IJREAT International Journal of Research in Engineering & Advanced Technology, Volume 1, Issue 1, March, 2013
ISSN: 2320 - 8791
www.ijreat.org

A Novel Collaborative - Filtering - Based Recommender System Using Iexpand

[1]S. VinaySelvam and [2] N. Saravanan, M.E.,

[1]S. VinaySelvam
Information Technology
Veltech Multi Tech Dr.RR Dr.SR
Engineering College,
Chennai

[2]N. Saravanan, M.E.,
Assistant Professor,
Veltech Multi Tech Dr.RR Dr.SREngineering College,
Chennai

***Abstract*—Recommender systems or recommendation system is a subclass of information filtering system that seek to predict the rating or preference that a user would give to an item. However collaborative–filtering–based recommender system recommends its users on user–item based recommendations by highly under exploring users' interests. In this paper we propose a new collaborative-filtering–based recommender system by expanding users' interests via personalized ranking, named. iExpand. This iExpand based collaborative-filtering approach could its users based on a three layered scheme that focuses on user–interest–item and deals with the issues that exist in traditional collaborative filtering approaches, namely the overspecialization problem and the cold start problem.**

I. INTRODUCTION

The system automatically recommend the few optimal items, which users might like or have interests to buy by learning the user profiles, users' previous transactions, the content of items, etc. [2]. In the recent 20 years, many different types of recommender systems, such as collaborative-filtering-based methods, content-based approaches [12], and hybrid approaches, have been developed.

A. Collaborative Filtering

Since collaborative-filtering methods only require the infor-mation about user interactions and do not rely on the content information of items or user profiles, they have more broad ap-plications [14], [16], [20], and more and more research studies on collaborative filtering have been reported [15]. These methods filter or evaluate items through the opinions of other users. They are usually based on the assumption that the given user will prefer the items which other users with similar preferences liked in the past [2].

In the literature, there are model-based and memory-based methods for collaborative filtering. Model-based approaches learn a model to make recommendation. Algorithms of this category include the matrix factorization, the graph-based approaches [14], etc. The common procedure of memory-based approaches is first to select a set of neighbor users for a given user based on the entire collection of previously rated items by the users. Then, the recommendations are made based on the items that neighbor users like. Indeed, these methods are referred to as user-oriented memory-based approaches.However, existing collaborative-filtering methods often directly exploit the information about the users' interaction with the systems. In other words, they make recommendations by learning a "*user–item*" dualistic relationship. Therefore, existing methods often neglect an important fact that there are many latent user interests which influence user behaviors. To that end, in this paper, we propose a three-layer, user–interests–item, representation scheme. Specifically, we interpret an interest as a requirement from the user to items, while for the corresponding item, the interest can be considered as one of its characteristics.

B. Motivating Example

User latent interests, we will have a better understanding about the users' requirements, since user interests can better connect users and items. Also, when leveraging the information of user interests for developing recommender systems, we must be aware that user interests can change from time to time under the influence of many internal and external factors. For instance, after watching the movie *CrouchingTiger, Hidden Dragon*, use) shows an example of a movie recommender system. In the figure, user *a* is interested in *kung fu* movies, while user *b* likes *Oscar* movies. While both of them have watched the movie *Crouching Tiger, Hidden Dragon*, which was recommended by the system, they have different reasons for watching this movie. Thus, if we can ide r interests may be affected by it.For user *a*, while he is a fan of *kung fu* movies, he may startwatching other movies directed by *Ang Lee*. Also, user *b* may become a fan of *kung fu* movies after her first-time exposure to this *kung fu* movie. If recommender systems cannot capture these changes and only make recommendations according to the user's past interests rather than exploring his/her new preferences, then they are prone to the "*overspecialization*" problem [2].In addition, in real scenarios, the training data are far less than plentiful and most of the items/users only have a few rating/buying records. At this time, typical measures fail to capture actual similarities between items/users and the system is unable to make meaningful recommendations. This situation is summarized as the cold-start problem.

