Journal of Zhejiang University SCIENCE A
ISSN 1673-565X (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Random walk models for top-*N* recommendation task*

Yin ZHANG†, Jiang-qin WU, Yue-ting ZHUANG

(*School of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China*)

†E-mail: zhangyin98@cs.zju.edu.cn

Received Jan. 8, 2009; Revision accepted Apr. 30, 2009; Crosschecked May 22, 2009

Abstract: Recently there has been an increasing interest in applying random walk based methods to recommender systems. We employ a Gaussian random field to model the top-*N* recommendation task as a semi-supervised learning problem, taking into account the degree of each node on the user-item bipartite graph, and induce an effective absorbing random walk (ARW) algorithm for the top-*N* recommendation task. Our random walk approach directly generates the top-*N* recommendations for individuals, rather than predicting the ratings of the recommendations. Experimental results on the two real data sets show that our random walk algorithm significantly outperforms the state-of-the-art random walk based personalized ranking algorithm as well as the popular item-based collaborative filtering method.

Key words: Random walk, Bipartite graph, Top-*N* recommendation, Semi-supervised learning
doi:10.1631/jzus.A0920021 **Document code:** A **CLC number:** TP393.09

INTRODUCTION

Internet surfers are often confronted with the problem of information overloading, i.e., finding desired information from huge amounts of content is time-consuming. Recommender systems aim at alleviating this problem by automatically delivering to users items that they should appreciate. In this paper we exclusively focus on the ratings-based recommender systems, in which matching and delivering recommendations are based on the historical ratings of items by users. The task of recommender systems can be described in two forms: prediction and top-*N* recommendation. The former is the prediction of a numerical value, expressing the predicted likeness of the item for the active user. The latter is the recommendation of a list of items that the active user will like most. We just focus on the latter form of top-*N* recommendation, since our random walk model directly generates the ranked list of items biased for the individual, instead of predicting scores of items unseen by the individual.

In this paper we consider the top-*N* recommendation task as a semi-supervised learning problem on the augmented user-item bipartite graph constructed from the ratings database. We employ a Gaussian random field (Zhu *et al.*, 2003) to model the top-*N* recommendation task as follows: (1) an active label is assigned to the active user for which the recommendations are generated; (2) a dummy label is assigned to the dummy node in the augmented bipartite graph, in order to penalize the long path of an absorbing random walk (ARW); (3) a set of unlabeled objects correspond to the remaining users and items; (4) a set of constraints correspond to associations between labeled and unlabeled objects, i.e., the augmented bipartite graph built from the ratings database. The recommendation task is then transformed into calculating probabilities of assigning the active label to unlabeled items given the observed object, i.e., the active user in the context of recommendation.

The contributions of our work are: (1) the use of a Gaussian random field to formulate the top-*N* recommendation task as a semi-supervised learning problem; (2) the use of an ARW algorithm for

* Project supported by the National Natural Science Foundation of China (Nos. 60525108 and 60533090), the National Hi-Tech Research and Development Program (863) of China (No. 2006AA010107), and the Program for Changjiang Scholars and Innovative Research Team in University, China (No. IRT0652)

generating the ranked list of items biased for the individual, which is inferred from a Gaussian random field; (3) the use of experimental results to demonstrate that our approach consistently outperforms the other graph- or item-based methods for the top-*N* recommendation task for the MovieLens and MovieRating data sets.

This paper is organized as follows: Section 2 introduces related works for applying random walk based methods to recommender systems. Section 3 formally formulates the recommendation task as a semi-supervised learning problem. Section 4 describes how to use a Gaussian random field to model the label distribution for each user or item under the constraint of an augmented user-item bipartite graph, based on which an ARW algorithm is inferred for generating the personalized items ranking. Section 5 presents the details of experiments, including experimental setup, evaluation metrics, scoring algorithms, comparison of algorithms, and parameter sensitivity of our approach. Finally, conclusions and future work are presented in Section 6.

RELATED WORKS

Recently there has been considerable interest in applying the random walk based methods in the field of recommender systems (Breese *et al.*, 1998). There are direct and indirect ways to employ a random walk model in the context of a top-*N* recommendation task. The former method (Gori and Pucci, 2007; Cheng *et al.*, 2007) is to directly compute the similarities between a given user and all the movies. The latter (Brand, 2005; Fouss *et al.*, 2007) is to compute the similarities among users (or items), which are then used to form the neighborhood of a given user (or item) in the classical framework of collaborative filtering.

For direct methods, Gori and Pucci (2007) exploited a way of projecting a bipartite graph built from a ratings database into the correlation graph that just consists of item nodes, and then applied a personalized ItemRank algorithm on the built correlation graph. Cheng *et al.*(2007) proposed a query centered random walk on the sub-graph induced by multi-way clustering on the *k*-partite graph constructed from multiple features relevant to the recommendation task. Singh *et al.*(2007) proposed an ARW on the user-item bipartite graph, enhanced with user-user (social) links. The probability distribution of an ARW can be interpreted as personalized ranks for each user. Kunegis and Schmidt (2007) introduced the method to incorporate negative edge weights into calculating the resistance distance on the user-item bipartite graph. Huang *et al.*(2004) applied a transitive retrieval framework and related spreading activation algorithms to explore transitive associations among consumers through their past transactions.

