Information Fusion 28 (2016) 75–82

Contents lists available at ScienceDirect

Information Fusion

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

Efficient recommendation methods using category experts for a large dataset

Won-Seok Hwang [a], Ho-Jong Lee [a,*], Sang-Wook Kim [a,*], Youngjoon Won [a], Min-soo Lee [b]

[a] Hanyang University, 222 Wangsimni-ro, Seongdong-gu, Seoul 133-791, Republic of Korea
[b] Ewha Womans University, 11-1 Daehyun-Dong, Seodaemoon-Ku, Seoul 120-750, Republic of Korea

ARTICLE INFO

Article history:
Available online 30 July 2015

Keywords:
Recommender system
Collaborative filtering
Expert
Performance evaluation

ABSTRACT

Neighborhood-based methods have been proposed to satisfy both the performance and accuracy in recommendation systems. It is difficult, however, to satisfy them together because there is a tradeoff between them especially in a big data environment. In this paper, we present a novel method, called a CE method, using the notion of *category experts* in order to leverage the tradeoff between performance and accuracy. The CE method selects a few users as experts in each category and uses their ratings rather than ordinary neighbors'. In addition, we suggest CES and CEP methods, variants of the CE method, to achieve higher accuracy. The CES method considers the similarity between the active user and category expert in ratings prediction, and the CEP method utilizes the active user's preference (interest) on each category. Finally, we combine all the approaches to create a CESP method, considering similarity and preference simultaneously. Using real-world datasets from MovieLens and Ciao, we show that our proposal successfully leverages the tradeoff between the performance and accuracy and outperforms existing neighborhood-based recommendation methods in coverage. More specifically, the CESP method provides 5% improved accuracy compared to the item-based method while performing 9 times faster than the user-based method.

1. Introduction

As the number of online items significantly grows, it becomes a difficult task for users to find items on their own. Good matching of users to suitable items is critical to enhance user satisfaction. It highlights the importance of *recommendation systems*, which automatically suggest items with which a user would be satisfied [1,2]. Previous recommendation systems are often based on *neighborhood-based methods* (NBMs) [3], which predict a rating of an item that an *active user* has not evaluated yet and suggest his/her preferable items based on the predicted ratings. NBMs have several advantages of simplicity, justifiability, efficiency, stability, and clear reasoning behind recommendations [4,5]. It finds the neighbors who rate the items in a range close to the active user and then predicts the rating on a target item by using the ratings of those neighbors [6]. Some NBMs search for the neighbors at every recommendation request to exploit the *latest ratings*. This approach enables to find the users whose preferences are more similar to that of an active user because it uses the latest ratings in finding neighbors. However, it suffers from a high execution time for reflecting latest ratings dynamically, which is more serious in a big data environment. On the other hand, another group of NBMs pre-compute similarities or pre-builds the models by ignoring the latest ratings just for reducing the execution time. However, those NBMs would have an accuracy problem because they ignore a portion of ratings (i.e., the latest ratings). Thus, NBMs have a *tradeoff between performance and accuracy*.

In this paper, we propose new recommendation methods based on 'category experts' rather than neighbors of NBMs. In the real world, users have a tendency to trust experts' opinion, so preferences prediction with the experts could be accurate [7,8]. We define *category experts* as the top-*k* users in giving ratings in a certain category. *k*, the number of experts in a category, is determined by the empirical analysis on each category rating. The CE method, one of the proposed methods aggregates the ratings given by the experts to predict ratings that the user will give to unevaluated items. We also extend the CE method by exploiting two metrics to improve the accuracy: 'similarity' between users and category experts and 'category interest,' defining the degree of interest of the users.

The CE method improves the performance significantly because the experts can be easily identified and maintained by counting the number of ratings given by each user. Also, its accuracy is not

* Corresponding authors.
E-mail addresses: hojonglee@agape.hanyang.ac.kr (H.-J. Lee), wook@hanyang.ac.kr (S.-W. Kim).

http://dx.doi.org/10.1016/j.inffus.2015.07.005
1566-2535/

shown worse than the existing methods since people want to look for advice from experts of specific fields in the real world. Thus, using the category experts leverages the tradeoff between performance and accuracy. The CE method and its extensions improve the coverage, a portion of unseen items that are actually predictable, because category experts tend to have rated more items than similar users. For evaluation, we used the datasets from MovieLens and Ciao [9,10]. We analyzed the effect with different numbers of category experts and compared the results with those of various NBMs, such as user-based, item-based, and k-Means clustering based methods, in terms of accuracy, performance, scalability, and coverage.

The organization of our paper is as follows. Section 2 presents the related work. Section 3 proposes the CE method and its extensions. Section 4 deals with the performance and accuracy issues with the real-world datasets. Finally, we conclude the paper in Section 5.

2. Related work

NBMs are common techniques for collaborative filtering (CF) [11,12]. The user-based method (UBM) looks for k users who have a similar preference with the active user and predicts the rating of an active user on a target item from the ratings given by k users. Its similarity can be calculated via Pearson's correlation coefficient, cosine similarity, or the extended generalized vector-space model [13]. The predicted rating $p_{u,i}$ of item i for user u is computed by $p_{u,i} = \bar{r}_u + \frac{\sum_{v \in S_u} (r_{v,i} - \bar{r}_v) \times |sim(u,v)|}{\sum_{v \in S_u} |sim(u,v)|}$ where $r_{v,i}$ is the rating by user v on item i and $\bar{r}_u$ is the average of all the ratings assigned by an active user u. $sim(u, v)$ is the similarity between user u and v, and S_u is the set of k users (neighbors) that are the most similar to active user u. The similarities between the active user and others are calculated online upon a recommendation request. Herlocker et al. [12] analyzes the accuracy of UBM with respect to the selected neighbors having similar tastes. A few others [14,15] select their neighbors from social network relations. To identify a closer neighborhood, the following approaches also have been proposed: Default voting, inverse user frequency, case amplification [16], and weighted-majority prediction [17,18]. UBM can provide accurate recommendations because it considers all ratings including the latest ratings for searching neighbors. The latest ratings are important because most users assign ratings to only a few items [19–21].

