Efficient Bayesian Methods for Graph-based Recommendation

Ramon Lopes[1,2]
rlopes@ufrb.edu.br

Renato Assunção[2]
assuncao@dcc.ufmg.br

Rodrygo L. T. Santos[2]
rodrygo@dcc.ufmg.br

[1]Exact Sciences and Technology Center
Universidade Federal do Recôncavo da Bahia
Cruz das Almas, BA, Brazil

[2]Department of Computer Science
Universidade Federal de Minas Gerais
Belo Horizonte, MG, Brazil

ABSTRACT

Short-length random walks on the bipartite user-item graph have recently been shown to provide accurate and diverse recommendations. Nonetheless, these approaches suffer from severe time and space requirements, which can be alleviated via random walk sampling, at the cost of reduced recommendation quality. In addition, these approaches ignore users' ratings, which further limits their expressiveness. In this paper, we introduce a computationally efficient graph-based approach for collaborative filtering based on short-path enumeration. Moreover, we propose three scoring functions based on the Bayesian paradigm that effectively exploit distributional aspects of the users' ratings. We experiment with seven publicly available datasets against state-of-the-art graph-based and matrix factorization approaches. Our empirical results demonstrate the effectiveness of the proposed approach, with significant improvements in most settings. Furthermore, analytical results demonstrate its efficiency compared to other graph-based approaches.

CCS Concepts

•**Information systems** → **Recommender systems;** *Top-k retrieval in databases;*

Keywords

collaborative filtering; item recommendation; Bayesian statistics

1. INTRODUCTION

The main objective of Recommender Systems (RS) is to guide users in a personalized way to interesting products to maximize user satisfaction and improve merchant revenue. These systems have become largely utilized in multiple business niches; the most relevant applications are recommendation of products in online shopping websites like Amazon, movies in video portals like YouTube and friends in social networks like Facebook. RS fundamentally take one of two approaches, namely, Collaborative Filtering (CF) or Content-based Filtering, or show a combination of them. Loosely speaking, Collaborative Filtering uses the preferences of similar users to select items, and Content-based Filtering uses features of items to find similar content.

The representation of users and the choice of a measure of similarity between them are two fundamental questions in the context of CF systems. In memory-based CF approaches, n users are represented as vectors embedded in an m-dimensional vector space where each dimension corresponds to an item, and measures of similarity are often based on operations defined over this vector space. Thus, data are represented as an $n \times m$ user-item matrix where rows correspond to users, columns to items and each entry usually represents a rating. On the one, hand such approaches provide a simple representation, but, on the other hand, it is computationally expensive to compute similarities between all pairs of users as the dataset size increases.

As an attempt to overcome the computational expense previously pointed out, the user-item matrix can be regarded as an adjacency matrix of a user-item undirected bipartite graph, giving rise to so-called graph-based approaches. As a result, measures of similarity between users are based on graph statistics such as commute or hitting time between nodes [4]. For instance, users that possess the same taste will be connected by a large number of short paths [4].

Page et al. [15] propose to treat an entity graph as a Markov chain whose long-term stationary distribution can be used as a global ranking scoring. There exists several works in the literature [1, 3, 4, 5, 9, 20] that make use of random walks in the context of RS. To this end, transition probabilities are defined so that they exploit the user-item graph structure. For instance, the transition probability between two items is proportional to the number of users that rated both items [9]. The authors in [1, 3] take a step further and show that it is possible to improve recommendation quality by obtaining the distribution within three or five steps instead of incurring the computational burden of obtaining the long-term stationary distribution as in [4, 5, 9, 20].

Due to the size of the transition matrix, the methods presented in [1, 3, 4, 5, 9, 20] suffer from memory limitation as the dataset size increases. In addition, these methods are computationally intensive as they require matrix multiplications. To overcome these issues, Christoffel et al. [1] and Cooper et al. [3] propose to approximate the final distribution by a sampling process. The proposed methods are applicable to medium or large size datasets, but recommen-

RecSys '16, September 15–19, 2016, Boston , MA, USA

DOI: http://dx.doi.org/10.1145/2959100.2959132

dation quality depends on the sample size adopted in the sampling process. Thus, we claim that there is a lack of less resource demanding methods in the context of graph-based approaches that do not degrade the quality of recommendations.

