Exploiting User Interests for Collaborative Filtering: Interests Expansion via Personalized Ranking

Qi Liu, Enhong Chen
University of Science and Technology of China
{feiniaol,cheneh}@mail.ustc.edu.cn

Hui Xiong
Rutgers University
hxiong@rutgers.edu

Chris H.Q. Ding
University of Texas at Arlington
chqding@uta.edu

ABSTRACT

In real applications, a given user buys or rates an item based on his/her interests. Learning to leverage this interest information is often critical for recommender systems. However, in existing recommender systems, the information about latent user interests are largely under-explored. To that end, in this paper, we propose an interest expansion strategy via personalized ranking based on the topic model, named iExpand, for building an interest-oriented collaborative filtering framework. The iExpand method introduces a three-layer, user-interest-item, representation scheme, which leads to more interpretable recommendation results and helps the understanding of the interactions among users, items, and user interests. Moreover, iExpand strategically deals with many issues, such as the *overspecialization* and the cold-start problems. Finally, we evaluate iExpand on benchmark data sets, and experimental results show that iExpand outperforms state-of-the-art methods.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information Filtering*

General Terms

Algorithms, Experimentation

Keywords

Collaborative filtering, interests expansion, personalized ranking

1. INTRODUCTION

Collaborative filtering techniques have broad applications and have been widely studied, since these techniques only require the information about user interactions. However, existing collaborative filtering methods often directly exploit the information of users' interaction with the systems. In other words, they make recommendations by learning a *"user-item"* dualistic relationship. Therefore, existing methods neglect an important fact that there are many latent user interests which can influence users' 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. Indeed, it is necessary to leverage this three-layer representation, since this representation leads to more interpretable recommendation results and helps the understanding of the interactions among users, items and user interests.

Furthermore, when leveraging the information of user interests, we must be aware that users' interests can change from time to time. Nevertheless, traditional collaborative filtering systems cannot capture these changes, and thus are prone to the *"overspecialization"* problem. The key challenge is how to model latent user interests and their potential changes in collaborative filtering systems.

To address the above challenges, we propose an interest-oriented collaborative filtering system, named iExpand. Specifically, each user interest is first captured by a latent factor. Then, we extract users' latent interests and learn the transition probabilities between different interests. Moreover, we model the possible expansion process of users' interests by personalized ranking. In other words, we exploit a personalized ranking strategy to predict the next possible interest for each user. There are three key advantages of iExpand. First, iExpand models the implicit relations between users and items through a set of latent interests, this representation leads to more interpretable recommendation results. Second, iExpand can save the computational cost and help to alleviate the sparseness problem by reducing the number of *item* dimensions. Third, iExpand enables diverse recommendations by the interests expansion. This can help to avoid the *overspecialization* problem.

Finally, iExpand makes recommendations by directly ranking the candidate items. Therefore, in the experiments, we report the ranking prediction accuracy. As collaborative filtering is often formulated as a regression or rating prediction problem, we also report the comparison results.

2. USER INTERESTS EXPANSION

In this section, we first introduce the framework of the iExpand. Then, we describe each step in detail. In addition, we address the parameters selection and computational complexity issues.

The iExpand model assumes that a user's rating behavior depends on an underlying 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. What's more, the model assumes that the order of items in a user's rating list can be neglected. In correspond to LDA, the users, items and latent interests are documents, words and topics respectively [2]. Figure 1 illustrates the framework of the iExpand model. Each step of the model is introduced in the fol-

CIKM'10, October 26–30, 2010, Toronto, Ontario, Canada.

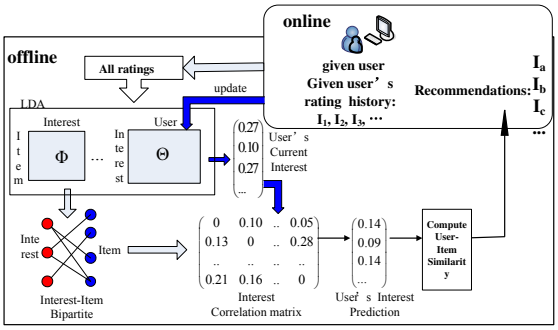

Figure 1: The Framework of the iExpand Model. Gray arrows shows the general process of the model, while blue arrows shows the procedure of online recommendations.

lowing subsections. Table 1 lists all mathematical notations used in the paper.

Table 1: Mathematical Notations.