The clustering-based methods (CBM) [22,23] improve the performance by determining the neighbors of each user before the recommendation request. Several CBM variations use various clustering models [24–27] to classify the users off-line. They make user clusters before the recommendation request, and then predict the active user's rating of item i based on the ratings given by those users who belong to the group (i.e., cluster) that the active user belongs to. In addition, there are other model-based methods that use the Bayesian model [28] and the latent semantic model [29] other than clustering models. The item-based methods (IBM) were proposed to overcome a performance problem by pre-calculating similarities among all items [30,31]. It calculates the similarity of items rather than that of users because the relationships between items are *relatively static* [31]. It predicts the active user's rating on a target item by referencing his/her ratings on those items similar to the target item. The predicted rating $p_{u,i}$ assigned by active user u on target item i can be calculated as $p_{u,i} = \frac{\sum_{j \in S_i} \{r_{u,j} \times |sim(i,j)|\}}{\sum_{j \in S_i} |sim(i,j)|}$ where $sim(i,j)$ represents the similarity between user i and j, and S_i is the set of items that are similar to item i. However, accuracy of CBM and IBM may decrease as time goes by because they exclude the latest ratings after the neighborhood is determined.

There are several previous methods considering experts to improve the accuracy. Amatrin et al. [7] utilize the ratings given by experts in Rotten Tomatoes in order to recommend to Netflix users. Yun et al. [32] also crawl the expert ratings from Rotten Tomatoes and predicts the ratings for users in the other domain via the SVD-based model [33]. However, these methods showed an additional overhead for crawling the expert ratings from other domains like Rotten Tomatoes. If there are no pre-defined experts, they cannot provide recommendations.

Cho et al. [8] define experts as the users who evaluate a lot of items. It predicts the rating on the target item by considering ratings from both experts and neighbors who have a similar taste to the active user. Thus, it suffers from long execution time of searching for experts and neighbors. Liu et al. [34] introduce 'star' users, which are close to experts. The star users are virtual users who represent interests of whole users, and a training algorithm selects the star users through analyzing their ratings.

Pham et al. [35] propose an expert-based method for interactive recommendation systems. This method defines experts for a given active user and his/her attributes (i.e., genre, actor, and director). To define the experts, it needs to analyze the attributes as well as correlation among users. Pham et al. [36] also propose another expert-based method that corrects the users' ratings based on experts' opinions for more accurate recommendations. In order to find the experts, the method uses a ⟨A,V⟩-SPEAR algorithm that refers to an ontology built based on items' attributes. To our knowledge, the existing expert-based recommendation methods ignore the performance issue. In this paper, we propose methods based on category experts instead of neighbors to balance the tradeoff between performance and accuracy of NBMs.

3. Category expert methods

This section proposes four recommendation methods using category experts. The *category expert* is a user who is considered to understand well the overall items in a specific category. In this paper, the decision of whether or not a user understands well the item is based on whether or not the user has evaluated the item. The reasoning behind this is, in order to evaluate an item, the user needs to know the item well. Therefore, a user who has evaluated many items in a specific category can be considered as *knowledgeable about the category*, thereby defined as a *category expert*. We formally define the category expert as follows:

Definition 1. For category c, we define the category experts E_c such that:

$$|I_u| \leqslant |I_v| \; (\forall u \in U - E_c, \forall v \in E_c)$$

where $|I_u|$ indicates the number of items that user u has evaluated. In order to determine the category experts, we compute the number of items in each category evaluated by each user. It is easy to maintain those numbers *incrementally* because we just need to increment the number whenever a user evaluates an item. For this reason, we can reduce dramatically the performance of NBMs by decreasing the effort for finding neighbors (i.e., category experts).

CE method: In this paper, we denote the recommendation method that uses category experts as a *CE* method. The CE method predicts the rating with which the active user would assign to the target item, based on the ratings given by the category experts. The intuition behind this is that a user trusts the opinions of experts even if he/she has somewhat different preference with the category experts. The predicted rating for user u on item i included in category c is denoted as $p_{u,i,c}$.

$$p_{u,i,c} = \bar{r}_{u,c} + \frac{1}{k}\sum_{v\in E_c}(r_{v,i} - \bar{r}_{v,c}) \tag{1}$$

$\bar{r}_{u,c}$ is an average rating assigned by user u to the items in category c. E_c is an expert group of category c and consists of the top-k users who assigned the most ratings on the category. $r_{v,i}$ is the rating of user v in expert group E_c assigned to item i. Most items belong to a single category and their expected ratings can be calculated by Eq. (1). When dealing with multiple categories, we consider the item ratings by all experts from their belonging categories. In this approach, the rating $p_{u,i}$ on item i for user u can be predicted as in the following.

$$p_{u,i} = \frac{1}{|C_i|}\sum_{c\in C_i} p_{u,i,c} \tag{2}$$

C_i indicates a set of categories where item i is included, and $|C_i|$ is the number of those categories.