For indirect methods, Brand (2005) developed a geometrization of the Markov chain that provides a key similarity measure for information retrieval—the cosine (correlation) angle between two states, which empirically proves to be highly predictive of future choices made by individuals. Fouss *et al.*(2007) presented the pseudoinverse of the Laplacian matrix (L^+) for characterizing the similarity between nodes of a weighted and undirected graph, based on a Markov-chain model of a random walk on the graph. Yildirim and Krishnamoorthy (2008) proposed a novel item-oriented algorithm that first infers transition probabilities between items based on their similarities and model finite length random walks on the item space to compute predictions.

Our approach is categorized into a direct method and based on the label propagation algorithm in (Zhu *et al.*, 2003), which is also employed for keyword generation from the query logs of the search engine (Fuxman *et al.*, 2008). Our approach is applied to a graph with absorbing states. However, PageRank-like methods are applied to a graph without absorbing states. Baluja *et al.*(2008) presented an adsorption algorithm based on the analysis of the user-video click graph from www.youtube.com to provide personalized video suggestions for users. The adsorption algorithm is a modification of the algorithm in (Zhu *et al.*, 2003). Our approach differs from the adsorption algorithm in that: (1) our approach takes into consideration the degree of the node, while the adsorption algorithm does not; (2) the adsorption algorithm introduces injection probabilities besides dummy probabilities.

In the field of collaborative filtering, there are also works for simultaneously utilizing the user and item dimensions. Symeonidis *et al.*(2008) proposed a novel nearest-bi-clusters algorithm, which uses a new similarity measure that achieves partial matching of users' preferences. George and Merugu (2005) used bi-clusters to improve the scalability of collaborative filtering. Wang *et al.*(2006) proposed the similarity fusion between the user- and item-based methods, using also data from a third source (ratings of other similar users on other similar items).

PROBLEM STATEMENT

Ratings-based recommender systems can be formulated in a perspective of semi-supervised learning. The input of the derived semi-supervised learning problem is given as follows:

(1) A ratings database R. The ratings database contains many records (u, y, r_{uy}), where user $u\in U$ rated $y\in Y$ with a score of r_{uy}, U denotes the set of all users, and Y denotes the set of all items. Intuitively, R can be deemed to be the bipartite graph $G=(V, E)$, where V consists of all user and item nodes, a weighted edge $e\in E$ between a user node u and item node y corresponds to a triple (u, y, r_{uy}) in R, r_{uy} denotes the weight of edge between user u and item y. Moreover, there are no edges among the same set of nodes (users or items). Fig.1 shows an augmented bipartite graph transformed from a pseudo ratings database, in which every user and item are connected to a dummy node d (dashed line in Fig.1).

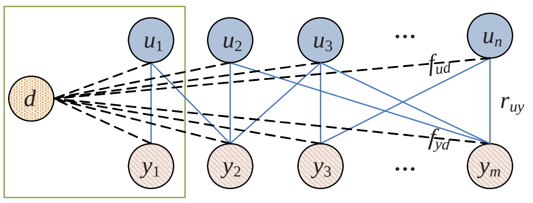

Fig.1 Viewing a ratings database as a bipartite graph augmented with a dummy node
f_{ud}: the weight of the edge between user u and dummy node d; f_{yd}: the weight of the edge between item y and dummy node d

(2) A set of labels $C=\{c_a, \omega\}$. The active label c_a is allowed only to be assigned to the active user u_a for whom the top-N recommendations are delivered. The dummy class ω is only allowed to be assigned to the dummy node d. If an ARW ends up in the dummy node d, this walk will generate a dummy label. The dummy node d is used to penalize the long path in an ARW. Since each user and item are connected to the dummy node, the longer an ARW, the higher the probability of ending up in the dummy node d.

(3) A seed set $S=\{<u_a, c_a>, <d, \omega>\}$, where the label c_a is manually assigned to the active user u_a. In other words, the active user u_a is waiting for recommendations.

Given R, C and S, the goal of our approach is to compute the probabilities of labeling $c_a\in C$ to items in Y unseen by the active user u_a, according to which the personalized ranked list $rl(u_a)$ of unseen items for the active user u_a is produced. More formally, the goal is to obtain the probability $l(y)$ that a random walk starting from an item y ends up in the active user u_a, in the context of augmented bipartite graph G built from a ratings database R.

ABSORBING RANDOM WALK MODEL ON AUGMENTED BIPARTITE GRAPH

We introduce a Gaussian Markov random field to model the label distribution for each user or item, and pair-wise relationships between users and items, in the context of the recommendation task. Let $l(u)$ and $l(y)$ be the probabilities of assigning the label c_a to user u and item y, respectively. The seed set S in our case indicates that user u_a is manually assigned to the label c_a, i.e., the value of $l(u_a)$ is definitely equal to 1, formally $l(u_a)=1$. Since the dummy node d is manually assigned to the label ω, the value of $l(d)$ is definitely equal to 0, i.e., $l(d)=0$.