As a concrete example, given two items where one has 80 positive out of 100 ratings while the other has 20 positive out of 20 ratings, which item is likely to maximize user satisfaction in face of this information? In this scenario, we believe that the set of ratings given by users to items contains a valuable source of latent information that should be taken into account for recommendation. Thus, in this work we hypothesize that graph-based RS can benefit from the exploitation of the set of ratings for recommendation and then improving user satisfaction. To the best of our knowledge, such hypothesis has not been investigated in the context of graph-based RS. Furthermore, the current state-of-the-art graph-based approaches do not extract any further information from the set of ratings to leverage recommendation. To fulfill this gap, we propose a CF method that not only relies on the user-item bipartite graph, but it also takes advantage of ratings given by the users to make recommendations.

In this work, we propose three scoring functions based on the Bayesian paradigm that are at the core of our recommendation approach. Similar to the works presented in [1, 3], we exploit three-step paths in the user-item bipartite graph starting from the target user with the advantage that we resort neither to matrix multiplications nor sampling processes. Instead, we propose the enumeration of all such paths hence overcoming the computational burden incurred by the allocation and multiplication of transition matrices. We analytically show that the enumeration process is more efficient than the approaches proposed in [1, 3].

We carried out experiments on seven datasets freely available. The results show that our method clearly outperforms the approaches proposed in [1, 3] in all metrics used for comparison. It is common sense in the RS literature that matrix factorization approaches are the state-of-the-art methods for item recommendation in explicit feedback domains. Thus, we compare our method against matrix-factorization approaches, and results show that our method is able to improve their results in some datasets. The contributions of this paper are three-fold: a) we propose a more computationally efficient graph-based approach than those presented in [1, 3]; b) we propose three Bayesian scoring functions that take advantage of the set of ratings for recommendation; c) we present a comprehensive empirical evaluation of our method against state-of-the-art RS approaches across several datasets.

2. RELATED WORKS

Fouss et al. [4, 5] propose a CF approach that relies upon random walks over user-item bipartite graph. They consider four similarity measures, namely, average first passage time, average commute time, Euclidean commute time distance and pseudo inverse of the Laplacian matrix C^+, where $C = B - A$, A represents the adjacency matrix, B is a diagonal matrix where each entry represents vertex degrees associated to the user-item bipartite graph. Computational results show C^+ provided the best results for a movie dataset.

ItemRank [9] is a random walk based scoring algorithm for recommendation. It makes use of an item correlation graph where edge weights represent item correlations. The correlation between two items amounts to the number of users that rated both items. In turn, the transition matrix is given by the normalized correlation matrix of items. Experiments show that ItemRank performs better than the methods proposed by Fouss et al. [4, 5].

Singh et al. [20] propose an approach to CF that combines social relationships and ownership data to make recommendations. To this end, they model user-item relations as a bipartite graph and augment it with user-user social links. The algorithm is based on random walks with absorbing states that induces a distribution per user over all items. A walker begins from a target user from where it may transition to a friend or to an item. Once the walker reaches an item, he cannot be transitioned out of it. The authors evaluate the proposed method using data from an online game service to suggest items the user might buy, and from text corpus to suggest words to papers.