C. Contributions

To address the aforementioned challenges, in our preliminary work, we proposed an item-oriented model-based collaborative-filtering method named iExpand. In iExpand, we assume that each user's rating behavior depends on an underlying set of hidden interests and we use a three-layer, user–interests–item, representation scheme to generate recommendations. Specifically, each user interest is first captured by a latent factor which corresponds to a "*topic*" in topic models. Then, we learn the transition probabilities between different latent interests. Moreover, to deal with the cold-start and "*overspecialization*" problems, we model the possible expansion process of user interests by personalized ranking. In other words, we exploit a personalized ranking strategy on a latent interest correlation graph to predict the next possible interest for each user. At last, iExpand generates the recommendation list by ranking the candidate items according to the expanded user interests. We should note that, compared with previous topic-model-based collaborative-filtering approaches, discovering the correlation between latent interests and using personalized ranking to expand user current interests are the main advantages of iExpand.

In this paper, we further explain why topic models can be used to simulate the user latent interests and we demonstrate the way of extracting these interests from the latent Dirichlet allocation (LDA) model by the Gibbs sampling method. In addition, we illustrate how to use iExpand for making online recommendations in the real-world applications. Finally, we provide systematic experiments on three data sets selected from a wide and diverse range of domains, and we use multiple evaluation metrics to evaluate the performance of iExpand. Since iExpand views collaborative filtering as a ranking problem and aims to make recommendations by directly ranking the candidate items, we report the ranking prediction accuracy. As shown in the experimental results, iExpand outperforms four benchmark methods: two graph-based algorithms and two algorithms based on dimension reduction. As many other algorithms formulate collaborative filtering as a regression problem (i.e., rating prediction), we also report the comparison results of the rating predictions. In addition to this, these new experiments provide more insights into the iExpand model, such as the effect of the parameters and the low computational cost.

D. Outline

The rest of this paper is organized as follows. Section II gives the detail of the iExpand method for effective recommendation. In Section III, we show the experimental results and many discussions. In Section IV, we introduce the related work. Finally, Section V concludes this paper.

II. USER INTEREST EXPANSION

In this section, we first introduce the framework of the iExpand model. Then, we describe each step of the model in detail. In addition, we show how to select parameters. Finally, we address the computational complexity issue.

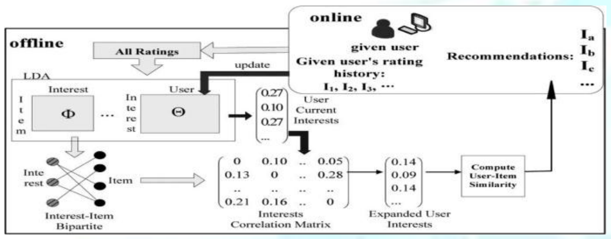


Fig. 1.Framework of the iExpand model. Gray arrows show the general process of the model, while black arrows show the procedure of online recommendations.

A. The Framework of the iExpand Model

First of all, the iExpand model assumes that, in recommender systems, a user's rating behavior depends on an under-lying set of hidden interests. Inspired by the topic models, in iExpand, each user is represented as a probability distribution over interests and each interest is a probability distribution over items. Figure shows the three-layer representation, *user–interests–item*. What is more is that the iExpand modelassumes that the order of items in a user's rating record can be neglected and the users' order in a user set can also be neglected, which means both items and users are exchangeable. In correspondence with the LDA model [8], a topic model that we use in iExpand for extracting user interests, the users are documents, the items are words, and the latent interests are topics, respectively.

Topic models are a type of statistical models, which were firstly proposed in machine learning and natural language processing for discovering the hidden topics (e.g., *Basketball*, *Travel*, and *Cooking*) that occur in a collection of documents.In terms of collaborative filtering, the documents can be viewed as the users, the words are items, and topics become the hidden interests. Based on the hypothesis of topic models, the co-occurrence structure of items in the rating records can be used to recover the latent interest structure and the items that often appear together in one rating record may tend to have the characteristics. In this way, the latent topics can be used to simulate the real-world interests.