While the CE method needs a short running time, its accuracy is lower than other proposed methods that we will introduce later. This is because, regardless of an active user, the CE method provides almost the same predicted rating for the same item. Specifically, it predicts a rating only based on the category experts who are selected independently of active users.

CES method: The CE method ignores that a user tends to agree on opinions of other users who have similar tastes to the user [16]. Based on this observation, we assume that users eagerly accept the opinions of category experts who have *similar tastes*, but not from all other experts. We adopt this assumption and extend the CE method (refer to as a CES method). The CES method calculates the similarity between the category experts and the active user to measure how close their interests are. $s_{u,v}$ represents the similarity between user u and v, by the Pearson's correlation coefficient shown in Eq. (3).

$$s_{u,v} = \frac{\sum_{i\in I}(r_{u,i} - \bar{r}_u)\cdot(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i\in I}(r_{u,i} - \bar{r}_u)^2}\sqrt{\sum_{i\in I}(r_{v,i} - \bar{r}_v)^2}} \tag{3}$$

I is the set of items that to which both user u and category expert v have assigned ratings to. $r_{u,i}$ is the rating assigned by user u on item i, and $\bar{r}_u$ represents the average of all the ratings assigned by user u. The predicted rating $p_{u,i,c}$ of target item i at category c is calculated by Eq. (4). We note it additionally uses the similarity $s_{u,v}$ between user u and expert v in ratings prediction.

$$p_{u,i,c} = \bar{r}_{u,c} + \frac{\sum_{v\in E_c}(r_{v,i} - \bar{r}_{v,c})\cdot s_{u,v}}{\sum_{v\in E_c} s_{u,v}} \tag{4}$$

The rating $p_{u,i,c}$ is only determined by the ratings of category experts at category c, so there are several $p_{u,i,c}$ when item i is included in multiple categories. The CES method aggregates all these predicted ratings $p_{u,i,c}$ to determine the final rating $p_{u,i}$ for user u on item i by applying Eqs. (4) to (2).

The CES method shows better accuracy than the CE method because it gives different weights to the opinions of category experts depending on an active user. On the other hand, it needs more execution time than the CE method due to the extra overhead for similarity computations. Even though it computes the similarity of users, it spends much less execution time than the existing UBM. This is because the CES method has to compute k similarity values for an active user while UBM does n similarity values. We note k and n are the numbers of category experts and all users, respectively.

CEP method: Some items are belonging to multiple categories. Both the CE and CES methods equally consider the opinions of each category's experts. In the real world, a user may have different levels of interest in each category and is more likely to respect the opinions of the users who have expertise on their favorite category. Based on this intuition, we propose the CEP method that is an extended version of the CE method. In the CEP method, we consider the category experts' opinions differently depending on the active user's interest on each category to enhance the accuracy of rating prediction. The CEP method first defines the category interest as the interest level of a user in a category. The category interest of a user on a category is proportional to the number of items that he/she has evaluated in the category. The CEP method predicts rating $p_{u,i}$ on item i for user u as Eq. (5).

$$p_{u,i} = \frac{\sum_{c\in C_i} p_{u,i,c}\cdot f_{u,c}}{\sum_{c\in C_i} f_{u,c}} \tag{5}$$

C_i denotes the categories that item i belongs to, and $f_{u,c}$ is the amount of interest that user u has in category c. In the CEP method, $p_{u,i,c}$, which is calculated by Eq. (1) is applied to Eq. (5), to predict the final rating.

The accuracy of the CEP method is higher than that of the CE method because it produces more personalized results based on the active user's category interest. In terms of the execution time, the CEP method performs worse than the CE method because it additionally utilizes category interests $f_{u,c}$ (compare the Eqs. (2) and (5)). However, we expect that the additional execution time is insignificant because each user's category interests are already computed to select the category experts. For this reason, compared to the CES method, the CEP method performs much faster.

CESP method: The similarities between the active user and the category experts can be applied to the CE method together with the category interest. In a CESP method, the rating of user u on item i is predicted through Eq. (4) in the CES method for $p_{u,i,c}$. The calculated $p_{u,i,c}$ is applied to Eq. (5) to obtain $p_{u,i}$, which is the final predicted rating that is a result of aggregating the predicted ratings $p_{u,i,c}$.

$$p_{u,i} = \frac{1}{\sum_{c\in C_i} f_{u,c}} \times \sum_{c\in C_i}\left(\bar{r}_{u,c} + \frac{\sum_{v\in E_c}(r_{v,i} - \bar{r}_{v,c})\cdot s_{u,v}}{\sum_{v\in E_c} s_{u,v}}\right)\cdot f_{u,c} \tag{6}$$

Table 1 summarizes the characteristics of four proposed methods. We propose the CE method to relieve the performance problem of UBMs. The CES, CEP, and CESP methods require additional operations on top of the CE method; yet, they are expected to be faster than UBMs because the computing overhead of category interest and similarity would be much less than the overhead of calculating similarity among all users. The CESP method is the most accurate than all other proposed methods (i.e., the CE, CES, and CEP methods). More details are discussed in Section 4.

4. Evaluation

We used MovieLens and Ciao datasets [9,10] for our evaluation. MovieLens has 100,000 ratings that range from 1 to 5 points for 1682 movies by 943 users and every user has rated at least 20 movies.[1] There are 19 movie genres (i.e., one category for one genre), and every movie is classified into at least one genre. Ciao has 26,660 ratings that also range from 1 to 5 points for 1848 items under 28 categories by 590 users. Every user has rated at least 5 items because we removed the users who have evaluated less than 5 items by following the assumption in [15] to avoid the new (cold-start) user problem [1,5]. The density of MovieLens and Ciao datasets are 0.06 and 0.02, respectively.