Cooper et al. [3] propose three scoring algorithms called P^3, P^5 and P^3_α, which are based on random walks on the graph representing associations between users and items. Let $G = (U \cup I, E)$ be an undirected bipartite graph of users and items, where U is the set of users, I is the set of items and there exists an edge between user u and item i if u rated i. Define the transition matrix $P = B^{-1}A$, where A is the adjacency matrix associated to G and B is a diagonal matrix where each entry equals the degree of the corresponding vertex. P^3 and P^5 are based on the distribution of random walks of three and five steps, respectively, starting from the target user vertex. In turn, P^3_α generalizes P^3 in the sense that its transition matrix is raised to the power of α. Experiments show that P^3_α provides better results than P^3, which in turn provides better results than P^5. Due to the memory burden of the proposed methods, the authors resort to estimating the distribution per user over all items via random walk sampling. They show that direct simulations of random walks for P^3 and P^5 are more memory efficient when compared to methods based on matrix calculations, so random walking sampling can be applied to larger datasets at the cost of recommendation quality. As a conclusion, the authors show P^3 and P^3_α provide better results than the methods proposed in [4, 5, 9].

Christoffel et al. [1] introduce an algorithm called RP^3_β which is based on P^3 to optimize accuracy and diversity. RP^3_β compensates for the influence of popular items by taking into account item popularity in the ranking given by P^3. Let P^3_{ui} be the original score of item i for target user u as the outcome of P^3. RP^3_β re-weights the score with $\tilde{P}^3_{ui} = P^3_{ui}/b^\beta_{ii}$, where b_{ii} represents the degree of vertex i and $\beta \in \mathbb{R}_+$. Experiments show that RP^3_β increases accuracy and diversity when compared to P^3. Similarly to Cooper et al. [3], due to memory limitations, the authors resort to a sampling approximation of the proposed method where distributions per user over all items are estimated by using 5 million random walks.

A naive implementation of the methods proposed in [1, 3] results in time complexity $\Theta((|I| + |U|)^3)$ and space complexity $\Theta((|I| + |U|)^2)$. Thus, the authors in [3] propose an approach that split the matrices in blocks and compute P^3 by a sequence of multiplications and additions of these blocks which results in time complexity $\Theta(|I|^2 \times |U|^2)$ and space complexity $\Theta(|I| \times |U|)$.

As we move to medium/large size datasets, the methods discussed here have computational limitations since the size

of the transition matrix may become too large to fit in memory. As an alternative, Cooper et al. [3] propose approximating transition probabilities by sampling random walks of a fixed length. However, random walk sampling may require a large number of simulations to converge. Thus, there is a lack of a method that provides exact calculations and decreases memory burden without degrading recommendation quality. To fulfil this lack, we also propose a strategy to compute P^3, P^5, P^3_α and RP^3_β without resorting to matrix calculations, so bypassing the memory burden due to matrix allocation.

The methods discussed in this section do not take into account for recommendation any latent information present on the set of ratings. In fact, these graph-based approaches solely make use of the set of user-item pairs to define the structure of the user-item bipartite graph. Thus, the rating system is completely discarded in these approaches. For instance, in these methods, there is no distinction between a scenario where a user gives the maximum possible rating for an item and a scenario where a different user gives the minimum possible rating for the same item.

3. EFFICIENT BAYESIAN SCORING

The Bayesian paradigm provides a principled way of incorporating prior information to data and, then, allows the assignment of non-zero probabilities to unseen events. These probabilities represent our beliefs in those events. For instance, the Bayesian paradigm naturally allows us to answer questions like which item should we buy where one item has 2 up-votes out of 2 ratings while the other has 35 up-votes out of 40 ratings. Thus, the scoring functions we propose relies on the Bayesian paradigm to leverage recommendation.

3.1 Model

Let U be the set of users and I be the set of items. We define the set of evaluations $D = \{(u, i, r_{ui}) | u \in U, i \in I, r_{ui} \in R\}$, where R represents the set of possible rating values. Thus, $(u, i, r_{ui}) \in D$ represents the evaluation made by user u where item i received rating r_{ui}. We assume that $R = \{0, 1\}$, where, given $u \in U, i \in I$, $r_{ui} = 1$ representing a positive assessment while $r_{ui} = 0$ representing a negative assessment. Such assumption is not unrealistic, for instance YouTube adopts a thumbs up/down rating scale so that users can provide explicit feedback. Also, according to the media Netflix[1] and Uber[2] may replace their 5-star rating system.

