JCSE Journal of Computing Science and Engineering

Regular Paper
Journal of Computing Science and Engineering,
Vol. 7, No. 1, March 2013, pp. 21-29

Using Experts Among Users for Novel Movie Recommendations

Kibeom Lee and Kyogu Lee*
Department of Transdisciplinary Studies, Seoul National University, Seoul, Korea
kiblee@snu.ac.kr, kglee@snu.ac.kr

Abstract
The introduction of recommender systems to existing online services is now practically inevitable, with the increasing number of items and users on online services. Popular recommender systems have successfully implemented satisfactory systems, which are usually based on collaborative filtering. However, collaborative filtering-based recommenders suffer from well-known problems, such as popularity bias, and the cold-start problem. In this paper, we propose an innovative collaborative-filtering based recommender system, which uses the concepts of Experts and Novices to create fine-grained recommendations that focus on being novel, while being kept relevant. Experts and Novices are defined using pre-made clusters of similar items, and the distribution of users' ratings among these clusters. Thus, in order to generate recommendations, the experts are found dynamically depending on the seed items of the novice. The proposed recommender system was built using the MovieLens 1 M dataset, and evaluated with novelty metrics. Results show that the proposed system outperforms matrix factorization methods according to discovery-based novelty metrics, and can be a solution to popularity bias and the cold-start problem, while still retaining collaborative filtering.

Category: Embedded computing

Keywords: Recommender systems; Collaborative filtering; Experts

I. INTRODUCTION

With the bursting growth of the Internet, and the virtually limitless storage space available, there are now hundreds of thousands of items that are available on online stores, whether they be books, movies, or music albums. The shopping experience of a customer has shifted from what was a simple browse-and-search experience among popular items in offline stores, to a search-and-find experience online, with thousands of items. Although this had led to lesser concentrated product sales, with the power of balance shifting from the few best-selling products to niche products that were previously difficult for consumers to discover, a new problem exists: information overload. Paradoxically, now that there are more available items, it has become extremely difficult to discover items that may be of interest to the user.

Recommender systems solve the problem of information overload by filtering unnecessary items, and showing only those that are relevant to the user. Finding items that are relevant to a user or not is determined with user profiles. Among the many existing methods of recommender systems, collaborative filtering is the most effective method, and the most widely used in commercial services. This is because collaborative filtering can be applied to any domain, being independent of the characteristics of the items. It is also relatively easier to implement, compared to content-based and hybrid algorithms,

Open Access http://dx.doi.org/10.5626/JCSE.2013.7.1.21 http://jcse.kiise.org

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.

Received 23 January 2013, **Accepted** 4 February 2013
*Corresponding Author

Copyright © 2013. The Korean Institute of Information Scientists and Engineers pISSN: 1976-4677 eISSN: 2093-8020

and provides the most satisfying results to users.

However, despite providing the most satisfying results, collaborative filtering does have drawbacks. One of the most notable drawbacks is, at the same time, also the reason why it is able to make relevant recommendations to users: popularity bias. Popularity bias, in which the system becomes skewed towards items that are popular amongst the general user population, makes it near impossible for the system to generate truly novel recommendations. A slightly different problem, but still related to popularity bias, is what is dubbed the ‘Harry Potter’ problem. The extreme popularity of Harry Potter causes it to be purchased with items that are totally unrelated. Once these unrelated co-purchases accumulate, the system detects a correlation between Harry Potter and the unrelated item, mistaking Harry Potter for future recommendations to other users who show interest in the unrelated item. Another serious drawback of collaborative filtering is the cold-start problem, in which the recommender system cannot provide recommendations to a new user, or offer new items as recommendations. The cold-start problem is a serious issue for recommendation systems, as there are hundreds of new users joining, and even more items being added to databases everyday.

- In this article, we propose a recommender system based on collaborative filtering and dynamically promoted experts, which can be applied to any domain, focusing on generating novel recommendations, while keeping them relevant. The main contributions of this work can be summarized as follows.
- The proposed system tackles the problem of recommendations from a new viewpoint, using the concept of experts and novices among users, and shows the potential of such systems in its ability to deliver novel, yet relevant, recommendations.
- The proposed system can act as a solution for popularity bias, while still using collaborative filtering, as it provides recommendations based on seed items that are not similar to the items that are mainly consumed by the user.
- The proposed system can also act as a solution to the cold start problem, without the help of content-based recommenders, or any other external information. Because the recommendations are based on experts in certain clusters, users with only a few ratings can be provided with recommendations that are novel and relevant.
- Our proposed recommender system utilizes dynamically created experts, who are knowledgeable, not in the entire item domain, but in a smaller sub-domain. This enables fine-grained recommendations that are novel and relevant at the same time.
- The proposed system generalizes the key idea of Lee and Lee [1], which was limited to the music domain. With the generalization, the key concepts can be applied to any domain, simply using user ratings.