The goal of semi-supervised classification is to find an optimal configuration to probability vectors $\boldsymbol{l}(\boldsymbol{u})$ and $\boldsymbol{l}(\boldsymbol{y})$ for all users and items by maximizing the posterior probability $P(\{\boldsymbol{l}(\boldsymbol{u})\},\{\boldsymbol{l}(\boldsymbol{y})\}|S)$. Given the Markov assumption, we can decompose the joint probability distribution into factors defined over maximal cliques in the augmented bipartite graph G. An example maximal clique C (outlined in the squared box of Fig.1) comprises three nodes: user u_1, item y_1 and dummy node d. Formally, the joint probability distribution is factorized as follows:

$$P(\{\boldsymbol{l}(\boldsymbol{u})\},\{\boldsymbol{l}(\boldsymbol{y})\},S)\propto\prod_{(u,y)\in E}\psi_{uyd}(l(u),l(y),l(d)), \quad (1)$$

where $\psi_{uyd}(l(u),l(y),l(d))$ is a potential function defined to quantize the relationships among user u, item

y and dummy node d, if an edge (u, y) exists on the augmented bipartite graph. Furthermore, we factorize $\psi_{uyd}(l(u),l(y),l(d))$ as follows:

$$\psi_{uyd}(l(u),l(y),l(d)) = \phi_{uy}(l(u),l(y)) \cdot \phi_{ud}(l(u),l(d)) \cdot \phi_{yd}(l(y),l(d)). \quad (2)$$

Considering that a high rating score r_{uy} implies a strong association between the user u and item y, we use the Gaussian function as the potential function, the minimization criterion of which requires that for edges (u, y) with a large rating score r_{uy}, the labels l_u and l_y should be close, so that the value $r_{uy}(l_u-l_y)^2$ is minimized. Likewise, the impacts of f_{yd} and f_{ud} are similar to that of r_{uy}. Moreover, in order to counteract the adverse effect of high-degree nodes whose neighboring nodes are often of no common interest, we propose that the value of probability of each node should be normalized by the square root of the sum of edge weights; hence, the potential functions are defined as follows:

$$\phi_{uy}(l(u),l(y)) = \exp\left(-r_{uy}\left(l(u)/\sqrt{d_u} - l(y)/\sqrt{d_y}\right)^2\right), \quad (3)$$

$$\phi_{ud}(l(u),l(d)) = \exp\left(-f_{ud}\left(l(u)/\sqrt{d_u}\right)^2\right), \quad (4)$$

$$\phi_{yd}(l(y),l(d)) = \exp\left(-f_{yd}\left(l(y)/\sqrt{d_y}\right)^2\right), \quad (5)$$

where $d_y = \sum_{u:(y,u)\in E} r_{uy}$ is the sum of edge weights between item y and its neighboring users, $d_u = \sum_{y:(u,y)\in E} r_{uy}$ is the sum of edge weights between user u and its neighboring items, f_{ud} is the weight of the edge from user u to dummy node d, and f_{yd} is the weight of the edge from item y to dummy node d. Due to $l(d)=0$, we omit $l(d)$ in Eqs.(4) and (5). We will show how to set the values of f_{ud} and f_{yd} later, in order to penalize the long path of a random walk.

Maximizing $P(\{\boldsymbol{l}(\boldsymbol{u})\},\{\boldsymbol{l}(\boldsymbol{y})\}|S)$ is the same as maximizing $P(\{\boldsymbol{l}(\boldsymbol{u})\},\{\boldsymbol{l}(\boldsymbol{y})\},S)$. Furthermore, maximizing the joint likelihood $P(\{\boldsymbol{l}(\boldsymbol{u})\},\{\boldsymbol{l}(\boldsymbol{y})\},S)$ is the same as minimizing the energy:

$$\begin{aligned} E &= -\log P(\{\boldsymbol{l}(\boldsymbol{u})\},\{\boldsymbol{l}(\boldsymbol{y})\},S) \\ &\propto \boldsymbol{l}^{\mathrm{T}}(\boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{W}\boldsymbol{D}^{-1/2})\boldsymbol{l} + \boldsymbol{l}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{l}, \end{aligned} \quad (6)$$

where the probability vector $\boldsymbol{l}^{\mathrm{T}}=\{\boldsymbol{l}(\boldsymbol{u})^{\mathrm{T}}, \boldsymbol{l}(\boldsymbol{y})^{\mathrm{T}}\}$, $\boldsymbol{u}$ is a vector of all user nodes, $\boldsymbol{y}$ is a vector of all item nodes, and the diagonal matrix $\boldsymbol{F}=\mathrm{diag}(\boldsymbol{F}(U), \boldsymbol{F}(Y))$ corresponds to the edges connecting each user or item to the dummy node d. The diagonal entry of $\boldsymbol{F}(U)$ is $F(u)=n_u f_{ud}/d_u$; the diagonal entry of $\boldsymbol{F}(Y)$ is $F(y)=n_y f_{yd}/d_y$. $\boldsymbol{D}=\mathrm{diag}(\boldsymbol{D}_U, \boldsymbol{D}_Y)$ is the sum of weights of edges incident to each user and item node. The weight matrix $\boldsymbol{W}$ is based on the adjacency matrix $\boldsymbol{W}_{UY}$ of the user-item bipartite graph, and each entry w_{uy} corresponds to the rating score r_{uy} of item y by user u, $\boldsymbol{W}_{YU}=\boldsymbol{W}_{UY}^{\mathrm{T}}$. $\boldsymbol{W}$ is then written as

$$\boldsymbol{W} = \begin{pmatrix} \boldsymbol{0} & \boldsymbol{W}_{UY} \\ \boldsymbol{W}_{YU} & \boldsymbol{0} \end{pmatrix}.$$

Setting the derivative of Eq.(6) with respect to $\boldsymbol{l}$ equal to zero results in the following iterative equation:

$$\boldsymbol{l} = (\boldsymbol{F} + \boldsymbol{I})^{-1}\boldsymbol{D}^{-1/2}\boldsymbol{W}\boldsymbol{D}^{-1/2}\boldsymbol{l},$$

where we set the weights of edges incident to the dummy node as follows: $f_{ud}=d_u\alpha/(n_u(1-\alpha))$, $f_{yd}=d_y\alpha/(n_y(1-\alpha))$, and then all the diagonal entries of $(\boldsymbol{F}+\boldsymbol{I})^{-1}$ possess the same value $1-\alpha$; hence,

$$\boldsymbol{l} = (1-\alpha)\boldsymbol{D}^{-1/2}\boldsymbol{W}\boldsymbol{D}^{-1/2}\boldsymbol{l}. \quad (7)$$