Notation	Description
M	number of users
N	number of items
K	number of latent interests
$U = \{U_1, U_2, ..., U_M\}$	the set of users
$I = \{I_1, I_2, ..., I_N\}$	the set of items
$T = \{T_1, T_2, ..., T_K\}$	the set of latent interests
ϕ	a matrix, with ϕ_{ij} equals to $P(I_i\|T_j)$
θ	a matrix, with θ_{ij} equals to $P(T_j\|U_i)$
$\vec{\vartheta}$	a vector, with $\vec{\vartheta}_i$ equals to $P(T_i)$
φ	a matrix, with φ_{ij} equals to $P(T_j\|I_i)$
ψ	a matrix, with ψ_{ij} equals to $P(T_j\|T_i)$
$\theta^{(s)}$	a matrix, and a row is represented as $\vec{\theta}_i^{(s)}$. $\theta_{ij}^{(s)}$ is the probability that a random walk starts from U_i and stops at T_j after s steps

TABLE I
MATHEMATICAL NOTATIONS

Fig. 1 shows the framework of the iExpand model. From Fig. 1 we can see that, when a user comes, the learning and recommendation process of the iExpand model generally consists of four steps. In the first step, the information about user latent interests is extracted by the inference of the LDA model. In the second step, the correlation graph/matrix of latent interests is established by an item–interest bipartite graph projection. In the third step, for a given user, his/her interest distribution is expanded by letting the current interest vector perform a random walk on the interest correlation graph/matrix. Finally, the candidate items are ranked using expanded user interests and the recommendation list is generated.Each step of the iExpand model is introduced in the following sections. For better illustration, Table I lists all mathematical notations used in this paper.

B. Extracting User Interests From the LDA Model

In this section, we show how to extract the information about user latent interests from the LDA model. The information about latent interests include the probability distribution of each user over interests, the probability distribution of each interest over items, and the distribution of each interest.

For collaborative filtering, the LDA model can be represented by a probabilistic graphical model, as shown in Fig. 3(b), where shaded and unshaded variables indicate observed and latent (i.e., unobserved) variables, respectively. In Fig. 3(b), each user in M users is represented as a bag of item tokens M_u, and each token is viewed as an observed variable$i$.Because LDA can provide an intuitive description of each observed variable i, it is a type of generative probabilistic model. Specifically, this item token is generated from a multinomial distribution over items φ_t, specific to an interest t, and interest t is chosen from a multinomial distribution over interestsθ_u, specific to this user. Both θ and φ are modeled by the Dirichlet distribution, with the hyperparametersα and β, respectively.

The Gibbs sampling algorithm begins with the assignment of each item token in users' rating records to a random interest, determining the initial state of the Markov chain. In each of the following iterations of the chain, for each item token, the Gibbs sampling method estimates the conditional distribution of assigning this token to each interest, conditioned on the interest assignments to all other item tokens. An interest is sampled from this conditional distribution and then stored as the new interest assignment for this token. After an enough number of iterations for the Markov chain, the interest assignment for each item token will converge and each token in the rating records is assigned to a "stable" interest. According to the assignment, the distribution of interest T_j over item.

G. Computational Complexity

In this section, we analyze the computational complexity issues for iExpand. Specifically, the time cost for the inference of LDA is $O(M\cdot N\cdot K\cdot l)$, where l is the iteration number of Gibbs sampling. For the bipartite graph projection, most of the time is used to construct the correlation matrix ψ and the time cost in this phase is $O(N\cdot K^2)$. For each user, the cost for random walk is $O(s\cdot K^2)$ on average. Thus, for all the users, it costs $O(s\cdot M\cdot K^2)$. Since KM and KN and the time cost for ranking the items and making recommendations can be neglected, the total computational complexity for the general iExpand process is $O(M\cdot N\cdot K\cdot l)$. As we discussed in Section II-D, in real-world applications, both the inference process and the correlation graph can be updated periodically offline; thus, for online computing, we just need to run less than 30 iterations of Gibbs sampling and one personalized ranking or rating prediction for the current user, both of which can be done efficiently. The online recommendations can be followed by the black arrows shown in Fig. 2.