Given $u \in U$, $I_u = \{i \in I | (u, i, r_{ui}) \in D\}$ represents the set of items evaluated by user u. Let $U_i = \{v \in U | (v, i, r_{vi}) \in D\}$ be the set of users who evaluated item i. Similarly, let $R_j = \{r_{vj} \in R | (v, j, r_{vj}) \in D\}$ be the set of ratings given to item j. Thus, we define the set $R^+_j = \{r_{vj} \in R_j | r_{vj} = 1\}$ of positive assessments and the set $R^-_j = \{r_{vj} \in R_j | r_{vj} = 0\}$ of negative assessments received by item j.

Let $G = (U \cup I, E)$ be an undirected bipartite graph, where $E = \{(u, i) | (u, i, r_{ui}) \in D\}$, that is, there exists an edge between user u and item i if u rated i. We define $\delta(x) = \{i | (x, i) \in E\}$ as the neigborhood of a vertex x, $\Delta_U = \max_{u \in U} |\delta(u)|$ as the maximum vertex degree in U and Δ_I is defined similarly for I, $\Gamma_U = \frac{\sum_{u \in U} |\delta(u)|}{|U|}$ as the average degree of vertices in U and Γ_I is defined similarly for I. Given $u \in U$, we define $\mathcal{P}_u = \{\langle u, v, w, x\rangle | v \in I_u, w \in U, x \in I, (u, v), (v, w), (w, x) \in E\}$ as the set of all three-step paths starting from vertex $u \in U$. Let $\mathcal{P} = \cup_{u \in U} \mathcal{P}_u$ be the set of all three-step paths in the graph starting from users.

Given $j \in I$, let Y_j be a binary random variable that assumes 1 if j receives a positive assessment and 0 otherwise, where $\mathbb{P}(Y_j = 1) = \theta_j$. We place a Beta distribution as the prior of θ_j, where $\theta_j \sim \text{Beta}(\hat{a}, \hat{b})$. Intuitively, θ_j represents the unknown reliability of item j within the range $(0, 1)$. As $|R_j|$ increases, the Beta distribution shape tends to concentrate around its mean, then such notion of reliability turns out to be more precise. After observing R_j, we update our knowledge about θ_j, so that

$$\theta_j | R_j \sim \text{Beta}(a, b), \tag{1}$$

where $a = \hat{a} + |R^+_j|$ e $b = \hat{b} + |R^-_j|$.

The method we propose in this work makes a recommendation list for a target user $u \in U$ based on $\mathcal{P}_u$. To this end, we define a ranking function $f_u : I \to \mathbb{R}_+$ defined as $f_u(x) = \sum_{\langle u, v, w, x\rangle \in \mathcal{P}_u} s(\langle u, v, w, x\rangle)$, where $s : \mathcal{P} \to \mathbb{R}_+$ is a scoring function. Intuitively, $f_u(x)$ represents the sum of the weights of all three-step paths connecting u and x.

```
   input  : G = (U ∪ I, E), u ∈ U, scoring function s
   output: f_u
1  for v ∈ δ(u) do
2      for w ∈ δ(v) do
3          for x ∈ δ(w) do
4              f_u(x) := f_u(x) + s(⟨u, v, w, x⟩)
5          end
6      end
7  end
```

Algorithm 1: f_u computation for target user u.

Algorithm 1 illustrates the method we propose in this work. The algorithm receives as input the user-item undirected bipartite graph G, target user u and a scoring function s as previously defined. In lines 1-3, we systematically iterate over each neighbor of the corresponding vertex. Thus, we enumerate all three-step paths starting from u. In line 4 we compute the value function f_u for the final vertex x in the path $\langle u, v, w, x\rangle \in \mathcal{P}$. The algorithm outputs the ranking function f_u. In this work, we propose three scoring functions that are based on the Bayesian paradigm. These scoring functions are discussed next.