Note that the iterative process defined by Eq.(7) converges and corresponds to an ARW (Doyle and Snell, 1984), since a labeled node in the set $A=\{u_{\mathrm{a}}\}$ whose probability value $l(u_{\mathrm{a}})=1$ is definite, corresponds to an absorbing state for an ARW. Let $B=\{\boldsymbol{u}/u_{\mathrm{a}}, \boldsymbol{y}\}$ denote the remaining unlabeled nodes, and then Eq.(7) can be rewritten as follows:

$$\begin{bmatrix} \boldsymbol{l}(A) \\ \boldsymbol{l}(B) \end{bmatrix} = (1-\alpha)\begin{bmatrix} (1/(1-\alpha))\boldsymbol{I}_A & \boldsymbol{0} \\ \boldsymbol{D}_B^{-1/2}\boldsymbol{W}_{BA}\boldsymbol{D}_A^{-1/2} & \boldsymbol{D}_B^{-1/2}\boldsymbol{W}_{BB}\boldsymbol{D}_B^{-1/2} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{l}(A) \\ \boldsymbol{l}(B) \end{bmatrix}.$$

It can be shown that our algorithm is

$$\boldsymbol{l}(B) = (1-\alpha)(\boldsymbol{D}_B^{-1/2}\boldsymbol{W}_{BA}\boldsymbol{D}_A^{-1/2}\boldsymbol{l}(A) + \boldsymbol{D}_B^{-1/2}\boldsymbol{W}_{BB}\boldsymbol{D}_B^{-1/2}\boldsymbol{l}(B)). \quad (8)$$

Iterating Eq.(8) leads to the unique configuration of probabilities of unlabeled nodes as follows:

$$\boldsymbol{l}(B)=(1-\alpha)(\boldsymbol{I}-(1-\alpha)\boldsymbol{D}_B^{-1/2}\boldsymbol{W}_{BB}\boldsymbol{D}_B^{-1/2})^{-1} \cdot \boldsymbol{D}_B^{-1/2}\boldsymbol{W}_{BA}\boldsymbol{D}_A^{-1/2}\boldsymbol{l}(A),$$

where given $0<\alpha<1$, $\boldsymbol{I}-(1-\alpha)\boldsymbol{D}_B{}^{-1/2}\boldsymbol{W}_{BB}\boldsymbol{D}_B{}^{-1/2}$ is invertible; hence the values of $\boldsymbol{l}(B)$ are harmonic and unique. The item entry in the unique solution $\boldsymbol{l}(B)$ satisfies $0\leq l(y)\leq 1$, because of the maximum principle of harmonic function (Doyle and Snell, 1984). We rewrite Eq.(7) in the perspective of a bipartite graph, resulting in the Eqs.(9) and (10) presented in Algorithm 1. Note that we clamp the labeled data $l(u_a)=1$ at each iteration step (the 6th line of Algorithm 1). When Algorithm 1 converges, the output $\boldsymbol{l}(\boldsymbol{y})$ is used to generate the ranking of items biased for the individual, and then the top-N items are delivered as the recommendations.

Algorithm 1 ARW on an augmented bipartite graph

```
Input: $S$, $\alpha$, $\boldsymbol{W}_{YU}$, $\boldsymbol{W}_{UY}$.
Output: $\boldsymbol{l}(\boldsymbol{y})$—probability vector for items $\boldsymbol{y}$.
1 $\boldsymbol{l}(\boldsymbol{u})=\{0,\cdots,0,\cdots,0\}^T$;
2 if ($<u_a, c_a>\in S$), $l(u_a)=1$;
3 repeat;
4     $\boldsymbol{l}(\boldsymbol{y})\leftarrow(1-\alpha)\boldsymbol{D}_Y^{-1/2}\boldsymbol{W}_{YU}\boldsymbol{D}_U^{-1/2}\boldsymbol{l}(\boldsymbol{u})$;    (9)
5     $\boldsymbol{l}(\boldsymbol{u})\leftarrow(1-\alpha)\boldsymbol{D}_U^{-1/2}\boldsymbol{W}_{UY}\boldsymbol{D}_Y^{-1/2}\boldsymbol{l}(\boldsymbol{y})$;    (10)
6     if ($<u_a, c_a>\in S$), $l(u_a)=1$;
7 until convergence.
```

EXPERIMENTS

We conducted several experiments to examine the performance of the proposed ARW algorithm, and address the following questions in particular:

(1) How does ARW compare with other random walk based models and the traditional collaborative filtering approach? For this question we compare our approach (ARW) with the ItemRank (Gori and Pucci, 2007) algorithm on the item correlation-graph, the random walk algorithm (Cheng *et al.*, 2007) on bipartite graph, and the item-based collaborative filtering algorithm (Sarwar *et al.*, 2001).

(2) How does the parameter α affect the ranking performance? Parameter α controls how much the ARW algorithm penalizes a long random walk on the bipartite graph. We vary the value of α from 0.1 to 0.9 to observe the differences in ranking performance.