III. EXPERIMENTAL RESULTS

In this section, we present the experimental results to eval-uate the performance of iExpand. Specifically, we demonstrate the following: 1) the results of parameter selection based on Algorithm 1; 2) a performance comparison between iExpand and many other benchmark methods; 3) an analysis of the parameters in personalized ranking; 4) the understanding of interests and interest expansion; and 5) the discussion about the advantages and limitations of the iExpand model.

A. Experimental Setup

All the experiments were performed on three real-world detailed information about these three data sets are described in Table II.

Data Set	Domain	#Users	#Items	#Records	Sparsity(%)
MovieLens	Movie	943	1,682	100,000	93.70
Book-Crossing	Book	996	1,696	91,084	94.61
Jester	Joke	2,000	100	36,596	81.70

TABLE II
DESCRIPTION OF THREE DATA SETS

For each user's rating record, we split it into a training set and a test set, by randomly selecting some percentage of the ratings to be part of the training set and the remaining ones to be part of the test set. To observe how each algorithm behaves at different sparsity levels, we construct different sizes of training sets from 10% to 90% of the ratings with the increasing step at 10%. In total, we construct nine pairs of training and test sets, and each split named as x−(100−x) means x percent ratings are selected to be the training data and the remaining (100−x) percent ratings for testing.

B. Evaluation Metrics

For the purpose of evaluation, we adopted *Degree* *of Agree-ment*(DOA) [14], *Top-K* [26], and *Recall* [20], [39] as the eval-uation metrics for ranking prediction accuracy. All of them are commonly used for ranking accuracy, and these three metrics try to characterize the recommendation results from different perspectives.

DOA measures the percentage of item pairs ranked in thecorrect order with respect to all pairs [14], [18]. Let $NW_{Ui}=I-(L_{Ui}\cup E_{Ui})$denote the set of items that do not occur in.

C. Parameters in LDA

In this section, we investigate the learning performances of two parameters, namely, hyperparameters and the number of interests, by Algorithm 1. Here, the first 893 users in Movie-Lens are used as training data and the remaining 50 users form the test set. Similarly, for Book-Crossing, the first 900 users are treated as training samples and the remaining users as test data. Also, for Jester data set, the first 1800 users are treated as training data and the remaining 200 users for testing. For each run of Algorithm 1, we initialize the parameters as $a= 0.5$ and $b = 0.5$and turn on Minka's updates after 15 iterations, andthese settings are similar to the ones in [5].

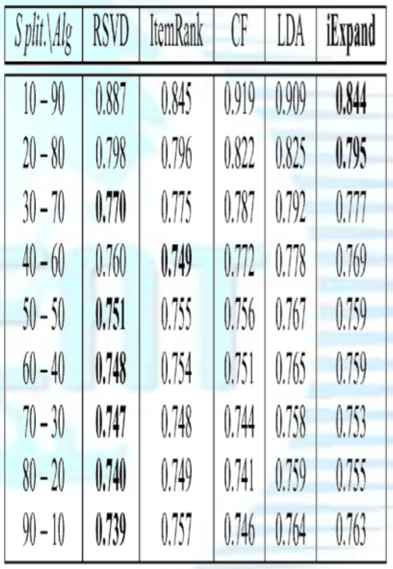
D. Performance Comparison

In this section, we present a performance comparison of both benchmark approaches: ItemRank [18], L^{+} [14], UCF, SVD, LDA, and RSVD [15]. For the purpose of comparison, we record the best performance of each algorithm by tuning their parameters. The training models of all these algorithms are learned