II. RELATED WORK

A. Collaborative Filtering-Based Recommender Systems

One of the first recommender systems was Tapestry, which was based on collaborative filtering [2]. Its goal was to tackle the problem of overflowing emails, by filtering out irrelevant ones. Thus, the remaining emails that were delivered to the user would be the essential, relevant emails. In order to achieve this, Tapestry used the opinions of relatively small groups, such as office workgroups, to recognise irrelevant and relevant emails. However, the limitation of Tapestry was that it worked in small networks, due to the fact that users had to be familiar with the preferences of other users.

Another recommender system based on collaborative filtering was a system called GroupLens, by Resnick et al. [3] and Konstan et al. [4]. GroupLens found news articles that were relevant to users, using the key concept that “people who agreed in the past will probably agree again,” and predicted the ratings that the users would give to the news articles. Since GroupLens’ system found similar users based on their past ratings, the users did not need to be aware of the preferences of other users, unlike Tapestry.

Two early examples of recommender systems that analyzed user profiles, or ratings, to filter similar users and generate recommendations, are Ringo and the Bellcore Video recommender. Ringo specialized in music recommendations that were personalized for each user, by using their preferences [5]. Ringo’s recommender system analyzed the ratings each user gave to music artists, and then found similar users to recommend artists to, who were extremely likely of being rated highly by the user. While Ringo focused its recommendations on music artists, Bellcore’s recommender was centered on movies [6]. Similar to Ringo, Bellcore’s recommender system analyzed user ratings to find users with similar profiles, which were then used to produce recommendations.

Among the various collaborative filtering algorithms used in different domains, Amazon.com’s recommender system is one of the most well-known recommenders. While the preceding recommenders used similar users to find relevant recommendations, Amazon.com analyzed item similarities, rather than user similarities. This approach brought many advantages; the main one being that the method was scalable to very large datasets, while still generating extremely relevant recommendations to the user [7].

B. Expert-Based Recommender Systems

Amatriain et al. [8] used a collaborative filtering-based approach that used expert opinions crawled from the web. The system used a set of expert neighbors from an independent dataset, whose weights were assigned, depending on their similarities with the user. This method uses actual experts of a domain, which can be an expensive approach, if they need to be found manually.

Kumar and Bhatia [9] propose a community expert-based recommendation (CER) system, which uses an algorithm to find shared interest relationships among people through their blog entries. These people are then grouped, so that those in the same group have similar interests. Next, an expert is chosen from each group, whose blog becomes the only source for generating recommendations. Although this CER system, together with the proposed system by Amatriain et al. [8], may be a solution to the cold-start problem, they both need additional information, such as expert reviews and user blogs. Such drawbacks limit the domains that they can be applied to, and also make it harder to apply to existing recommender systems.

Kim et al. [10] present a recommender system that uses expert groups, who are selected from the user pool, to evaluate Web documents that will be provided to the users. The authority and make-up of these experts depend on user feedback from the Web documents provided to the users. The system performs well when there is an active community of experts evaluating web documents, and active users providing feedback of the retrieved documents. This study by Kim et al. [10] is similar, in that the experts are created dynamically from the user pool. It also can be applied to other domains, other than Web documents. However, the system heavily depends on the feedback that users provide, which is used to improve the quality of experts. Another drawback is that the experts need to examine the contents of the items, when deciding whether it is a relevant recommendation or not. Such examinations are time-consuming, and could be a bottleneck in the performance and scalability of the recommender system.

III. CONCEPT

The basic concept of the system proposed in this paper is to use experts, when providing recommendations to 'novices'. A broad definition of an expert would be a person who is knowledgeable in a certain area, while a novice is one who lacks such knowledge. Thus, it is only natural that knowledge is transferred from experts to novices. In this regard, we employ users who are considered experts in their domain, to provide recommendations to novices in those domains.