3.2 Scoring Functions

Posterior Inequality Scoring. We propose a scoring function dubbed Posterior Inequality Scoring (PIS) where

$$s(\langle u, v, w, x\rangle) = \begin{cases} \mathbb{P}(\theta_x > \theta_v | \mathcal{R}_x, \mathcal{R}_v), & \text{if } v \neq x \\ 0, & \text{otherwise} \end{cases}$$

given $\langle u, v, w, x\rangle \in \mathcal{P}_u$. We can obtain $\mathbb{P}(\theta_i > \theta_j | \mathcal{R}_i, \mathcal{R}_j)$ from (1). Intuitively, PIS represents the probability of the reliability of candidate item x being greater than the reliability of item v in the user history.

Posterior Prediction Scoring We propose a scoring

[1]http://www.businessinsider.com/netflix-wants-to-ditch-5-star-ratings-2016-1

[2]http://qz.com/574033/uber-may-replace-its-five-star-driver-rating-system-with-emoji/

function called Posterior Prediction Scoring (PPS) where

$$s(\langle u,v,w,x\rangle) = \begin{cases} \mathbb{P}(Y_v = 1|R_v) \times \mathbb{P}(Y_x = 1|R_x), & \text{if } v \neq x \\ 0, & \text{otherwise} \end{cases}$$

given $\langle u,v,w,x\rangle \in \mathcal{P}_u$. We can obtain $\mathbb{P}(Y_j = 1|R_j) = \frac{a+1}{a+b+1}$ from (1). Intuitively, PPS represents the probability of both v and x receiving positive assessments where we assume that Y_v and Y_x are independent.

Posterior Odds Ratio Scoring We propose a scoring function denominated Posterior Odds Ratio Scoring (PORS) where

$$s(\langle u,v,w,x\rangle) = \begin{cases} \frac{\psi_x}{\psi_v}, & \text{if } v \neq x \\ 0, & \text{otherwise} \end{cases}$$

given $\langle u,v,w,x\rangle \in \mathcal{P}_u$. We define $\psi_i = \frac{\mathbb{P}(Y_i=1|R_i)}{\mathbb{P}(Y_i=0|R_i)}$ for $i \in I$. Intuitively, PORS represents how large the odds of x receiving a positive assessment is when compared to the odds of v receiving a positive assessment.

The scoring functions we propose are global in the sense that they do not depend on the target user. However, the ranking function f_u is in fact customized for each user since the final result depends on the paths starting from vertex u.

Complexity Analysis. Our method incurs space complexity $\mathcal{O}(|U| \times \Delta_U)$ and time complexity $\mathcal{O}(|U| \times \Delta_I^2 \times \Delta_U)$ at worst case scenario considering the set of target users U. Thus, both space and time complexity of our method is lower than those presented in Section 2.

Implementation Details. We represent G as an adjacency list and obtain $\mathcal{P}_u$ by systematically enumerating according to G all three-step paths starting from vertex u. Thus, our implementation for P^3_α and RP^3_β makes use of such enumeration instead of relying on matrix calculations or sampling as proposed by Christoffel et al. [1] and Cooper et al. [3]. Cook [2] proposes an analytical approach to compute Beta inequalities. However, this approach is computationally expensive since it relies on recursive function calls. Thus, we resort to numerical integration methods to compute $\mathbb{P}(\theta_i > \theta_j|R_i, R_j)$. To this end, we make use of GNU Scientific Library [6] as the numerical integration library for PIS and memoization technique for storing results and, then, avoid recomputing such values. To avoid the occurrence of probabilities equal to zero in the proposed scoring functions, we make use of pseudo counts. For each item, we add one positive and one negative pseudo-assessment as if they both were given by a dummy user. The reason to add one for each end of the scale is to balance out the analysis so we do not induce bias.

4. EXPERIMENTAL SETUP

Our experiments are designed to answer the following questions: i) can we improve the accuracy of recommendations by taking into account the latent information present in the set of ratings by means of the scoring functions we propose in this work? ii) how do our proposed scoring methods compare to one another? To answer these questions we make use of several datasets and baselines in our experiments.