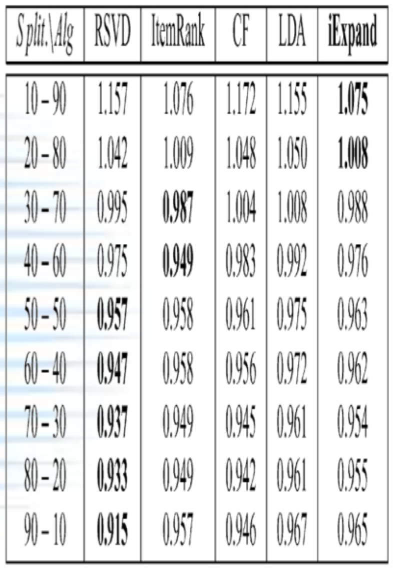
only once, and ratings in the test set have never been used in the training process. Therefore, in order to make a clearer and fairer comparison, we do not take the online recommendation into consideration.First of all, we show a comparison of the effectiveness of all the algorithms. Tables IV and V and the last figure show the performances of their recommendations with respect to different splits and different evaluation metrics. Table IV(a)–(c) illustrates the evaluation results of the DOA/Recall measures. The final figure demonstrates the top Kresults on the three data sets.

TABLE III
PARAMETER SETTINGS

Data set	α	β	K	l
MovieLens	0.001	0.080	300	1000
Book-Crossing	0.017	0.237	50	1000
Jester	0.545	0.118	40	1000



*Split**Alg*	RSVD	ItemRank	CF	LDA	iExpand
10 – 90	0.887	0.845	0.919	0.909	**0.844**
20 – 80	0.798	0.796	0.822	0.825	**0.795**
30 – 70	**0.770**	0.775	0.787	0.792	0.777
40 – 60	0.760	**0.749**	0.772	0.778	0.769
50 – 50	**0.751**	0.755	0.756	0.767	0.759
60 – 40	**0.748**	0.754	0.751	0.765	0.759
70 – 30	**0.747**	0.748	0.744	0.758	0.753
80 – 20	**0.740**	0.749	0.741	0.759	0.755
90 – 10	**0.739**	0.757	0.746	0.764	0.763



*Split**Alg*	RSVD	ItemRank	CF	LDA	iExpand
10 – 90	1.157	1.076	1.172	1.155	**1.075**
20 – 80	1.042	1.009	1.048	1.050	**1.008**
30 – 70	0.995	**0.987**	1.004	1.008	0.988
40 – 60	0.975	**0.949**	0.983	0.992	0.976
50 – 50	**0.957**	0.958	0.961	0.975	0.963
60 – 40	**0.947**	0.958	0.956	0.972	0.962
70 – 30	**0.937**	0.949	0.945	0.961	0.954
80 – 20	**0.933**	0.949	0.942	0.961	0.955
90 – 10	**0.915**	0.957	0.946	0.967	0.965

In addition, both LDA and iExpand reduce data dimensions, so they perform better when the data are dense, while SVD, another algorithm based on dimension reduction, does not perform well. This may because of the use of different de-composing techniques. Finally, as the main difference betweeniExpand and LDA is interest expansion or not and because iExpand can expand user interests and increase the diversity in a properly controlled manner, it performs much better than LDA in all the cases. This means interest expansion can lead to a better performance than only exploiting the current user interests.

E. Analysis of Parameters in Personalized Ranking

In this section, we provide an analysis of two parameters: the restart probability c and the step of random walk s.

To study the effect of c, we let it vary in the range of [0, 1). When it is 0, random walk will never restart. When c is close to 1, the performance of iExpand will be similar to the LDA algorithm. Fig. 9(a)–(c) shows the relationships between the best value of c with regard to Recall/DOA metrics and the size of training data set for iExpand on three benchmark data sets.

F. Understanding of Interests and Interest Expansion

In this section, we first show the interrelationships between latent interests and explicit interests, and then, we explain the advantages of interest expansion by examples.