Notation	Description
$U = \{U_1, U_2, ..., U_M\}$	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

2.1 Representation of User Interests

In this subsection, we show how to extract the information about user latent interests from the LDA model. The information about latent interests includes the probability distribution of each user over each interest (i.e.,user U_i's distribution over interest T_j is θ_{ij}), the probability distribution of each interest over each item (i.e., the distribution of interest T_j over item I_i is ϕ_{ij}), and the distribution of each interest ($\vec{\vartheta}_i$). In this paper, we choose the Gibbs sampling technique [6], which provides an efficient method for extracting a set of interests from a large ratings data set.

It is worth distinguishing between our user interests and the latent topics in topic models. In iExpand, each user has a distribution on the spectrum of interests, whereas in PLSA/LDA a topic is a latent variable and the distributions are specified by the topic,i.e., they are class (topic)-conditional distributions. Thus the model representation of iExpand and PLSA/LDA are significantly different.

2.2 User Interests Correlation Graph

In this subsection, we describe how to compute the transition probabilities between latent interests. In order to construct the correlation graph of latent interests, we use the items as intermediary entities. φ is created to estimate each item's probability distribution over interests and φ_{ij} can be estimated by Equation (1):

$$\varphi_{ij} = P(T_j|I_i) = \frac{P(T_j, I_i)}{P(I_i)} = \frac{\phi_{ij}\vec{\vartheta}_j}{\sum_{k=1}^{K}\vec{\vartheta}_k\phi_{ik}} \quad (1)$$

In iExpand, we model the correlations between interests in the form of probabilities. At first, we use a bipartite graph $G =< X, E >$ to represent the relationships between items and interests, with the vertice set $X=I \cup T$. In G, the weight of the edge from interest T_j to item I_i is ϕ_{ij}, and the weight of the edge from I_i to T_j is φ_{ij}. Then, by projecting G, we get the relationships between interests represented by ψ. Also, ψ_{ij} indicates the recommending strength of interest T_i for T_j and it can be computed by Equation (2):

$$\psi_{ij} = P(T_j|T_i) = \sum_{n=1}^{N} P(T_j|I_n)P(I_n|T_i) = \sum_{n=1}^{N}\varphi_{nj}\phi_{ni} \quad (2)$$

At last, the bipartite graph is transformed into a correlation graph which describes the interest relations, and ψ is the correlation matrix. In terms of correlation matrix, ψ_{ij} means the coefficient of correlation between T_i and T_j from $T_i's$ view. In terms of random walk, ψ_{ij} is the probability that current state jumps from T_i to T_j.

2.3 User Interests Expansion

In this subsection, we describe the solution for user interests expansion, to which we use **PageRank** personalized ranking strategy on the user interests correlation graph. Given a users' interest vector, we do repeat PageRank iterations until convergence. The final converged vector contains the expanded user interests. One can also view this as predicting the next possible interest for each user. Thus, we can make diverse recommendations in a systematic way.

The algorithmic approach here is the personalized ranking [7]. First, we represent U_i's current interest model through vector $\vec{\theta}_i^{(0)}$ in which the j-th entry $\vec{\theta}_i^{(0)}(j)$ corresponds to latent interest T_j, initialized as θ_{ij}, and $\vec{\theta}_i^{(0)}$ is the probability distribution when random walk starts. In the next, let $\vec{\theta}_i^{(0)}$ perform Random Walk with Restart (RWR) [5] on the correlation graph. Let us consider a random walk that starts from $\vec{\theta}_i^{(0)}$, when arriving at T_j, it randomly chooses T_j's neighbors and keeps walking. In addition to making such decisions, the random walker goes back to the starting point with a certain probability c. For all the users, their one-step updates from step s to step $(s+1)$ can be formalized as Equation (3):

$$\begin{cases} \theta^{(s)} = \theta, & s = 0 \\ \theta^{(s+1)} = (1-c)\theta^{(s)}\psi + c\theta, & s \geqslant 0 \end{cases} \quad (3)$$

In $\theta^{(s)}$, $\theta_{ij}^{(s)}$ means the steady-state probability that a random walk starting from U_i and stops at T_j after s steps, meanwhile it implies the affinity of T_j with respect to U_i. The personalized ranking is run for all users simultaneously, and it only takes several steps before $\theta^{(s)}$ converges. The parameter c indicates the restart probability, and $(1-c)$ represents how much relationship is lost in each step.