We carried out the experiments with a different number of category experts to measure its effectiveness. The accuracy was measured by dividing each of the MovieLens and Ciao datasets into 5

[1] This data cleaned up users who have less than 20 ratings (http://files.grouplens.org/datasets/movielens/ml-100k-README.txt).

Table 1
The CE method and its extensions.

Method	Similarity	Category interest
CE method	X	X
CES method	O	X
CEP method	X	O
CESP method	O	O

subsets and using 5-fold cross validation, which is a widely accepted in the recommendation systems area [7,32]. To evaluate the accuracy, we used Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). MAE is a measure that calculates the average error, which is the difference between the predicted ratings and the actual ratings, while RMSE is a measure putting more emphasis on larger errors.

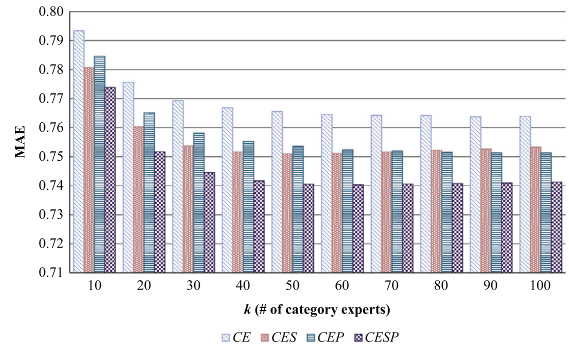
Fig. 1 shows MAE and RMSE according to different numbers of category experts. The CES and CEP methods show higher accuracy than the CE method. This is due to the fact that considering the similarity between the active user and category experts helps to improve the accuracy of recommendations, which coincides with the results of the existing research [16]. The CESP method shows the highest accuracy. As a result, we can confirm that combining the similarity between the active user and category experts and the category preference of the active user helps improving the accuracy as well. When the category expert count is 60, the CES, CEP, and CESP methods show the lowest MAE and RMSE values. The CE method shows the lowest values when k is 80. The lowest MAE (RMSE) values of the CE, CES, CEP, and CESP methods are 0.764, 0.751, 0.752, and 0.740 (0.874, 0.867, 0.867, and 0.860), respectively.

For accuracy, we compared the CE method and its extensions to the following collaborative filtering methods: UBM, IBM, and CBM. For CBM, we employed k-means clustering that produces different results depending on initial seeds. To avoid the problem caused by its randomness, we performed CBM 10 times and selected the most accurate one as the final result. We tried various parameter values, such as the similar user count (k_u) for UBM, the similar item count (k_i) for IBM, and the cluster count (k_c) for CBM. We evaluated their accuracy according to the parameter values. We compared their accuracy to the best one of the CESP method with the category expert count of 60.

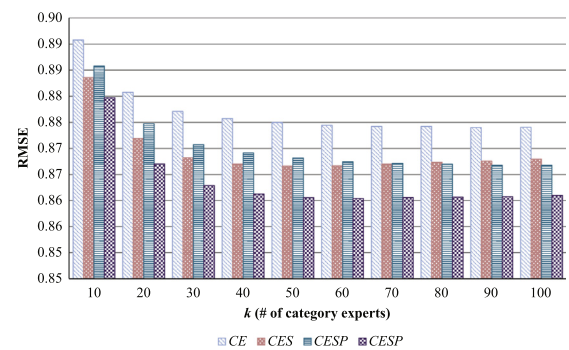
Fig. 2 presents the accuracy with different parameters. The light-colored[2] dotted line represents RMSE of the CESP method, and the dark-colored dotted line represents MAE of the CESP method, which are presented for the comparison purpose. Fig. 2a shows that UBM has the highest accuracy where k_u is 60 (MAE: 0.743, RMSE: 0.862). The MAE and RMSE values of UBM are higher than those of the CESP method. In addition, we observed that the CE, CES, and CEP methods show lower accuracy than UBM in Figs. 1 and 2a. In Fig. 2b, IBM has the best accuracy with the similar user count of 100 (MAE: 0.860, RMSE: 0.928). The MAE and RMSE values are, however, higher than those of the CESP method. In Fig. 2c, CBM with 5 clusters shows the highest accuracy (MAE: 0.780, RMSE: 0.833). As the cluster count increases, the accuracy decreases. CBM method shows low accuracy when compared with the CESP method.

We claim that IBM and CBM cannot reflect the latest ratings given by users since these methods rely on the pre-computed values before recommendation requests. To show this limitation, we made new training sets that contains only 90%, 92.5%, 95%, and 97.5% ratings, and then we applied this set to IBM and CBM to build the similarity and clustering model. After that, we used original

[2] For interpretation of color in Fig. 2, the reader is referred to the web version of this article.



(a) MAE values for the CE, CES, CEP, and CESP methods.



(b) RMSE values for the CE, CES, CEP, and CESP methods.

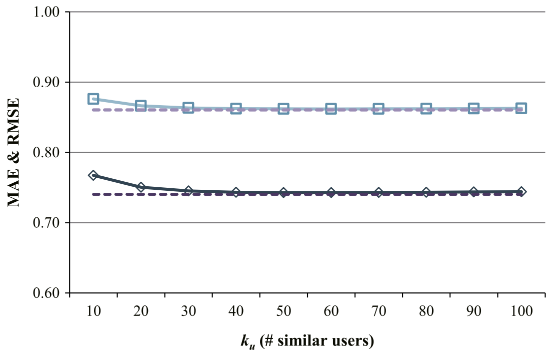
Fig. 1. Accuracy measures for the CE, CSE, CEP, and CESP methods; a. MAE and b. RMSE.

training sets for predicting ratings in IBM and CBM. The parameters of each method were set as $k_i = 100$ (50) and $k_c = 5$ (5) for Movielens (Ciao).