In this recommender system, each user has a domain, in which he or she is a novice, and an expert. An expert is defined as a user whose item consumption is skewed, or focused, on a certain set of similar items. Likewise, a user is a novice in areas where the consumption rate is low. Taking movies as an example, it is only reasonable that a person who enjoys watching sci-fi movies can provide helpful recommendations to a user who usually watches drama, but occasionally finds some sci-fi movies engaging.

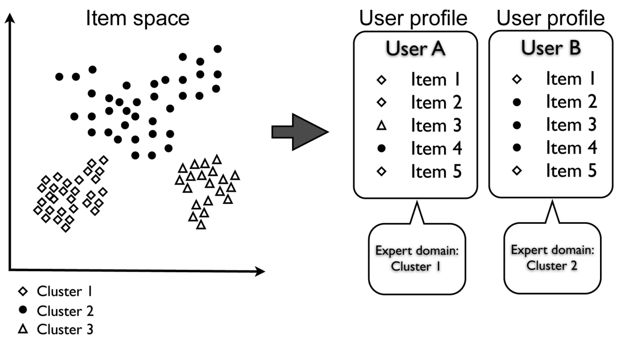

Fig. 1. Concept of the proposed system.

Thus, in order to find experts, the items are placed in an n-dimensional space, so that similar items are placed together, and dissimilar items are apart. Similar items are then clustered, which define the areas that a user can be an expert or novice in. Next, each user is analyzed to see the distribution among the clusters, or areas, that the consumed items are in, and are accordingly labeled as experts for specific clusters, as shown in Fig. 1. When providing recommendations for a novice, the experts of the cluster in which the user is a novice are used to generate novel and relevant recommendations.

IV. DYNAMICALLY PROMOTED EXPERT-BASED RECOMMENDER SYSTEM

The proposed dynamically promoted expert (DPE)-based recommender system is largely comprised of five steps (Fig. 2), in which the most computationally heavy steps can be processed offline, for better performance. The goal of the five steps is to define experts and novices for a sub-set of items in the entire domain. These experts are then used to generate recommendations for a given novice. In order to achieve this, the entire user-item matrix is used to calculate item-item similarities. Next, the items are projected into an n-dimensional space, so that similar items are close to each other, and dissimilar ones are far apart. This projection of items will create clusters of similar items. Using these clusters, we define an expert to be a user whose profile is concentrated on one of the clusters, making him or her an expert in that cluster. A novice is a user whose profile is concentrated on one cluster, but has few, high-rated items in another cluster, making him or her a novice in the latter cluster. We describe this process in detail in the next sections.

A. Item-Item Similarity

One of the most important steps in collaborative filter-

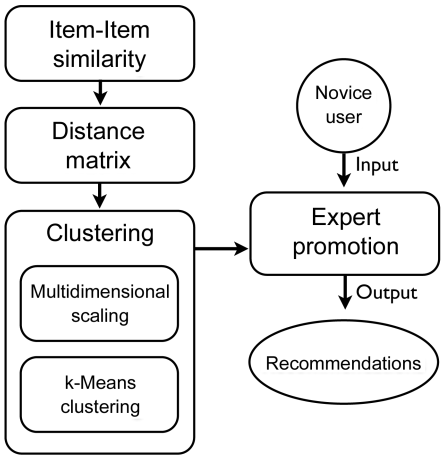

Fig. 2. Overview of the proposed system.

ing is computing the similarity between items, given the user-item matrix. The basic idea of computing similarities between any given two items, is to find users who have rated both items, and applying a similarity computation technique to calculate the similarity between the two items. There are many methods of computing the similarity between items, such as cosine similarity, adjusted cosine similarity, and correlation based similarity.

B. Distance Matrix

The resulting item-item similarity matrix is similarity values, where the diagonal of the matrix is ones, indicating complete equality and other values ranging from 0 to 1. However, the following step (multidimensional scaling, MDS) requires a distance matrix as input. Thus, the similarity matrix from the previous step needs to be converted into a dissimilarity matrix, or distance matrix. In other words, the values in the similarity matrix need to be inverted, where the diagonal becomes zeros, and the remaining values are converted, to show dissimilarity.

C. Item Clustering

In order to cluster the items, the items first have to be placed in an n-dimensional space. This is done by using MDS, and the item-item distance matrix. Next, similar items are clustered together, using the items that are placed in the n-dimensional space.