4.1 Datasets

The BookCrossing (BX as an abbreviation) dataset [21] contains data about book lovers that exchange books all around the world. The original dataset contains 278,858 anonymous users providing 1,149,780 ratings about 271,379 books spanning a period of time from August to September 2004. Ratings are given by users in a ten-star rating scale. In our experiments, we removed ratings related to implicit feedback.

The MovieLens datasets [10] are widely used in the Recommender Systems literature and comprise four datasets of increasing sizes. Each dataset consists of users' preferences for movies expressed in a five-star rating scale. In this work we consider the MovieLens 1M (ML 1M for short) dataset that comprises 1 million ratings from 6,000 users on 4,000 movies.

The Amazon dataset [14] contains 142.8 million product reviews spanning May 1996 - July 2014 and product metadata from Amazon. Ratings are given by users in a five-star rating scale. In this work, we select three representative product categories from that dataset, namely, Cds & Vynil (Cds as an abbreviation), Electronics and Kindle.

The FilmTrust dataset [8] contains 35,497 movie ratings crawled in June 2011 from 1,508 users on 2,071 movies. This dataset also provides ratings among users denoting how much a user trusts another in the social network. In turn, the Epinions dataset [13] contains 664,824 product ratings from 49,290 users on 139,738 products.

Since we do not focus on cold start issues, we keep users and items with at least 10 ratings. As a result, the data used for experiments differ from the original datasets and are summarized in Table 1. A common sense in the RS literature is that most users consume a small fraction of the items present in the catalogue and many items have few ratings [16]. One can see that except for MovieLens 1M dataset the maximum and average degree for items and users are significantly less than the corresponding total number in the dataset which turns out to favor our enumeration process.

4.2 Recommendation Baselines

We consider P^3_α, RP^3_β, Most Popular (MP), Bayesian Personalized Ranking Matrix Factorization (BPRMF) [17] and Weighted Regularized Matrix Factorization (WRMF) [11] as recommendation baselines in our investigations. MP sorts items according to their global popularity and so the recommendation list does not depend on the target user. In turn, WRMF and BPRMF are two representative matrix-factorization approaches for recommendation. WRMF extracts latent factors from implicit feedback and introduces regularization to prevent overfitting. BPRMF optimizes a maximum posterior estimator derived from a Bayesian analysis of the problem and learning relies on stochastic gradient descent with bootstrap sampling. In our experiments, we make use of the WRMF and BPRMF implementations provided in the Java port MyMediaLite [7] recommender system framework.[3]

4.3 Evaluation Methodology

The datasets are not compliant with the assumption $R = \{0, 1\}$. For instance, in the Amazon dataset $R = \{1, 2, 3, 4, 5\}$. Given a dataset and user u present on it, we calculate the average $\bar{r}_u$ of all ratings given by u. Assume that item i was graded $\hat{r}_{ui}$ by u. We define r_{ui} equals 1 if $\hat{r}_{ui} \geq \bar{r}_u$ or 0 otherwise. We replace $(u, i, \hat{r}_{ui})$ in D by (u, i, r_{ui}). As a result,

[3]https://github.com/jcnewell/MyMediaLiteJava

Table 1: Datasets properties.

DataSet	$\|D\|$	$\|U\|$	$\|I\|$	Δ_U	Δ_I	Γ_U	Γ_I
BookCrossing	42,137	1,842	2,065	964	225	22.87	20.40
Cds & Vynil	445,412	15,592	16,184	2,069	603	28.56	27.52
Electronics	347,393	20,247	11,589	317	1,376	17.16	29.98
Kindle	367,478	14,356	15,885	664	377	22.60	23.13
MovieLens 1M	998,539	6,040	3,260	2,233	3,428	165.32	306.30
FilmTrust	28,320	949	159	83	848	29.84	178.11
Epinions	300,304	10,706	8,945	525	1,491	28.05	33.57

we make a transformation of the original set of ratings into a dichotomized set where the user bias is taken into account.