2.4 Optimal Item Recommendation

In this subsection, we describe the ranking of the items and the generation of recommendation lists. In iExpand, the items are ranked by their relevance with any given user. The user's possible distribution on latent interests, serves as intermediary entities:

$$P(I_j|U_i) = \sum_{k=1}^{K} P(I_j|t=k)P_s(t=k|U_i) = \sum_{k=1}^{K}\phi_{jk}\theta_{ik}^{(s)} \quad (4)$$

It is easy to obtain the top-K recommendations by ranking the candidate items. Thus, iExpand directly generates recommendations without the step of predicting rating scores.

iExpand can also be used as a rating prediction method. Here, *Pearson Correlation* can be used to compute user similarities. Then, the rating from user U_i to item I_j can be predicted by Equation (5):

$$\hat{r}_{i,j} = \bar{r}_i + \frac{\sum_{U_h \in Neighbor(U_i)} Sim(U_i, U_h) * (r_{h,j} - \bar{r}_h)}{\sum_{U_h \in Neighbor(U_i)} |Sim(U_i, U_h)|} \quad (5)$$

What we discussed above is about how to make recommendations in a general iExpand process. However, in real-world applications, we face the challenge of online recommendations. Since users' interest distributions may change quickly and the correlation of interests evolves slowly, we can update both the inference pro-

cess and the correlation graph periodically offline, while renew the user's interests whenever he/she rates.

2.5 Estimating the Parameters

In this subsection, we show the value selections for parameters: the hyperparameters α and β, and interest number K. At first, we select values for α and β. There are many ways for learning them, among which Minka's fixed-point iteration is widely used [10]. In iExpand, each step of iteration is formalized as Equation (6):

$$\begin{aligned} \alpha^* &\longleftarrow \frac{\alpha \sum_{m=1}^{M}\sum_{k=1}^{K}[\Psi(C_{mk}^{MK}+\alpha)-\Psi(\alpha)]}{K\sum_{m=1}^{M}[\Psi(\sum_{k=1}^{K}C_{mk}^{MK}+K\alpha)-\Psi(K\alpha)]}, \\ \beta^* &\longleftarrow \frac{\beta \sum_{k=1}^{K}\sum_{n=1}^{N}[\Psi(C_{nk}^{NK}+\beta)-\Psi(\beta)]}{N\sum_{k=1}^{K}[\Psi(\sum_{n=1}^{N}C_{nk}^{NK}+N\beta)-\Psi(N\beta)]} \end{aligned} \tag{6}$$

Next, we choose the right value for the interest number K. Until now, one possible approach for setting this value is to compute the likelihood of the test data under different K, then the best one is chose by a grid search. In this paper, we refer to an approach named *Chib-style estimation* [9]. As the posterior probability depends on both α, β and K, we combine these factors together and propose a parameter learning algorithm, as shown in Algorithm 1.

Algorithm 1: Estimating Parameters (a,b)

input : a, the initial value of α; b, the initial value of β;
output: the best values for α, β and K
for *all candidate* K **do**
  Initialize $\alpha = a, \beta = b$;
  for *loop* $\leftarrow 1$ *to MAX_LOOP* **do**
    Gibbs sampling;
    Update α, β by Equation 6;
  posterior=log(*Chib-style estimation*(α, β, K));
  Record the maximum posterior with its α, β and K;
Return the best values for α, β and K

2.6 Computational Complexity

In this subsection, 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. The time cost of bipartite graph projection is $O(N \cdot K^2)$, and for random walk it is $O(s \cdot M \cdot K^2)$. Since $K << M$ and $K << N$, the total computational complexity for general iExpand process is $O(M \cdot N \cdot K \cdot l)$. As 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 Gibbs sampling and personalized ranking or rating prediction for current user, both of which can be done efficiently.

3. EXPERIMENTAL RESULTS

In this section, we present the experimental results, and we demonstrate: (1) a performance comparison between iExpand and many other methods, (2) the understanding of interests and the expansion.

Data Sets. All the experiments were performed on two real-world data sets: MovieLens and Book-Crossing. The former one [1] contains 100,000 ratings from 943 users for 1,682 movies. In the latter one [12], we choose the most rated 996 users and their 91,084 ratings on 1,696 books. The split named as x-(100-x) means x percent ratings serve for training and the remaining ratings for test.

Benchmark Methods. For the ranking purpose, we compare iExpand with ItemRank [5], L^+[3], as well as LDA and SVD. For the rating purpose, we implemented the user based collaborative filtering (UCF) [8], RSVD [4], LDA and ItemRank [5].