To this end, we consider the first three latent interests ex-tracted from the MovieLens data set. Table VI lists the top five movies for each latent interest identified. As can be seen, all five movies in the first latent interest have the same genres which can be tagged as *Action*, *Adventure*, and *Fantasy*[3] or they can be labeled “Harrison Ford” (and contain one mistake), while movies in the second column all fall into *Comedy* and *Drama*. However, there are several types of movie genres for the third one. After a closer look, we find that all of these movies are generally recognized as *classic* movies and they all have won more than one *Oscar* award. Another observation is that themovie *Star Wars* is given high probability in both latent interests 1 and 3. This verifies that topic models can capture the multiple characteristics of each movie, and each characteristic can be resolved by other movies in the corresponding latent interest.

G. Discussion

In this section, we analyze the advantages and limitations of the iExpand method. From the experimental results, we can see that there are many key advantages of iExpand. First, iExpand models the implicit relations between users and items through a set of latent user interests. This three-layer representation leads to more accurate ranking recommendation results. Second, iExpand can save the computational cost by reducing the num-ber of *item* dimensions. This dimensionality reduction can also help to alleviate the sparseness problem which is inherent to many traditional collaborative-filtering systems. Third, iExpand enables diverse recommendations by the interest expansion. This can help to avoid the *overspecialization* problem. Finally, iExpand can deal with the cold-start recommendations. This means we only need several items or interests input by the new user, and then, the corresponding items this user may like can be predicted and recommended.

IV. RELATED WORK

In general, related work can be grouped into four categories. **The first category** has a focus on the graph-based collaborative-filtering methods. Here, the graph-basedcollaborative-filtering methods refer to those approaches which use the similarity of graph vertices to make recommendations[14], [18].

The second category includes the research work related totopic models, which are based upon the idea that documents are mixtures of topics, where a topic is a probability distribution over words. Many kinds of topic models have been proposed, among which PLSA [21] and LDA [8] are most widely used and studied.

The third category of related work has a focus on solvingthe *overspecialization* problem in recommender systems. This happens when the user is limited to being recommended the items that are "similar" (with respect to content) to those already rated [2].

The fourth category of related work is focused on solvingthe cold-start problem. Cold-start problem will happen when the recommender systems try to give recommendations to the users whose preference are underexplored or try to recommend the new items whose characteristics are also unclear [2]. Thus, it can be further classified as the item-side cold-start problem and the user-side cold-start problem.

V. CONCLUDING REMARKS

In this paper, we exploited user latent interests for developing an item-oriented model-based collaborative framework, named iExpand. Specifically, in iExpand, a topic-model-based method is first used to capture each user's interests. Then, a personalized ranking strategy is developed for predicting a user's possible interest expansion. Moreover, a diverse recommendation list is generated by using user latent interests as an intermediate layer between the user layer and the item layer. There are two key benefits of iExpand. First, the three-layer representation enables a better understanding of the interactions among users, items, and user interests and leads to more ac-curate ranking recommendation results. Second, since the user interests and the change of the interests have been taken into the consideration, iExpand can keep track of these changes and significantly mitigate the *overspecialization* problem and the cold-start problem.Finally, an empirical study has been conducted and the corresponding experimental results demonstratethat iExpand can lead to better ranking performances than state-of-the-art methods including two graph-based collaborative-filtering algorithms and two dimension-reduction-based algorithms.

REFERENCES

[1] Movielens Datasets, 2007. [Online]. Available: http://www.grouplens.org/ node/73#attachments

[2] G. Adomavicius and A. Tuzhilin, "Toward the next generation of rec-ommender systems: A survey of the state-of-the-art and possible exten-sions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.

[3] G. Adomavicius and A. Tuzhilin, "Context-aware recommender systems," in *Recommender Systems Handbook*. New York: Springer-Verlag, 2011, pp.217–253.