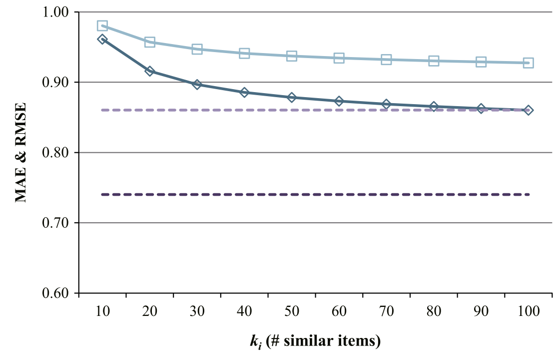
Table 2 shows the MAE and RMSE values of IBM and CBM with new training sets that exclude some ratings. The results show that both IBM and CBM produce higher MAE and RMSE values as they use a less number of ratings for a training set. It means that IBM and CBM are less accurate when they ignore the latest ratings given by users. In IBM with Ciao, the differences of MAE and RMSE values among different training sets are small because Ciao has relatively a small number of ratings considering its numbers of users and items. For this reason, the ratings are insufficient to make accurate results even when no ratings are excluded from the training set. On the other hand, CBM has similar MAE and RMSE values for different training sets in Movielens. This is because Movielens has relatively a more number of ratings, so CBM can build similar clustering models even when training sets exclude some ratings.

We listed the lowest MAE and RMSE values for each method using the Movielens and Ciao datasets in Table 3. In addition, we compared them with the values of excluding 5% ratings from the training set for IBM and CBM. The parameters of each method were set as $k = 60$ (100), $k_u = 60$ (100), $k_i = 100$ (50), and $k_c = 5$ (5) for Movielens (Ciao). In this evaluation, we adopted 5-fold cross validation, so the MAE and RMSE values for different folds (pairs of training and test sets) could be different. In order to see the degree of difference, we show standard deviation in the parenthesis. In the results, the standard deviation values are very small, so the MAE and RMSE values for different folds are similar in each method.

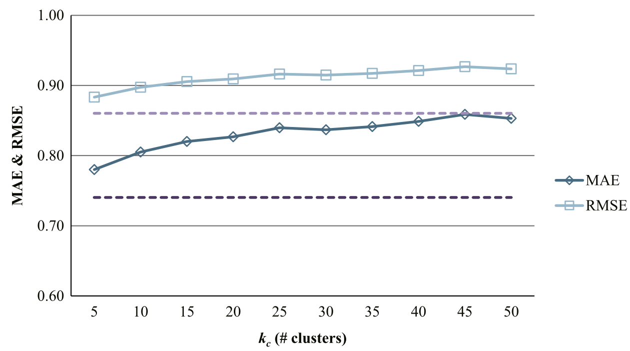
For Movielens, the CESP method shows higher accuracy than IBM and CBM. The MAE value of the CESP method is up to 9% lower than that of IBM and 5% lower than that of CBM. The CESP method even achieves better accuracy than UBM. IBM using all ratings shows better accuracy than that using 95% ratings. In CBM, the



(a) User-based method.



(b) Item-based method.



(c) Cluster-based method.

Fig. 2. MAE and RMSE values for all methods with different parameters.

Table 2
The MAE and RMSE values for IBM and CBM using the training sets excluding some ratings.

Methods	Ratio of ratings (%)	Movielens		Ciao	
		MAE	RMSE	MAE	RMSE
IBM	100	0.810	0.900	0.831	0.911
	97.5	0.846	0.917	0.835	0.913
	95	0.863	0.929	0.838	0.915
	92.5	0.871	0.936	0.839	0.916
	90	0.876	0.940	0.839	0.916
CBM	100	0.784	0.885	0.868	0.932
	97.5	0.784	0.885	0.875	0.938
	95	0.784	0.885	0.884	0.944
	92.5	0.785	0.885	0.890	0.950
	90	0.786	0.887	0.892	0.953

Table 3
Best MAE and RMSE values for all the methods.

Methods	Movielens		Ciao	
	MAE (Std. Dev.)	RMSE (Std. Dev.)	MAE (Std. Dev.)	RMSE (Std. Dev.)
CESP method	**0.740** (0.007)	**0.860** (0.004)	0.836 (0.004)	0.914 (0.002)
UBM	0.743 (0.004)	0.862 (0.003)	0.838 (0.002)	0.915 (0.001)
IBM	0.810 (0.004)	0.900 (0.002)	**0.831** (0.005)	**0.911** (0.003)
IBM (95% ratings)	0.863 (0.004)	0.929 (0.003)	0.838 (0.004)	0.915 (0.002)
CBM	0.784 (0.006)	0.885 (0.003)	0.868 (0.001)	0.932 (0.000)
CBM (95% ratings)	0.780 (0.006)	0.883 (0.003)	0.892 (0.010)	0.944 (0.005)

accuracy does not change when 95% ratings are used, since the clustering model does not change significantly. The MAE and RMSE values of all the methods with Ciao are higher than those with Movielens because Ciao is sparser than Movielens (i.e., the densities of Ciao and Movielens are 0.02 and 0.06, respectively). IBM is more accurate than the CESP method while IBM without 5% ratings is less accurate. The MAE value of the CESP method is up to 0.2% lower than that of IBM and 3.6% lower than that of CBM when 95% ratings are used. The comparison between UBM and the CESP method is similar to that between IBM and the CESP method. CBM provides the lowest accuracy and the lowest coverage among all the methods. The accuracies of UBM and IBM are similar to each other for Ciao because the average number of items evaluated by each user (about 36.2) is not significantly different than that of users who have evaluated each item (about 22.7).