Experimental setup

We conducted experiments on two real data sets: MovieLens and MovieRating. The MovieLens data set (Sarwar *et al.*, 2001) contains 100 000 ratings for 1682 items by 943 users, where each user has rated at least 20 movies. The MovieRating data set contains 43 865 ratings for 1000 items by 500 users (http://www.cs.usyd.edu.au/~irena/movie_data.zip). Each vote in the two data sets is in essence a triple $t_{uy}=(u, y, r_{uy})$, where $u\in U$ is a user, $y\in Y$ is a movie, and r_{uy} is an integer score between 1 (bad movie) and 5 (good movie).

For each data set, we split it into five disjoint subsets. Each subset is used as the test set and the remaining four subsets are merged as the training set. For each splitting, we refer to the set of triples used for training and testing as L and T, respectively. Moreover, we refer to L_u as the set of votes by user u in the training set and denote by T_u the votes by user u in the test set. Formally, $L_u=\{t_{u'y}\in L: u'=u\}$ and $T_u=\{t_{u'y}\in T: u'=u\}$. By performing 5-fold cross validation runs, we compared ranking performances of five scoring algorithms. Experimental results are reported and discussed in the later section.

In order to further demonstrate the advantage of our approach, we obtain precision and recall values with respect to various top-N settings, by applying scoring algorithms to the 0.8/0.2 training/test data set. In order to establish statistical significance of the findings, we have repeated the data set splitting procedure 10 times with different random seeds. The reported numbers are the mean performance averaged over these 10 runs. In the end we report and discuss the parameter sensitivity of the ARW algorithm for the MovieLens data set.

Evaluation metrics

To evaluate the performance of scoring algorithms, we used two metrics: (1) degree of agreement (DOA) used by Gori and Pucci (2007); (2) precision and recall often used in the field of information retrieval.

1. MacroDOA

DOA is a way of measuring how well an items ranking is for a given user. To compute the DOA for a single user u, we split the movie set Y into three subsets for the user u: L_u, T_u and NW_u. L_u is the set of movies that have been rated by user u in the training set. T_u is the set of movies that have been rated by user u in the test set. NW_u is the set of movies that have not been rated by user u, formally $NW_u=Y\backslash(L_u\cup T_u)$. Furthermore, we define the Boolean function $check\_order_u$ as

$$check\_order_u(m_j,m_k)=\begin{cases}1, & \text{if } IR_u^{m_j} > IR_u^{m_k},\\ 0, & \text{otherwise,}\end{cases}$$

where $IR_u^{m_j}$ is the score assigned to movie m_j by the examined algorithms with respect to user u. Then the individual DOA for user u is:

$$DOA_u=\frac{\sum_{(m_j\in T_u,m_k\in NW_u)} check\_order_u(m_j,m_k)}{|T_u|\times|NW_u|}.$$

So the DOA_u measures for user u the percentage of movie pairs ranked in the correct order with respect to the total number of pairs. A good scoring algorithm should rank the movies that have indeed been watched in higher positions than movies that have not been watched. A random ranking produces a DOA of 50% while an ideal ranking corresponds to a DOA of 100%. The global macro-averaged degree of agreement can be computed by averaging the individual DOA for each user:

$$MacroDOA=\sum_{u\in U}DOA_u\Big/|U|.$$

2. Precision and recall

For the top-N recommendation task, let Rel denote the number of relevant recommended items in the top-N recommendation list. In the later experiments, items with high scores (r_{uy}>3) are considered relevant. Precision is then defined as the ratio of Rel to N. Recall is defined as the ratio of Rel to the total number of relevant items for the test user.

Scoring algorithms

We still use ARW to denote our absorbing random walk model. We compared the performance of ARW with the other four scoring algorithms: Maximum-frequency (MaxF), Item-based CF algorithm using adjusted cosine similarities (IACOS), ItemRank, and BGRank.

1. MaxF

The maximum frequency algorithm simply ranks the movies by the number of persons who have rated them. In other words, movies are suggested to each person in order of decreasing popularity. MaxF acts as the baseline for appreciating the quality of other algorithms.

2. IACOS

The IACOS serves as the reference for appreciating the quality of random walk algorithms. The similarity between items i and j (Sarwar *et al.*, 2001) is written as

$$ACOS_{ij}=\frac{\sum_{u\in U_{ij}}(v_{ui}-\bar{v}_u)(v_{uj}-\bar{v}_u)}{\sqrt{\sum_{u\in U_{ij}}(v_{ui}-\bar{v}_u)^2}\sqrt{\sum_{u\in U_{ij}}(v_{uj}-\bar{v}_u)^2}},$$

where v_{ui} is the rating of item i by user u, $\bar{v}_u$ is the mean value of all ratings given by user u in the training set, U_{ij} is the set of users who co-rated items i and j. After computing adjusted cosine similarities among items, we further exploit the item-based aggregation function to predict the possible ratings of items unrated previously by the individual, which is defined as

$$v_{ui}=\frac{\sum_{j\in N_i}ACOS_{ij}\times v_{uj}}{\sum_{j\in N_i}|ACOS_{ij}|},$$

where N_i denotes the neighborhood items of item i, whose size $K=|N_i|$ is the key parameter to affect the efficiency and accuracy of the algorithm; the denominator is the sum of the absolute value of $ACOS_{ij}$, which performs better in practice than the sum of $ACOS_{ij}$.