1) Multidimensional Scaling

MDS is a set of statistical techniques that are often used in information visualization for exploring the similarities or dissimilarities in data. An MDS algorithm, given a distance matrix between items, assigns a location to each item in an n-dimensional space, while preserving their distances according to the distance matrix. Thus, in our proposed system, MDS is used for its ability to calculate locations for each item in a pre-defined dimension, while minimizing the error of their distances.

2) Clustering

Once the items are scattered in a given space, the next step is to cluster items, so that similar, or nearby, items are clustered together. While there exist many clustering algorithms, the k-means clustering algorithm is used in the proposed system, because of its simplicity and effectiveness.

D. Finding Experts for Clusters

Now that every item belongs in a certain cluster, and that these clusters contain similar items, the next step is to assign every user to a cluster, if the users qualify. An assignment to a cluster will indicate that a user is considered an expert for that specific cluster only. For each user, we first analyze the history of rated items, and see where these items lie in the n-dimensional space, which was the output from the MDS stage. If the majority of the distribution of the user's items lies in a certain cluster, then that user is considered an expert for that cluster. At the same time, the user is considered to be a novice in the clusters to which the remaining minority items belong. Deciding whether the majority distribution of the user's items lies in a certain cluster, is done based on the Shannon entropy of the items. If the association of each item to a cluster in the user's history has low entropy, the user is considered to be an expert in the most populated cluster.

E. Generating Recommendations

The direction or flow of recommendations goes from experts to novices. Since the previous step analyzed the entire user database, the system simply looks up which cluster a given user is a novice in. Next, the experts of that particular cluster are isolated, and then analyzed to generate recommendations to the novice. In this proposed system, a simple method of using concurring items amongst the experts for recommendations was used.

V. EXPERIMENT

A. Parameters

The proposed recommender system has many parameters that greatly affect the resulting recommendations. The available parameters are as follows.

1) Item similarity computation method:
Adjusted cosine similarity. Item-item similarity can be computed using cosine-based similarity, Pearson correlation, adjusted cosine-based similarity, etc. Among the many available methods, we use

the adjusted cosine similarity in this paper.

2) Number of dimensions to use for MDS:
Various dimensions. In this experiment, we test various dimensions to see the effects it has over the produced recommendations.

3) Clustering algorithm:
k-means clustering. Clustering can be done using algorithms, such as linkage clustering, k-means clustering, and expectation-maximization (EM) clustering. In this experiment, we use the k-means clustering algorithm, because of its simplicity and effectiveness.

4) Number of clusters to use for clustering:
Various numbers of clusters. In this experiment, we test with various numbers of clusters, to see the effects they have over the produced recommendations.

5) Qualification of an expert:
Entropy. In this experiment, a user was eligible as an expert, if his or her entropy was lower than 3.9442.
Min. number of items in expert area. A user was eligible as an expert if he or she had at least [*Number of clusters*/*5*] items in his or her expert cluster.
Item ratings of expert cluster. The average rating of items in the expert cluster of the user has to be greater than the average rating of all the items of the user.

B. Data

The proposed recommender system was experimented on the MovieLens 1 M dataset, which contains 1,000,209 anonymous ratings of approximately 3,900 movies, made by 6,040 MovieLens users (available at http://www.grouplens.org/node/73).

The user ratings were randomly split into two partitions with a 20% to 80% split ratio. The system was built using 80% of the ratings. Although the aim of the recommender is not to predict items from the 20% split, we had to proceed with the experiment this way because of the evaluation metric that we would be using, which will be discussed in the next section.

VI. EVALUATION OF NOVELTY IN RECOMMENDER SYSTEMS

A. Issues in Evaluating Novelty

There are numerous ways to evaluate recommender systems, most of them focusing on accuracy metrics. While accuracy metrics are able to evaluate a recommender to some extent, there are crucial limitations that arise, because of the differences in how the user perceives the actual recommendations, and how the metric evaluates them [11]. With any existing metric, it is virtually impossible to know which items are actually novel to the user, and the novelty can only be speculated on, indirectly.

The bottom line is that the best method in evaluating recommender systems, especially when the recommender system focuses on novelty, is to carry out live user tests. However, this seemed difficult to carry out using the MovieLens dataset, which is not a 'live' dataset, but an archived one for research purposes, which contains movies from the 1990s. In addition, there is no user pool to begin the experiment, meaning that new users have to be invited in, and user profiles have to be accumulated, before carrying out user studies.

