21] B. Allison, D. Guthrie, and L. Guthrie, "Another look at the data sparsity problem," in *Text, Speech and Dialogue*. Heidelberg: Springer, 2006, pp. 327-334. https://doi.org/10.1007/11846406_41
- [22] D. Billsus and M. J. Pazzani, "Learning collaborative information filters," in *Proceedings of the 15th International Conference on Machine Learning*, Madison, WI, pp. 46-54, 1998.
- [23] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th International Conference on World Wide Web*, Hong Kong, China, pp. 285-295, 2001. <https://doi.org/10.1145/371920.372071>
- [24] M. Sun, G. Lebanon, and P. Kidwell, "Estimating probabilities in recommendation systems," *Journal of the Royal Statistical Society Series C (Applied Statistics)*, vol. 61, no. 3, pp. 471-492, 2012. <https://doi.org/10.1111/j.1467-9876.2011.01027.x>
- [25] Z. B. Zabinsky, "Random search algorithms," in *Wiley Encyclopedia of Operations Research and Management Science*. Chichester: John Wiley & Sons, 2009.
- [26] R. Akbani, S. Kwek, and N. Japkowicz, "Applying support vector machines to imbalanced datasets," in *Machine Learning: ECML 2004*. Heidelberg: Springer, pp. 39-50, 2004. https://doi.org/10.1007/978-3-540-30115-8_7
- [27] E. H. Zheng, P. Li, and Z. H. Song, "Cost sensitive support vector machines," *Control and Decision*, vol. 21, no. 4, pp. 473-476, 2006.
- [28] A. Bellogin, P. Castells, and I. Cantador, "Neighbor selection and weighting in user-based collaborative filtering: a performance prediction approach," *ACM Transactions on the Web*, vol. 8, no. 2, article no. 12, 2014. <https://doi.org/10.1145/2579993>

- [29] J. Wilson, S. Chaudhury, and B. Lall, “Improving collaborative filtering based recommenders using topic modelling,” in *Proceedings of the 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, Warsaw, Poland, 2014, pp. 340-346. <https://doi.org/10.1109/WI-IAT.2014.54>

Yeounoh Chung received his B.S. in Electrical and Computer Engineering and his M.S. in Computer Science from Cornell University, Ithaca, USA, in 2008 and 2009, respectively. He is currently pursuing Ph.D. in Computer Science at Brown University, Providence, RI, USA. His current research interests focus on big data management and data mining.

E-mail: yeounoh_chung@brown.edu

Noo-ri Kim received the B.S. in computer engineering from Sungkyunkwan University, Suwon, Korea in 2013. He is currently pursuing his M.S.-Ph.D. in Computer Engineering at Sungkyunkwan University. His research interests include recommender systems, text mining, and machine learning.

chine learning.

E-mail: pd99j@skku.edu

Chang-yong Park received his B.S. in Computer Engineering from Dongguk University, Korea, in 2010, and his M.S. in Computer Engineering from Sungkyunkwan University in 2014. Now he works at LG Electronics as a software engineer. His research interests include software engineering, context-aware recommender system, and intelligent agents.

E-mail: changyong1.park@lge.com

Jee-Hyong Lee received his B.S., M.S., and Ph.D. in computer science from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 1993, 1995, and 1999, respectively. From 2000 to 2002, he was an international fellow at SRI International, USA. He joined Sungkyunkwan University, Suwon, Korea, as a faculty member in 2002. His research interests include recommender systems, intelligent systems, and machine learning.

E-mail: john@skku.edu