3. ItemRank

Gori and Pucci (2007) proposed an ItemRank algorithm for recommendation tasks, which apply random walks to the correlation graph of items generated by one-mode projection from the user-item bipartite graph. The correlation graph just consists of item nodes, whose entry between any two items is the number of users who co-rated them. Formally,

$$U_{ij}=\begin{cases}\{u:(r_{ui}\in L_u)\wedge(r_{uj}\in L_u)\}, \text{ if } i\neq j,\\ \varnothing, \text{ if } i=j,\end{cases}$$
$$\tilde{C}_{ij}=|U_{ij}|,$$

where L_u is the set of ratings given by user u, r_{ui} and r_{uj} denote the ratings of items y_i and y_j given by the user u, respectively, and $|U_{ij}|$ denotes the number of users in U_{ij}, which is the set of users who co-rated the items y_i and y_j. $\tilde{\boldsymbol{C}}$ is therefore the adjacency matrix for the correlation graph. Then the ItemRank algorithm is written as follows:

$$IR_u=(1-\alpha)col\_norm(\tilde{\boldsymbol{C}})IR_u+\alpha d_u, \tag{11}$$

where d_u has been built according to scores of items rated by user u in the training set, $col\_norm(\tilde{\boldsymbol{C}})$ transforms the adjacency matrix $\tilde{\boldsymbol{C}}$ into a column stochastic matrix. Next, with probability α, the random surfer will visit the initial item nodes, or with probability $1-\alpha$, transit to one of its neighboring nodes in the graph.

4. BGRank

Cheng *et al.*(2007) presented several recommendation algorithms that perform a query-dependent random walk on a k-partite graph. Hence we implement its simple version as a reference, which performs a random walk on a user-item bipartite graph. The update rules for users and items are as follows:

$$\boldsymbol{p}(\boldsymbol{y})=(1-\alpha)col\_norm(\boldsymbol{W}_{YU})\boldsymbol{p}(\boldsymbol{u}), \tag{12}$$
$$\boldsymbol{p}(\boldsymbol{u})=(1-\alpha)col\_norm(\boldsymbol{W}_{YU}^{\mathrm{T}})\boldsymbol{p}(\boldsymbol{y})+\alpha\boldsymbol{q}, \tag{13}$$

where $\boldsymbol{p}(\boldsymbol{u})$ and $\boldsymbol{p}(\boldsymbol{y})$ denote the probabilities that a random surfer arrives at each user and item node, $\boldsymbol{q}$ corresponds to an initial vector whose elements are all zeros, except for one unit element corresponding to the active user. Next, probability α has a similar interpretation as in ItemRank.

Comparison of algorithms

In order to examine the performance of the ARW algorithm, we compare the ARW algorithm with other scoring algorithms by performing 5-fold cross validation runs. Since the PageRank-like algorithms often set α=0.15, we set α=0.15 for all methods to fairly compare the examined algorithms in the 5-fold cross validation experiments. The number of the nearest neighbors in IACOS was set to 100.

For the MovieLens data set, Table 1 summarizes MacroDOA results for five scoring algorithms by performing 5-fold cross validation runs. The best results are obtained by the ARW algorithm. The differences between the results of the ARW and other algorithms are statistically significant, with p<0.001 in paired t-tests. The ARW algorithm outperforms BGRank in an absolute improvement of 1.76%. The ARW algorithm obtains a relative gain of about 6.3% in MacroDOA over MaxF. The ARW algorithm also improves relatively by 3% against ItemRank. The standard deviation of the results (STD) across the 5-cross-validation runs is also reported in Table 1. The STD values of the ARW algorithm for two data sets are smaller than those of the ItemRank and IACOS algorithms, i.e., the ARW can achieve a more stable ranking performance than the ItemRank and IACOS algorithms. For the MovieRating data set, performing 5-fold cross validation runs for scoring algorithms achieves the analogous results for MacroDOA, which are presented in Table 2.

The ARW and BGRank algorithms outperform the item-based CF algorithm in that random walk models consider the relationships between users and items from a global perspective, as opposed to the local pair-wise similarity methods used by the item-based methods. The ARW and BGRank algorithms both outperform the ItemRank algorithm. The reason is that transforming a bipartite graph into a correlation

Table 1 MacroDOA values of 5-fold cross validation runs of algorithms on the MovieLens data set

No	MacroDOA values (%)				
	ARW	BGRank	ItemRank	IACOS	MaxF
1	89.85	88.15	87.73	87.71	84.57
2	90.04	88.30	87.61	88.06	84.74
3	90.30	88.51	87.69	88.34	84.82
4	90.16	88.35	87.47	88.12	84.77
5	90.02	88.23	88.28	87.91	84.65
AVG.	90.07	88.31	87.76	88.02	84.71
STD	0.15	0.12	0.28	0.21	0.09

Table 2 MacroDOA values of 5-fold cross validation runs of algorithms on the MovieRating data set

No	MacroDOA values (%)				
	ARW	BGRank	ItemRank	IACOS	MaxF
1	84.51	81.88	78.35	80.86	77.33
2	84.93	82.41	78.78	81.52	77.63
3	84.46	81.95	78.19	81.20	77.17
4	84.33	81.64	78.18	80.96	77.13
5	84.85	82.43	78.82	81.39	77.76
AVG.	84.62	82.06	78.46	81.19	77.40
STD	0.23	0.31	0.28	0.25	0.25

graph results in the following four types of information loss: (1) ignoring users who just rated one item; (2) ignoring the saturated effect on items co-rated by a large number of users; (3) treating indifferently user nodes of different degrees; (4) ignoring rating scores. The shortcomings of one-mode projection are overcome by the ARW and BGRank. The ARW algorithm performs better than the BGRank because that ARW takes into account the degree of each node in a bipartite graph and then prevents the noise induced by the popular items or users with many votes. For example, a popular item is usually given a high score by many users, while many of them are of no common interest.