[4] H. J. Ahn, "A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem," *Inf. Sci.*, vol. 178, no. 1, pp. 37–51, Jan. 2008.

[5] A. Asuncion, M. Welling, P. Smyth, and Y. W. Teh, "On smoothing and inference for topic models," in *Proc. Int. Conf. UAI*, 2009, pp. 27–34.

[6] D. Billsus and M. J. Pazzani, "User modeling for adaptive news access," *User Model.User-Adapted Interaction*, vol. 10, no. 2, pp. 147–180,2000.

[7] D. M. Blei and J. D. Lafferty, "A correlated topic model of science," *Ann.Appl. Statist.*, vol. 1, no. 1, pp. 17–35, 2007.

[8] D. M. Blei, Y. N. Andrew, and I. J. Michael, "Latent Dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, 2003.

[9] K. R. Canini, L. Shi, and T. L. Griffiths, "Online inference of topics with Latent Dirichlet allocation," in *Proc. 12th Int. Conf. AISTATS*, 2009, vol. 5, pp. 65–72.

[10] W. Chen, J. C. Chu, J. Luan, H. Bai, Y. Wang, and E. Y. Chang, "Collabo-rative filtering for orkut communities: Discovery of user latent behavior," in *Proc. 18th Int. Conf. WWW*, 2009, pp. 681–690.

[11] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *J. Amer. Soc. Inf.Sci.*, vol. 41, no. 6, pp. 391–407, 1990.

[12] S. Debnath, N. Ganguly, and P. Mitra, "Feature weighting in content based recommendation system using social network analysis," in *Proc. 17th Int.Conf. WWW*, 2008, pp. 1041–1042.

[13] Y. Ding and X. Li, "Time weight collaborative filtering," in *Proc. 14thACM Int. CIKM*, 2005, pp. 485–492.

[14] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens, "Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 3, pp. 355–369, Mar. 2007.

[15] S. Funk, Netflix Update: Try This at Home, 2006. [Online]. Available: http://sifter.org/~simon/journal/20061211.html

[16] Y. Ge, H. Xiong, A. Tuzhilin, K. Xiao, M. Gruteser, and M. J. Pazzani, "An energy-efficient mobile recommender system," in *Proc. 16th ACMSIGKDD Int. Conf. KDD*, 2010, pp. 899–908.

[17] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: A con-stant time collaborative filtering algorithm," *Inf. Retrieval*, vol. 4, no. 2, pp.133–151, Jul. 2001.

[18] M. Gori and A. Pucci, "A random-walk based scoring algorithm applied to recommender engines," in *Proc. 8th Int. Workshop Knowl. Discov. Web(WebKDD)—Advances in Web Mining and Web Usage Analysis*, 2006, pp.127–146.

[19] T. L. Griffiths and M. Steyvers, "Finding scientific topics," *Proc. Nat.Acad. Sci. U.S.A. (PNAS)*, vol. 101, pp. 5228–5235, 2004.

[20] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluat-ing collaborative filtering recommender systems," *ACM Trans. Inf. Syst.(TOIS)*, vol. 22, no. 1, pp. 5–53, Jan. 2004.

[21] T. Hofmann, "Probabilistic latent semantic analysis," in *Proc. 15th Conf.UAI*, 1999, pp. 289–296.

[22] T. Hofmann, "Latent semantic models for collaborative filtering," *ACMTrans. Inf. Syst. (TOIS)*, vol. 22, no. 1, pp. 89–115, Jan. 2004.

[23] Z. Huang, H. Chen, and D. Zeng, "Applying associative retrieval techniques to alleviate thesparsity problem in collaborative filter-ing," *ACM Trans. Inf. Syst. (TOIS)*, vol. 22, no. 1, pp. 116–142, Jan. 2004.

[24] L. Iaquinta, M. de Gemmis, P. Lops, and G. Semeraro, "Introducing serendipity in a content-based recommender system," in *Proc. HIS*, 2008,