B. Novelty Metric for Recommender Systems

Thus, in order to evaluate our proposed recommender system, we use a novelty metric proposed by Vargas and Castells [12], which uses both rank and relevance. We use two metrics that are introduced in the paper, which focus on novelty: the expected popularity complement (EPC), and the expected profile distance (EPD).

To explain the metrics briefly, EPC is a metric that measures the ability of a system to recommend relevant items that reside in the long-tail

$$\mathrm{EPC} = C\sum_{i_k \in R} disc(k)p(rel|i_k,u)(1-p(seen|i_k)), \qquad (1)$$

where, $disc(k)$ is a discount function, $p(rel|i_k,u)$ is the relevancy of the item in the recommendation list, and $(1-p(seen|i_k))$ reflects a factor of item novelty.

EPD, on the other hand, is a distance-based novelty measure, which looks at distances between the items in the user's profile and the recommended items, as in the following equation

$$\mathrm{EPD} = C'\sum_{i_k \in R}\sum_{j \in u} disc(k)p(rel|i_k,u)p(rel|j,u)d(i_k,j), \qquad (2)$$

where, $C' = C/\sum_{j \in u} p(rel|j,u)$, and $d(i_k,j)$ is the distance between items i_k and j.

VII. RESULTS AND DISCUSSION

The system was evaluated a total of five times, with slight modifications to the parameters on each iteration. The modified algorithms were as follows:

- No modifications (NM): The basic algorithm as explained in Section IV, and using parameters as in Section V, with the exception of omitting the second requirement in promoting experts (i.e., minimum number of consumed items in expert area).
- NM + genre vectors (Mod1): The output of the MDS algorithms are coordinates in k-dimensions for each item. In order to spread the items out further in the

space, we add additional 18 dimensions, in which the values are either 0 or 1. Each dimension represents a genre in the MovieLens dataset, which is given a value of 1 if the item falls under that genre.

- NM + minimum movie consumption (Mod2): This algorithm is the same as NM, but this time includes the second requirement in promoting experts (i.e., minimum number of consumed items in expert area).
- NM + remove popular cluster (Mod3): We found that the distribution of experts is usually biased to one cluster. Here, we do not provide recommendations to novices who are a novice in a cluster that has the most number of experts.
- Hybrid: This modification of the recommender system is a combination of all the above (i.e., NM + Mod1 + Mod2 + Mod3).

First, we compared the EPC and EPD values of our proposed system with the different diversifications of the matrix factorization (MF) baseline recommender, whose results are presented in the study by Vargas and Castells [12]. The recommendation list was cutoff at the top 50 recommendations, and the discount function *disc*(k) was set to 0.85^k. The results of different diversifications of the MF baseline recommender are shown in Table 1, and the results for the proposed system and its variations are shown in Table 2. 'No relevance' indicates that the recommended items are assumed to be relevant to the user by default. 'Relevance' indicates that the metric uses the relevancy values for each item.

We see that the proposed DPE-based recommender system outperforms the MF baseline recommender and its variations, from the perspective of the EPC metric. Although the proposed system does not aim to generate recommendations with items in the long-tail, it seems that it is able to provide recommendations, while avoiding the most popular items, as can be seen with the EPC values for both 'Relevance' and 'No relevance'. Also, we see that the scores of the variations of the DPE-based recommender system do not change significantly, but see that each modification does improve the recommendations, albeit slightly. Surprisingly, the combination of all the variations of the DPE-based recommender system (i.e., Mod4) actually receives a significantly lower score, compared to individual variations. We speculate that the recommender sees a loss in performance, because Mod4 introduces too many constraints that begin to hinder the recommendations. The constraint with the most impact on the loss of performance may be the experts. Such constraints result in fewer experts, and with fewer experts, it is difficult to find concurring items with confidence.