For the MovieLens data set, Fig.2 shows the plots of precision and recall for examined algorithms with respect to different values of top-N. We first randomly split the whole data set into the training/test data set in a ratio of 0.8/0.2 ten times, and then all the algorithms with the same setting α=0.15 were applied to each splitting. Finally, the mean precision/recall values averaged over 10 runs are plotted in Fig.2. At N=10, ARW achieves a precision of 29.56%, compared to 26.14% by BGRank and 21.36% by MaxF, i.e., there is a much higher probability that ARW will choose the high-rated items within the top-10 recommendations than MaxF. Moreover, the ARW algorithm performs the best in terms of both precision and recall at each choice of N.

For the MovieRating data set, analogous plots of precision and recall of the examined algorithms versus top-N are drawn in Fig.3. As expected, the precision of each algorithm decreases as N increases, and the recall of each algorithm increases as N increases.

Impact of the parameter α

To evaluate the impact of the parameter α, we first randomly split the whole MovieLens data set in a ratio of 0.8/0.2 ten times, and then applied the ARW, BGRank with various values of parameter α to each splitting. Finally, the mean MacroDOA values averaged over 10 runs were used to measure the quality of those generated rankings. Fig.4 shows the plots of mean MacroDOA scores of ARW and BGRank with respect to α from 0.1 to 0.9 by a step of 0.1. ARW achieves about a 2% improvement in MacroDOA over BGRank at various choices of α. On the whole, the larger the value of α, the more locally the ARW and BGRank algorithms behave, and the better the ranking performance of the two algorithms. This fact indicates that it is not necessary to propagate

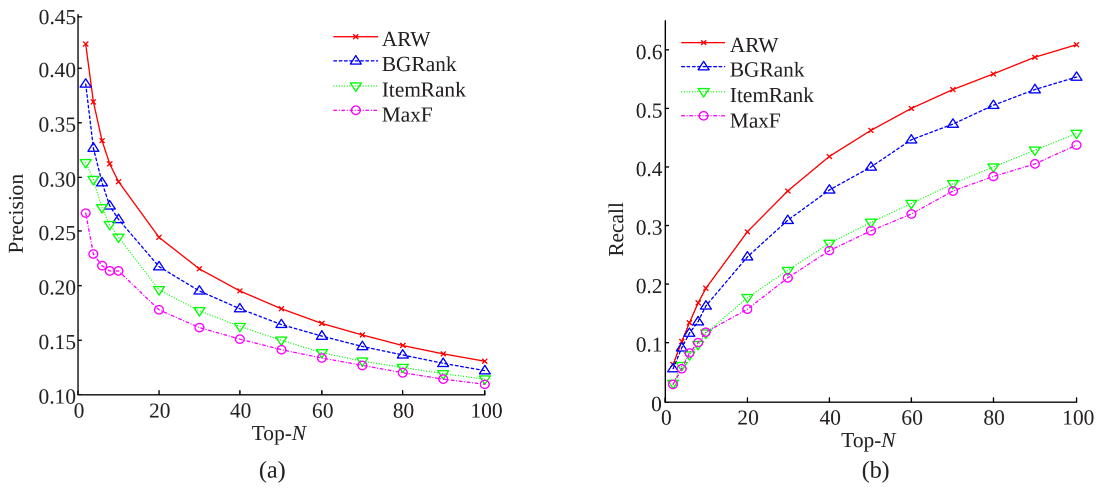

Fig.2 Comparison of algorithms in terms of (a) precision and (b) recall for the MovieLens data set

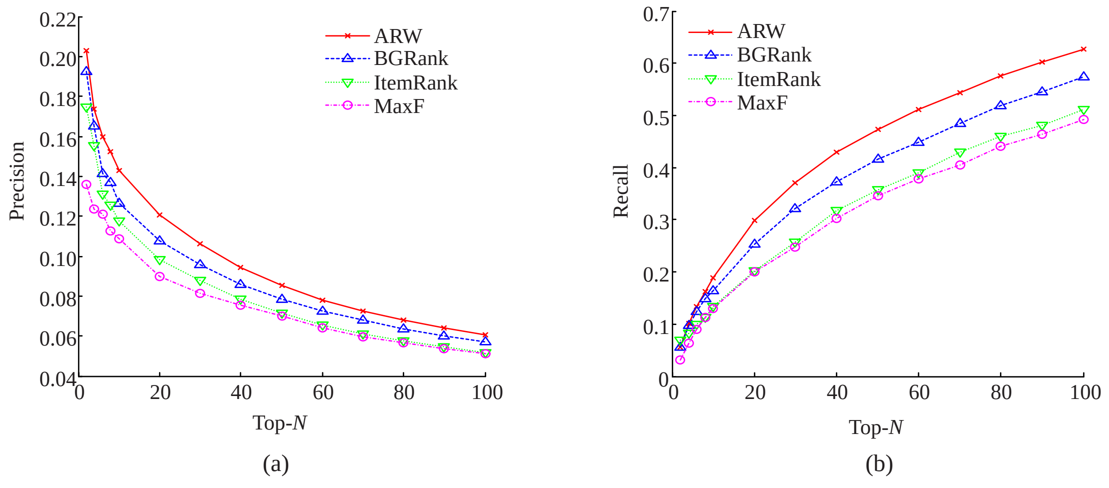

Fig.3 Comparison of algorithms in terms of (a) precision and (b) recall for the MovieRating data set