In order to show that the CESP method and the existing methods provide *statistically different* results, we employed the Wilcoxon signed rank test [37] by comparing the ratings predicted by the CESP and existing methods (i.e., UBM, IBM, and CBM). Table 4 shows the results of the Wilcoxon tests for Movielens

Table 4
The results of Wilcoxon ranks tests for comparing the CESP and existing methods.

Data sets	Statistics	Pair of methods compared		
		CESP-UBM	CESP-IBM	CESP-CBM
Movielens	*Z*	−57.189	−91.561	−60.014
	Asymp. Sig. (2-tailed)	.000	.000	.000
Ciao	*Z*	−5.885	−10.719	−5.370
	Asymp. Sig. (2-tailed)	.000	.000	.000

and Ciao datasets. We observe that all significance values are 0; therefore, we reject the null hypothesis that there is no difference in ratings between the CESP and existing methods. In other words, the CESP method predicts ratings statistically different from the ratings predicted by the existing methods. Observing Tables 3 and 4, we conclude that the CESP method outperforms UBM, IBM, and CBM on both Movielens and Ciao datasets, which is statistically significant.

We illustrate the improvements in terms of execution time. We ran the 64 bit Windows Server 2008 with a 12 GB RAM and a 3.50 GHz Intel Core-i7 processor and measured the execution time for predicting 100,000 ratings. The parameters k, k_u, k_i, and k_c from Figs. 3 and 4 denote the numbers of category experts, similar users, similar items, and clusters required in each method.

Fig. 3 shows that the more category experts there are, the longer the execution time is. The left among the two y-axes indicates the execution time of the CE and CEP methods while the right one shows the time of the CES and CESP methods. We can see that the CE and CEP methods spend relatively less execution time since they are not required to calculate the similarities between the active user and category experts. In Fig. 1, there is almost no change in accuracy when the numbers of category experts is more than 40; Fig. 3 shows that the execution time increases continuously when it is more than 40. Thus, setting the number of category experts as 40 is reasonable for both the accuracy and the execution time.

Fig. 4 shows the execution time for UBM, IBM, and CBM using the Movielens dataset. In UBM, the more similar users there are, the longer the execution time is. The same amount of computing similarities is needed for all the active users regardless of the parameter k_u, so the difference between the execution times is not significant. IBM requires more time as the number of similar items increases. In CBM, the execution time decreases as the number of clusters increases. This is because the number of users in each cluster may decrease as the number of clusters increases.

In Table 5, we compare the average execution times of all the methods for predicting only one rating when it has the highest

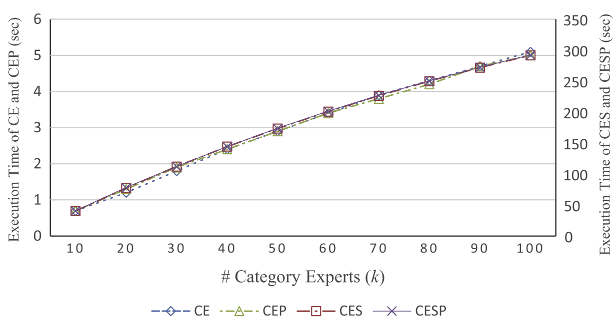
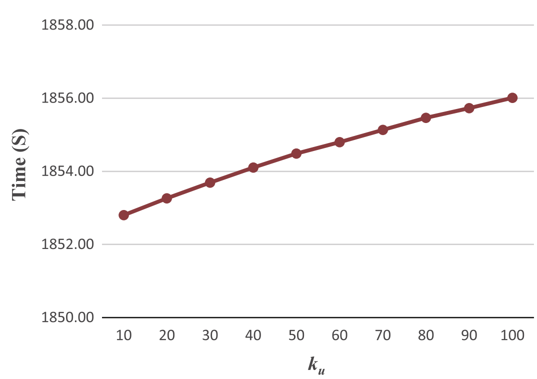


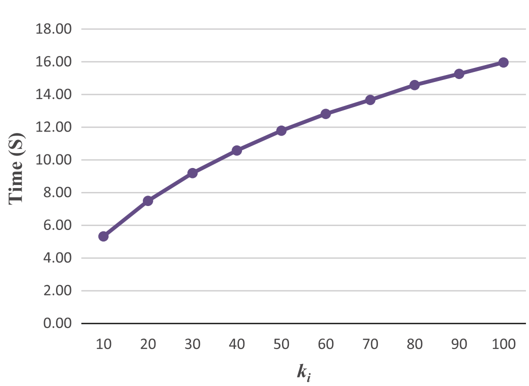
Fig. 3. Execution time for the CE, CES, CEP, and CESP methods.

Table 5
Execution time with parameter settings providing the best accuracy.