Switching our perspective to the EPD metric, we observe that for 'No relevance', random recommendations score highest in the metric. However, the proposed DPE-based recommender system does not perform as well, receiving values of about 0.56. In the case of 'Relevance', the proposed system performs significantly better than novelty-based greedy diversification algorithm (NGD) and random, but falls short of the performance of MF and its variations. The reason why the proposed system does not perform well according to the EPD metric, is that the EPD metric scores relevancy using the hidden 20% split rating data. In other words, an item in the final recommendation list is relevant, only if it exists in the hidden 20% of the user's ratings. When it does exist, the relevancy score is exponentially proportional to the item's

Table 1. Results on EPC and EPD on different diversification of the MF baseline recommender, with discount function of 0.85^k and recommendation list cutoff at 50 (Results adapted from [12])

	Algorithm	EPC	EPD
No relevance	MF	0.8876	0.7466
	IA-Select	0.8886	0.7577
	MMR	0.8769	0.7428
	NGD	0.9795	0.7551
	Random	0.9527	0.7699
Relevance	MF	0.1043	0.0944
	IA-Select	0.1161	0.1032
	MMR	0.1131	0.1020
	NGD	0.0223	0.0200
	Random	0.0218	0.0179

EPC: expected popularity complement, EPD: expected profile distance, MF: matrix factorization, IA: intent aware, MMR: maximal marginal relevance, NGD: novelty-based greedy diversification.

Table 2. Results on EPC and EPD on different diversification of the proposed recommender, with discount function of 0.85^k and recommendation list cutoff at 50

	Algorithm	EPC	EPD
No relevance	NM	0.9942	0.5634
	Mod1	0.9998	0.5686
	Mod2	0.9928	0.5678
	Mod3	0.9945	0.5634
	Mod4	0.9925	0.5695
Relevance	NM	0.1401	0.0841
	Mod1	0.1271	0.0735
	Mod2	0.1425	0.0849
	Mod3	0.1436	0.0852
	Mod4	0.1310	0.0762

EPC: expected popularity complement, EPD: expected profile distance, NM: no modifications, Mod1: NM+genre vectors, Mod2: NM+minimum movie consumption, Mod3: NM+remove popular cluster, Mod4: the combination of all the variations of the dynamically promoted expert-based recommender system.

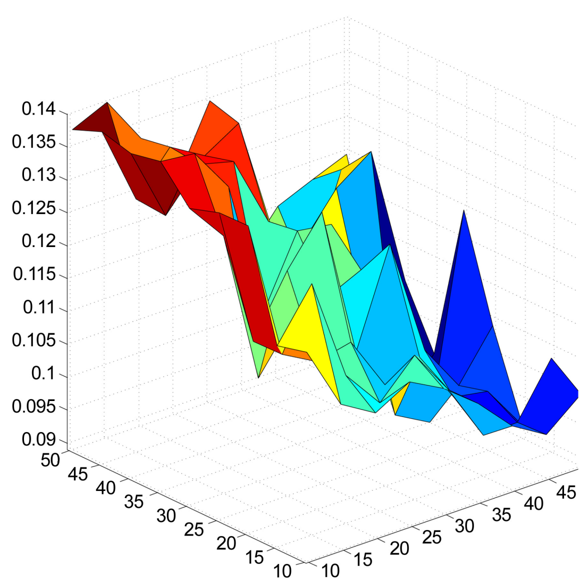

Fig. 3. Changes in expected popularity complement score of the proposed system with varying combinations in the number of dimensions and clusters. The x-axis is the number of clusters/5 and the y-axis is the number of dimensions/5.

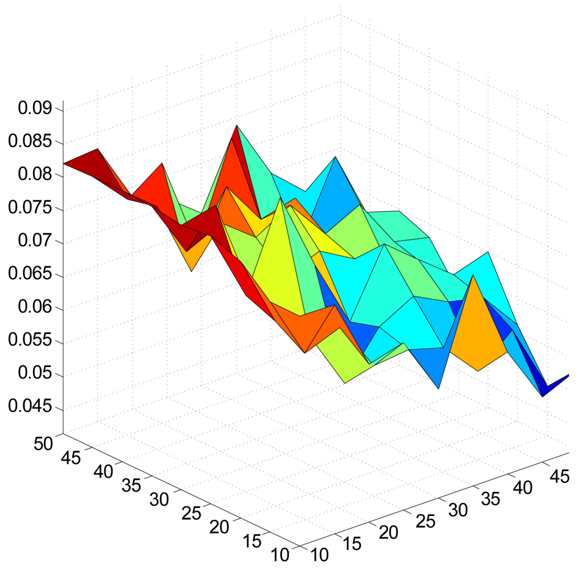

Fig. 4. Changes in expected profile distance score of the proposed system with varying combinations in the number of dimensions and clusters. The x-axis is the number of clusters/5 and the y-axis is the number of dimensions/5.