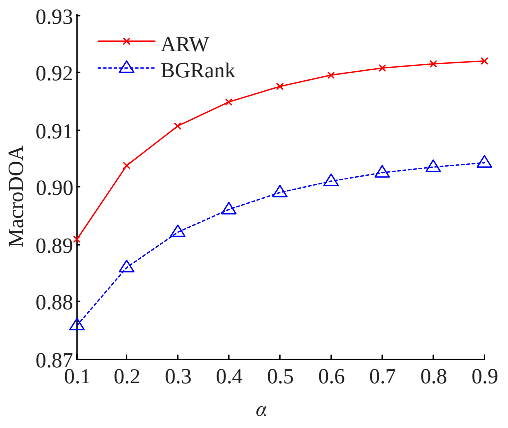

Fig.4 MacroDOA scores of ARW/BGRank at various choices of *α* for the MovieLens data set

too much, as a remote random walk tends to introduce more noise than better covering.

CONCLUSION

We have employed a Gaussian field to model the top-*N* recommendation task as a semi-supervised classification problem, from which an ARW algorithm is inferred. Experimental results on two real data sets show that our approach significantly outperforms the state-of-the-art random walk-based algorithms and the traditional item-based collaborative filtering method for the top-*N* recommendation task. In the future we will employ metric learning techniques to learn the intrinsic distance between users and items, and then apply a random walk based algorithm to the learnt metric distances instead of heuristic rating scores of items by the individual. Moreover, enhancing our models to deal with negative ratings is another interesting research direction.

References

Baluja, S., Seth, R., Sivakumar, D., Jing, Y., Yagnik, J., Kumar, S., Ravichandran, D., Aly, M., 2008. Video Suggestion and Discovery for YouTube: Taking Random Walks through the View Graph. Proc. 17th Int. Conf. on WWW, Beijing, China, p.895-904. [doi:10.1145/1367497.1367618]

Brand, M., 2005. A Random Walks Perspective on Maximizing Satisfaction and Profit. Proc. SDM, p.12-19.

Breese, J.S., Heckerman, D., Kadie, C.M., 1998. Empirical Analysis of Predictive Algorithms for Collaborative Filtering. Proc. 14th Conf. on Uncertainty in Artificial Intelligence, University of Wisconsin Business School, Madison, Wisconsin, USA, p.43-52.

Cheng, H., Tan, P.N., Sticklen, J., Punch, W.F., 2007. Recommendation via Query Centered Random Walk on K-partite Graph. Proc. 7th IEEE Int. Conf. on Data Mining, Omaha, Nebraska, USA, p.457-462. [doi:10.1109/ICDM.2007.8]

Doyle, P., Snell, L., 1984. Random Walks and Electrical Networks. Mathematical Association of America.

Fouss, F., Pirotte, A., Renders, J.M., Saerens, M., 2007. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Trans. Knowl. Data Eng.*, **19**(3):355-369. [doi:10.1109/TKDE.2007.46]

Fuxman, A., Tsaparas, P., Achan, K., Agrawal, R., 2008. Using the Wisdom of the Crowds for Keyword Generation. Proc. 17th Int. Conf on WWW, Beijing, China, p.61-70. [doi:10.1145/1367497.1367506]

George, T., Merugu, S., 2005. A Scalable Collaborative Filtering Framework Based on Co-clustering. Proc. 5th IEEE Int. Conf. on Data Mining, Houston, Texas, USA, p.625-628. [doi:10.1109/ICDM.2005.14]

Gori, M., Pucci, A., 2007. ItemRank: A Random-walk Based Scoring Algorithm for Recommender Engines. Proc. 20th Int. Joint Conf. on Artificial Intelligence, Hyderabad, India, p.2766-2771.

Huang, Z., Chen, H., Zeng, D.D., 2004. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Trans. Inf. Syst.*, **22**(1):116-142. [doi:10.1145/963770.963775]

Kunegis, J., Schmidt, S., 2007. Collaborative Filtering using Electrical Resistance Network Models. 7th Industrial Conf. on Data Mining, Leipzig, Germany, p.269-282. [doi:10.1007/978-3-540-73435-2_21]

Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J., 2001. Item-based Collaborative Filtering Recommendation Algorithms. Proc. 10th Int. Conf. on WWW, Hong Kong, China, p.285-295. [doi:10.1145/371920.372071]

Singh, A.P., Gunawardana, A., Meek, C., Surendran, A.C., 2007. Recommendations Using Absorbing Random Walks. North East Student Colloquium on Artificial Intelligence (NESCAI).

Symeonidis, P., Nanopoulos, A., Papadopoulos, A.N., Manolopoulos, Y., 2008. Nearest-biclusters collaborative filtering based on constant and coherent values. *Inf. Retr.*, **11**(1):51-75. [doi:10.1007/s10791-007-9038-4]

Wang, J., Vries, A., Reinders, M., 2006. Unifying User-based and Item-based Collaborative Filtering Approaches by Similarity Fusion. Proc. 29th ACM SIGIR Conf., p.501-508. [doi:10.1145/1148170.1148257]

Yildirim, H., Krishnamoorthy, M.S., 2008. A Random Walk Method for Alleviating the Sparsity Problem in Collaborative Filtering. Proc. 2008 ACM Conf. on Recommender Systems, Lausanne, Switzerland, p.131-138. [doi:10.1145/1454008.1454031]

Zhu, X., Ghahramani, Z., Lafferty, J., 2003. Semi-supervised Learning Using Gaussian Fields and Harmonic Functions. Proc. 20th Int. Conf. on Machine Learning, Washington DC, USA, p.912-919.