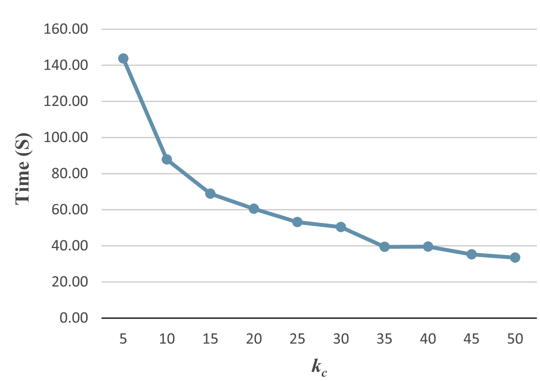
Methods	Execution time (μs)	
	Movielens	Ciao
CE method	92.2	7.6
CES method	2852.8	252.8
CEP method	93.6	8.2
CESP method	3026.3	254.7
UBM	27561.0	5486.5
IBM	240.2	42.5
CBM	2117.7	437.0



(a) User-based method.



(b) Item-based method.



(c) Clustering-based method.

Fig. 4. Execution time for UBM, IBM, and CBM.

accuracy. This set of experiments is meaningful because each method has a different set of ratings that cannot be predicted. The proposed methods need much less execution time than UBM. The CE and CEP methods perform 32 times faster than UBM, and the CES method; the CESP method is 9 times faster. UBM requires the longest time because it calculates the similarities between the active user and all other users. The execution time for IBM is about 2.5 times longer than the CE and CEP methods. This is because each method aggregates a different number of ratings for prediction (i.e., 80 ratings for the CE method, 60 ratings for the CEP method, and 100 ratings for IBM). IBM spends less execution time than the CES and CESP methods because it does not calculate similarities in the rating prediction step. CBM requires more execution time than the CE and CEP methods while it needs less execution time than the CES and CESP methods. The reason is that the category experts tend to evaluate more items than other users, so more rating-comparisons are needed for computing a similarity value. All the methods need less execution time with the Ciao dataset because the dataset is relatively sparser than the Movielens dataset. We checked that the data sparsity considerably affects the time for similarity computations; the computation on Movielens (17.5 μs) needs more time than Ciao (1.7 μs).

Table 6 shows the coverage of the proposed methods with parameter k of their best accuracy using both Movielens and Ciao. The coverage represents how much percentage of unseen items is predictable with each method; the higher, the better [12,38]. On Movielens, the CE method shows the highest coverage (98.19%) among all the proposed methods while the CESP shows the lowest (94.31%). The coverages of the CES and CEP methods are 97.93% and 94.55%, respectively. The coverage of the CEP and CESP methods is lower than that of the CE and CES methods since the CEP and CESP methods cannot compute the category preference for a specific category when the active user has not evaluated any items in that category.

The CESP method outperforms IBM and CBM (up to 17% better than CBM) while it shows lower coverage than UBM (4% worse than UBM) on Movielens. UBM shows higher coverage than IBM because the similarities of users would be more accurate than those of items. On Movielens, each user has more ratings (i.e., about 144.1) than each item has (i.e., about 66.2). CBM has the lowest coverage because most users who evaluated more items are classified into one cluster. We found that the difference between the average numbers of items evaluated by users in each cluster is up to about 40. Every result, excluding 5% ratings, shows lower coverage than those of using all ratings. The coverage using Ciao is worse than that using Movielens because Ciao is sparser than Movielens. In Ciao, the CESP method shows coverage 2–7 times better than the other methods.

In this section, we compared our methods with the existing methods in terms of accuracy, performance, and coverage. We observed that the CESP method, which combines all above approaches, leverages the tradeoff between the accuracy and execution time, and outperforms the existing methods in terms of coverage.

Table 6
Coverage with parameter settings providing the best accuracy.

Methods	Coverage (%)	
	Movielens	Ciao
CESP method	94.31	**27.36**
UBM	**97.86**	11.29
IBM	87.81	7.17
IBM (excluding 5% ratings)	87.28	5.05
CBM	80.27	4.27
CBM (excluding 5% ratings)	80.11	2.44

5. Conclusions

NBMs provide users with the items with which they are likely to be satisfied. Existing NBMs suffer from either performance or accuracy in taking the latest ratings into consideration. In this paper, we introduced a novel approach, called the CE method that uses the notion of category experts in order to leverage the tradeoff between performance and accuracy in previous NBMs. We suggested the CES and CEP methods to improve the accuracy of the CE method, and finally combined all approaches to create the extended CESP method. By using the MovieLens and Ciao datasets, we confirmed that the ‘similarity’ between users and category experts and ‘category interest’ help increase the accuracy of the ratings prediction. In addition, the proposed CESP method showed accuracy close to the NBMs without any pre-computations of the similarity and model, and illustrated higher accuracy than CBM. By measuring the execution time for 100,000 sample items, we revealed that the CESP method performs 9 times faster than UBM and showed the maximum coverage among all existing methods.

Acknowledgements

This work was supported by (1) the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. NRF-2014R1A2A1A10054151), (2) the ICT R&D program of MSIP/IITP (14-824-09-001, Development of High Performance Visual BigData Discovery Platform for Large-Scale Realtime Data Analysis), and (3) MSIP (Ministry of Science, ICT, and Future Planning), under the ITRC (Information Technology Research Center) support program (NIPA-2014-H0301-14-1022) supervised by the NIPA (National IT Industry Promotion Agency).