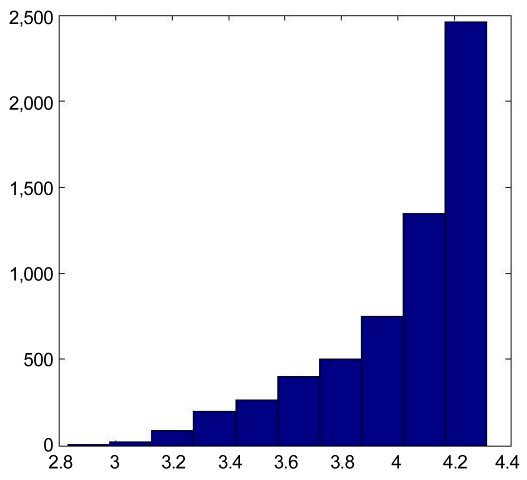

Fig. 5. The entropy distribution of promoted experts. The x-axis shows the entropy values and the y-axis is the number of experts who have such entropy values.

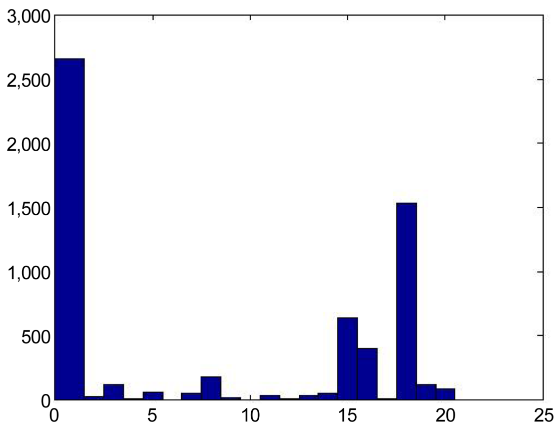

Fig. 6. An example of the distribution of promoted experts for 30 dimensions and 20 clusters. The x-axis is the cluster number and the y-axis is the number of users who are considered experts in the cluster.

rating from the user. The recommendations of the proposed system are generated by analyzing the user's profile, and finding his or her novice area. Earlier, we had defined a novice area as a cluster of items that the user does not frequently interact with. Hence, while we are generating recommendations from this cluster, it will be extremely difficult to find items in the hidden 20% of the ratings.

The recommender system was also evaluated, using different combinations of numbers of dimensions and clusters. The numbers of dimensions and clusters ranged from 10 to 50, in increments of 5. Figs. 3 and 4 show that the system scores better for both EPC and EPD metrics, as the number of dimensions increases. On the other hand, increasing the number of clusters results in a loss of performance. We also observe that the increases in EPC and EPD values, when increasing the number of dimensions and keeping the number of clusters low, are not drastic, but still show noticeable improvements.

Fig. 5 shows the distribution of entropy among the users. More than half the population of users have entropy values larger than 4. By selecting only users that have entropy values less than 3.94, we are able to select users who show patterns of consuming similar items (i.e., from the same cluster) frequently, while remaining significantly less active in other items and clusters.

Fig. 6 shows an example of the distribution of promoted experts among the available clusters in the item space. One cluster is heavily populated with experts, while the remaining clusters have significantly fewer numbers. This can be observed independently of the number of dimensions and clusters. A similar observation is made in research by Connor and Herlocker [13], where they clustered movies together, and found that the movie distributions were focused on a few clusters. Although it may be unnatural for experts to have a uniform distribution among the clusters, it certainly would be better than having experts focused on a single cluster. One solution may be in finding a clustering algorithm that can create clusters of fairly equal sizes.

VIII. CONCLUSION AND FUTURE WORK

In this paper, we proposed a collaborative filtering-based approach that used dynamically promoted experts among the user pool to generate recommendations. These recommendations were more focused on novelty, while still maintaining their relevance. This was done by creating clusters of similar items, which would act as areas of expertise that the users could be experts on. The proposed system was implemented using movie rating data from the MovieLens 1 M dataset, and evaluated using a novelty metric that took into consideration rank and relevance. We used two different approaches in measuring novelty, EPC and EPD metrics, in which the system performed well with the EPC metric, but did not perform up to par with the EPD metric.