We carried out experiments on the datasets described by using 5-fold cross validation. To this end, we randomly partitioned D into 5 subsets, where a subset T is retained for testing, and the remaining 4 subsets are used as training data. We repeated this process 5 times and report the average result across all the test folds in 5 trials.

Given $u \in U$, we define the set $T_u \subset I_u$ of items rated by u and used for testing. We partition $T_u = T_u^+ \cup T_u^-$, where $T_u^+ = \{i \in I_u | (u, i, r_{ui}) \in T, r_{ui} = 1\}$ is the set of items used for testing that u rated positively and T_u^-, in turn, is the set of items used for testing that u rated negatively. To express our lack of information about data we adopt an uninformative prior distribution by setting $\hat{a}$ and $\hat{b}$ to one so that Beta distribution reduces to Uniform distribution.

4.4 Evaluation Criteria

To answer the questions we raised, we make use of metrics largely adopted in the RS literature [18]. The metrics we report are computed by averaging over all the users in testing. Given $u \in U$, let L_u be the recommendation list ordered according to f_u. For a given $N \in \mathbb{N}$, let L_u^N be the recommendation list composed of the N highest scored items according to f_u. Given L_u, $L_u[i]$ represents the item at position i and the same applies for L_u^N. Finally, let A be the set of all $|I|$ length ordered lists composed of items in I where repetition is not allowed.

Precision. Given L_u^N, precision is defined as

$$P@N = \frac{|L_u^N \cap T_u^+|}{N} \tag{2}$$

Mean Average Precision (MAP). Given L_u, Mean Average Precision (MAP) is defined as

$$MAP = \sum_{N:=1}^{|L_u|} \frac{P@N \times \mathbf{1}_{T_u^+}(L_u[N])}{|T_u^+|} \tag{3}$$

where $\mathbf{1}_{T_u^+}$ is an indicator function.

Mean Reciprocal Rank (MRR). In the Information Retrieval literature Mean Reciprocal Rank is defined as the average of the multiplicative inverse of the rank position of the first correct answer for a set of queries. Shi et al. [19] define RR for a given recommendation list of a user by how early the first relevant recommended item is in the list and MRR as the average of the RR across all the recommendation lists for individual users. Thus, given L_u we define

$$\text{RR} = \frac{1}{\text{rank}(L_u)} \tag{4}$$

where $\text{rank} : A \to \mathbb{N}, \text{rank}(L_u) = \min\{i | 1 \leq i \leq |L_u|, L_u[i] \in T_u^+\}$.

Normalized Discounted Cumulative Gain (NDCG). It is a commonly used measure of the performance of web search engines in Information Retrieval. In the context of RS, NDCG measures the quality of a ranking based on the graded relevance of the recommended entities with respect to an ideal ranking of entities. It is defined as

$$\text{NDCG@}N = \frac{DCG@N}{IDCG@N} \tag{5}$$

$DCG@N$ is defined as

$$DCG@N = \text{rel}(L_u^N[1], T_u) + \sum_{i:=2}^{N} \frac{\text{rel}(L_u^N[i], T_u)}{log_2(i)}$$

where $\text{rel} : I \times \mathcal{P}(I) \backslash \emptyset \to \{0, 1, 2\}$ is a function that indicates the relevance of an item and we define it as

$$\text{rel}(i, T_u) = \begin{cases} 2, & \text{if } i \in T_u^+ \\ 1, & \text{if } i \in T_u^- \\ 0, & \text{otherwise} \end{cases}$$

In turn, $IDCG@N$ is the maximum possible $DCG@N$ where L_u^N is sorted in non-increasing order by item relevance. In this work, we consider items in T_u^+ as highly relevant, items in T_u^- as fairly relevant and those not in T_u as irrelevant items for recommending to a target user u. For further information we refer the reader to [12].