References

[1] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions, IEEE Trans. Knowl. Data Eng. 17 (6) (2005) 734–749.
[2] H. Chiang, T. Huang, User-adapted travel planning system for personalized schedule recommendation, Inf. Fus. 21 (2015) 3–17.
[3] X. Su, T. Khoshgoftarr, A survey of collaborative filtering techniques, Adv. Artif. Intell. (4) (2009).
[4] R. Bell, Y. Koren, Improved neighborhood-based collaborative filtering, in: KDD Cup and Workshop at the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2007.
[5] B. Shapira, Recommender Systems Handbook, Springer, 2011.
[6] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, J. Riedl, Grouplens: an open architecture for collaborative filtering of netnews, in: Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work, 1994, pp. 175–186.
[7] X. Amatriain, N. Lathia, J.M. Pujol, H. Kwak, 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, 2009, pp. 532–539.
[8] J. Cho, K. Kwon, Y. Park, Collaborative filtering using dual information sources, IEEE Intell. Syst. 22 (3) (2007) 30–38.
[9] B.N. Miller, I. Albert, S.K. Lam, J.A. Konstan, J. Riedl, Movielens unplugged: experiences with an occasionally connected recommender system, in: Proceedings of International Conference on Intelligent User Interfaces, 2003, pp. 263–266.
[10] J. Tang, H. Gao, H. Liu, A. Das Sarma, etrust: Understanding trust evolution in an online world, in: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2012, pp. 253–261.
[11] R.M. Bell, Y. Koren, Lessons from the Netflix prize challenge, ACM SIGKDD Explor. Newslett. 9 (2) (2007) 75–79.
[12] J.L. Herlocker, J.A. Konstan, A. Borchers, J. Riedl, An algorithmic framework for performing collaborative filtering, in: Proceedings of the 22nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 1999, pp. 230–237.
[13] I. Soboroff, C. Nicholas, Combining content and collaboration in text filtering, in: Proceedings of the International Joint Conference on Artificial Intelligence Workshop: Machine Learning for Information Filtering, vol. 99, 1999, pp. 86–91.
[14] J. Golbeck, Computing and Applying Trust in Web-Based Social Networks, Ph.D. thesis, University of Maryland College Park, 2005.
[15] P. Massa, B. Bhattacharjee, Using trust in recommender systems: an experimental analysis, in: Trust Management, 2004, pp. 221–235.

[16] J.S. Breese, D. Heckerman, C. Kadie, Empirical analysis of predictive algorithms for collaborative filtering, in: Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence, 1998, pp. 43–52.
[17] J. Delgado, N. Ishii, Memory-based weighted-majority prediction for recommender systems, in: Proceedings of ACM SIGIR'99 Workshop Recommender Systems: Algorithms and Evaluation, 1999.
[18] A. Nakamura, N. Abe, Collaborative filtering using weighted majority prediction algorithms, in: Proceedings of 15th International Conference on Machine Learning, 1998, pp. 395–403.
[19] Z. Huang, H. Chen, D. Zeng, Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering, ACM Trans. Inf. Syst. 22 (1) (2004) 116–142.
[20] D. Billsus, M. Pazzani, Learing collaborative information filters, in: Proceedings of the 15th International Conference on Machine Learning, 1998, pp. 46–53.
[21] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Application of Dimensionality Reduction in Recommender System – A Case Study, Tech. Rep. TR-00-043, DTIC Document, 2000.
[22] L.H. Ungar, D.P. Foster, Clustering methods for collaborative filtering, in: Proceedings of AAAI Workshop on Recommendation Systems, vol. 1, 1998, pp. 114–129.
[23] B.M. Sarwar, G. Karypis, J. Konstan, 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, 2002, pp. 285–295.
[24] J. MacQueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, 1967, pp. 281–297.
[25] M. Ester, H.P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1996, pp. 226–231.
[26] M. Ankerst, M. Breunig, H. Kriegel, J. Sander, Optics: ordering points to identify the clustering structure, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, 1999, pp. 49–60.
[27] J. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufman, 2001.
[28] K. Miyahara, M. Pazzani, Collaborative filtering with the simple bayesian classifier, in: Proceedings of the 6th Pacific Rim International Conference on Artificial Intelligence, 2000, pp. 679–689.
[29] T. Hofmann, Latent semantic models for collaborative filtering, ACM Trans. Inf. Syst. 22 (1) (2004) 89–115.
[30] G. Karypis, Evaluation of item-based top-N recommendation algorithms, in: Proceedings of the 10th International Conference on Information and Knowledge Management, 2001, pp. 247–254.
[31] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaboration filtering recommendation algorithms, in: Proceedings of the 10th International Conference on World Wide Web, 2001, pp. 285–295.
[32] L. Yun, Y. Yang, J. Wang, G. Zhu, Improving rating estimation in recommender using demographic data and expert opinions, in: Proceedings of IEEE 2nd International Conference on Software Engineering and Service Science, 2001, pp. 120–123.
[33] S. Zhang, W. Wang, J. Ford, F. Makedon, J. Pearlman, Using singular value decomposition approximation for collaborative filtering, in: Proceedings of 7th IEEE International Conference on E-Commerce Technology, 2005, pp. 257–264.
[34] Q. Liu, B. Cheng, C. Xu, Collaborative filtering based on star users, in: Proceedings of the 23rd IEEE International Conference on Tools with Artificial Intelligence, 2011, pp. 223–228.
[35] X.H. Pham, J.J. Jung, N.T. Nguyen, Integrating multiple experts for correction process in interactive recommendation systems, J. Univ. Comp. Sci. 19 (4) (2013) 581–599.
[36] X.H. Pham, T.T. Nguyen, J.J. Jung, N.T. Nguyen, $\langle a, v \rangle$-spear: a new method on expert-based recommendation systems, Cybernet. Syst. 45 (1) (2014) 165–179.
[37] F. Wilcoxon, Individual comparisons by ranking methods, Biomet. Bullet. 1 (6) (1945) 80–83.
[38] S. Middleton, N. Shadbolt, D.D. Roure, Ontological user profiling in recommender systems, ACM Trans. Inf. Syst. 22 (1) (2004) 54–88.