Evaluation Metrics. We adopted *Degree of Agreement* (DOA) and *Hit Ratio* (HR) to evaluate the ranking accuracies. DOA measures the percentage of item pairs ranked in the correct order [5]. HR measures the ratio of the number of hits [11]. For the evaluation of the rating effectiveness, we choose the widely used *Mean Absolute Error* (MAE) and *Root Mean Squared Error* (RMSE).

Parameters selection. Before the performance comparison, we investigate the learning of two parameters: hyperparameters and the interest number. Here, the first 893 users in MovieLens are used as training data and the rest users for test, while For Book-Crossing the first 900 users are treated as training samples. Finally, the results of parameters selection are summarized in Table 2.

Table 2: Parameter settings.

Data set	α	β	K	l
MovieLens	0.001	0.08	300	1000
Book-Crossing	0.017	0.237	50	1000

In iExpand, there are two other parameters for personalized ranking: the restart probability c and the step s.In experiments, we let c lie in the range of [0, 1), and iExpand achieves best performance by just a few steps of random walk (less than 10 steps).

Performance Comparison. The performances of their recommendations are illustrated in Table 3. In terms of ranking, iExpand outperforms the other algorithms with a significant margin in each split. Another observation is that when the training set becomes larger and denser, the improvement made by iExpand compared to LDA becomes less obvious. The reason is that, when there are enough interactions between a user and the system, the user's preference has been decided and there will be no much difference from his current interest distribution to the next possible interest distribution. In terms of rating, iExpand performs best in the sparsest splits, while in general RSVD outperforms the other methods. On the sparse splits, the methods that can discover the indirect correlations (i.e., iExpand and ItemRank) get better results. While on the remaining splits, the rating oriented methods (i.e., RSVD and UCF) generally perform better. Another observation is that, the two types of evaluation metrics DOA/HR and MAE/RMSE lead to inconsistent judgements which have been discussed in previous works.

The Understanding of Interests and Interests Expansion. In the previous, we do not distinguish latent interests and explicit interests. The former is a latent factor extracted by topic model, while the latter is the one identified in the real world, and we use latent interests to simulate explicit interests. The researches on topic models have shown their one-to-one correspondence. The question is whether every latent interest has a real meaning for use in iExpand? To this end, we consider the first three latent interests extracted from the MovieLens data set. Table 4 lists the top-5 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* or they can be labeled "Harrison Ford"(and contain one mistake). While movie 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 movie *Star wars* is given high probability in both latent interests 1 and 3. This verifies that topic models can capture the multiple aspects of each movie, and each aspect can be resolved by other movies in the corresponding latent interest. The above analysis means that, even for collaborative filtering, every latent factor extracted by topic models still has a real meaning, although the in-

Table 3: A Performance comparison of the effectiveness of different algorithms.

(a) The comparison of ranking performances on the MovieLens data set (Left: DOA in %, Right: HR in %).

Split.*Alg*	SVD	ItemRank	L^+	LDA	**iExpand**
10 – 90	52.504	76.694	66.853	75.726	**79.043**
20 – 80	67.990	84.402	84.022	83.477	**85.789**
40 – 60	79.882	87.390	88.774	88.683	**89.060**
60 – 40	85.567	88.723	90.340	89.992	**90.626**
80 – 20	85.243	89.002	90.636	90.823	**91.310**

Split.*Alg*	SVD	ItemRank	L^+	LDA	**iExpand**
10 – 90	10.521	24.901	13.467	21.416	**28.196**
20 – 80	13.578	31.940	22.260	33.093	**35.514**
40 – 60	17.456	31.639	28.055	34.957	**35.240**
60 – 40	17.530	28.183	25.249	30.463	**31.507**
80 – 20	11.363	20.761	19.342	23.008	**23.467**

(b) The comparison of ranking performances on the Book-Crossing data set (Left: DOA in %, Right: HR in %).

Split.*Alg*	SVD	ItemRank	L^+	LDA	**iExpand**
10 – 90	46.359	57.734	52.839	57.276	**60.174**
20 – 80	52.711	62.145	60.368	59.891	**62.670**
40 – 60	62.478	66.998	65.611	67.165	**68.061**
60 – 40	67.380	68.157	69.152	69.885	**70.437**
80 – 20	69.881	69.576	71.346	71.927	**72.582**

Split.*Alg*	SVD	ItemRank	L^+	LDA	**iExpand**
10 – 90	5.058	9.210	5.663	7.839	**12.771**
20 – 80	5.708	12.240	6.468	10.130	**13.380**
40 – 60	6.870	12.509	7.070	12.528	**13.589**
60 – 40	6.990	9.815	7.854	11.653	**12.172**
80 – 20	4.547	6.414	6.591	8.184	**8.897**

(c) The comparison of rating performances on the MovieLens data set (Left: MAE, Right: RMSE).