The proposed DPE-based system has many parameters that can greatly influence the outcome of the generated recommendations. In this paper, we examined the changes in performance according to the novelty metrics, for different combinations of dimensions and clusters. Future work will examine the effects of different clustering algorithms, changing the parameters regarding the qualification of experts, etc. A deeper study could include the optimal parameter settings and optimal algorithm selections for specific domains, as the characteristics or consumption patterns of the items may be different for various domains.

However, the most important future work should be a live user test, in order to obtain direct user feedback, as this particular recommender system focuses on providing novel recommendations, while also keeping them relevant. The EPC and EPD metrics were not the optimal metrics to evaluate this system, as the goal of the recommender system and the assessment criteria of these metrics were not in sync. While a live user test may not be easy using the MovieLens dataset, as the data is very old, and users are ‘offline’, it will be possible to do a live user evaluation using other datasets.

ACKNOWLEDGMENTS

This research was supported by the Basic Science Research Program, through the National Research Foundation of Korea (NRF), funded by the Ministry of Education, Science and Technology (2011-0013476).

REFERENCES

1. K. Lee and K. Lee, “My head is your tail: applying link analysis on long-tailed music listening behavior for music recommendation,” in *Proceedings of the 5th ACM Conference on Recommender Systems*, Chicago, IL, 2011, pp. 213-220.
2. D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, “Using collaborative filtering to weave an information tapestry,” *Communications of the ACM*, vol. 35, no. 12, pp. 61-70, 1992.
3. P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, “GroupLens: an open architecture for collaborative filtering of netnews,” in *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, Chapel Hill, NC, 1994, pp. 175-186.
4. J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl, “GroupLens: applying collaborative filtering to Usenet news,” *Communications of the ACM*, vol. 40, no. 3, pp. 77-87, 1997.
5. U. Shardanand and P. Maes, “Social information filtering: algorithms for automating “word of mouth”,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Denver, CO, 1995, pp. 210-217.
6. W. Hill, L. Stead, M. Rosenstein, and G. Furnas, “Recommending and evaluating choices in a virtual community of use,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Denver, CO, 1995, pp. 194-201.
7. G. Linden, B. Smith, and J. York, “Amazon.com recommendations: item-to-item collaborative filtering,” *IEEE Internet Computing*, vol. 7, no. 1, pp. 76-80, 2003.
8. 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.
9. A. Kumar and M. P. S. Bhatia, “Community expert based recommendation for solving first rater problem,” *International Journal of Computer Applications*, vol. 37, no. 10, pp. 7-13, 2012.
10. 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.
11. S. M. McNee, J. Riedl, and J. Konstan, “Accurate is not always good: how accuracy metrics have hurt recommender systems,” in *Extended Abstracts of the 2006 ACM Conference on Human Factors in Computing Systems*, Montreal, Canada, 2006, pp. 1097-1101.

12. S. Vargas and P. Castells, "Rank and relevance in novelty and diversity metrics for recommender systems," in *Proceedings of the 5th ACM Conference on Recommender Systems*, Chicago, IL, 2011, pp. 109-116.

13. M. O'Connor and J. Herlocker, "Clustering items for collaborative filtering," presented at *ACM SIGIR Workshop on Recommender Systems: Algorithms and Evaluation*, Berkeley, CA, Aug 19, 1999.

Kibeom Lee

Kibeom Lee is a Ph.D. candidate at the Music and Audio Research Group (MARG) in the Department of Transdisciplinary Studies at Seoul National University. He received his B.S. degree in Computer Science and his M.S. degree in Culture Technology from the Korea Advanced Institute of Science and Technology (KAIST) in 2008 and 2010, respectively. His main research interest is in recommender systems and its applications to various contents.

Kyogu Lee

Kyogu Lee received the B.S. degree in Electrical Engineering from Seoul National University, Seoul, Korea, in 1996, the M.M. degree in Music Technology from New York University, New York, in 2002, and the M.S. degree in Electrical Engineering and the Ph.D. degree in Computer-based Music Theory and Acoustics from Stanford University, Stanford, CA, in 2007 and 2008, respectively. He worked as a Senior Researcher in the Media Technology Lab at Gracenote from 2007 to 2009. He is now an assistant professor in the Department of Transdisciplinary Studies at Seoul National University, Seoul, Korea and is leading the Music and Audio Research Group (MARG). His research focuses on signal processing and machine learning applied to music/audio.