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**
40 – 60	0.760	**0.749**	0.772	0.778	0.769
60 – 40	**0.748**	0.754	0.751	0.765	0.759
80 – 20	**0.740**	0.749	0.741	0.759	0.755

Split.*Alg*	RSVD	ItemRank	CF	LDA	**iExpand**
10 – 90	1.157	1.076	1.172	1.155	**1.075**
20 – 80	1.042	1.008	1.048	1.050	**1.007**
40 – 60	0.975	**0.949**	0.983	0.992	0.976
60 – 40	**0.947**	0.958	0.956	0.972	0.962
80 – 20	**0.933**	0.949	0.942	0.961	0.955

Table 4: Top movies in the first three latent user interests.

Latent interests	Interest 1	Interest 2	Interest 3
movies	Back to the Future Return of the Jedi Raiders of the Lost Ark Star Wars The Empire Strike Back	Secrets & Lies Il Postino: The Postman My Life as a Dog Sunset Blvd A Room with a View	Star Wars The English Patient The Silence of the Lambs Godfather Pulp Fiction

terpretation of each factor may not be as easy and precise as that in text applications based on topic models.

In the experiments, we can see that iExpand with interest expansion can lead to a better performance than the LDA which only exploiting the current user interests. The reason is that interest expansion can predict the next possible interest for each user in a properly controlled manner, and make diverse recommendations. Thus, this helps to avoid the overspecialization problem. In other words, the interest expansion is more appropriate to capture the diversified interests and find potential interests for the users. While the problem of how these interests expands from one to another needs more detailed analysis and this is beyond the discussion of this paper. Meanwhile, we would like to point out that this advantage is meaningful to most of the users which can be seen from the results of the performance comparisons shown in Table 3, while this does not mean it will work for every single user, and there may exist users whose interest expansion is different from the majority.

4. CONCLUDING REMARKS

In this paper, we exploited latent interests for developing an interest-oriented 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 user's possible interests 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. Finally, an empirical study has been conducted on two benchmark data sets, and the results demonstrate that iExpand can lead to better ranking performances than state-of-the-art methods.

Acknowledgment. This work was supported by grants from Natural Science Foundation of China (*No*.60775037), the Key Program of Natural Science Foundation of China (*No*.60933013), the National High Technology Research and Development Program of China (*No*.2009*AA*01*Z*12), and the Research Fund for the Doctoral Program of Higher Education of China (20093402110017).

5. REFERENCES

[1] Movielens datasets. *URL: http://www.grouplens.org/node/73#attachments*, 2007.

[2] D.M. Blei, Y. N. Andrew, and I.J. Michael. Latent Dirichlet Allocation. *Journal of Machine Learning Research,3*, pages 993–1022, 2003.

[3] 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.*, 19(3), pages 355–369, 2007.

[4] S. Funk. Netflix update: Try this at home. *URL: http://sifter.org/ simon/journal/20061211.html*, 2006.

[5] M. Gori, and A. Pucci. A random-walk based scoring algorithm applied to recommender engines. In *WebKDD'06*, pages 127–146, 2006.

[6] T.L. Griffiths and M. Steyvers. Finding scientific topics. In *PNAS'04* vol. 101, pages 5228– 5235. 2004.

[7] G. Jeh, and J. Widom. Scaling Personalized Web Search. In *WWW'03*, pages 271–279, 2003.

[8] R. Paul, I. Neophytos, S. Mitesh, B. Peter and R. John. GroupLens: an open architecture for collaborative filtering of netnews. In *CSCW'94*, pages 175–186, 1994.

[9] H.M. Wallach, I. Murray, R. Salakhutdinov and D.M. Mimno Evaluation methods for topic models. In *ICML'09*, pages 1105–1112, 2009.

[10] H.M. Wallach. Structured topic models for language. *PhD thesis, University of Cambridge*, 2008.

[11] H. Yildirim and M. S. Krishnamoorthy. A random walk method for alleviating the sparsity problem in collaborative filtering. In *RecSys'08*, pages 131–138, 2008.

[12] C. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In *WWW'05*, pages 22–32, 2005.