5. EXPERIMENTAL RESULTS

In this section we assess the performance of our approach and the baselines. We use the symbols $\triangle$ ($\triangledown$) and $\blacktriangle$ ($\blacktriangledown$) to denote statistically significant increase (decrease) at $p < 0.05$ and $p < 0.01$ levels, respectively, while the symbol $\circ$ is used to denote no significant difference according to Student's t-test. Since PPS shows the best results among the scoring functions we propose, as can be seen in Table 2, we present the results for the other methods when compared to PPS.

Table 2 provides the results for the methods and datasets considered in this work. The first columns presents the dataset name. The second column shows the method considered in the experiments. The next two columns give the results for MAP and MRR, respectively. The next two columns present results for P@N respectively for $N = 5$ and $N = 10$. Finally, the last two columns show results for NDCG@N for $N = 5$ and $N = 10$, respectively. Each result entry presents the proper symbol representing the t-test result, metric value and in parenthesis the mean percentage improvement of PPS over the considered method.

In the BX dataset, PPS provides statistically significant increase at the $p < 0.01$ level over the other methods for all metrics. When compared to P^3_α and RP^3_β, all the methods we propose provide better results, where PPS respectively shows results at least 219.8% and 296.5% superior than P^3_α and RP^3_β. When compared to the matrix factorization approaches, PPS respectively shows results at least 140.3% and 13.5% superior than BPRMF and WRMF. Finally, an interesting finding here is that MP provides results better than P^3_α, RP^3_β and even BPRMF in all metrics.

In the Cds dataset, PPS provides statistically significant increase at the $p < 0.01$ level over the other methods for all metrics. When compared to P^3_α and RP^3_β, all the methods we propose provide better results, where PPS respectively shows results at least 280.6% and 626.9% superior than P^3_α and RP^3_β. The matrix factorization approaches provide results inferior when compared to PPS, namely, the latter respectively shows results at least 143.8% and 44.8% better than BPRMF and WRMF.

In the Electronics dataset, PPS provides statistically significant increase at the $p < 0.01$ level over the other methods for all metrics. When compared to P^3_α and RP^3_β, all the methods we propose provide better results, where PPS respectively shows results at least 299.7% and 829% better than P^3_α and RP^3_β. When compared to the matrix factorization approaches, PPS respectively shows results at least 80.4% and 20% superior than BPRMF and WRMF. Finally, MP provides results better than P^3_α, RP^3_β and even BPRMF in all metrics.

In the Kindle dataset, PPS provides statistically significant increase at the $p < 0.01$ level over the other methods for all metrics. When compared to P^3_α and RP^3_β, all the methods we propose provide better results, where PPS respectively shows results at least 284.4% and 441% superior than P^3_α and RP^3_β. The matrix factorization approaches provide results inferior when compared to PPS, namely, the latter respectively shows results at least 291.3% and 209.1% better than BPRMF and WRMF.

In the ML 1M dataset, PPS provides statistically significant increase at the $p < 0.01$ level over MP, P^3_α and RP^3_β for all metrics. When compared to P^3_α and RP^3_β, all the methods we propose provide better results, where PPS respectively shows results at least 50.3% and 492.1% superior than P^3_α and RP^3_β. In turn, MP provide better results than P^3_α and RP^3_β for all metrics. When compared to BPRMF and WRMF, our methods do not show any statistically significant increase. In fact, except to MRR, PPS respectively show results at most 29.4% and 38.8% inferior than BPRMF and WRMF. Even though PPS shows results 4.3% better than BPRMF in MRR, there is no significance difference according to Student's t-test.

In the FilmTrust dataset, PPS provides statistically significant increase at the $p < 0.01$ level over P^3_α and RP^3_β for all metrics. When compared to P^3_α and RP^3_β, all the methods we propose provide better results, where PPS respectively shows results at least 23.3% and 128.8% superior than P^3_α and RP^3_β. When compared to BPRMF and WRMF, PPS provides statistically significant increase at the $p < 0.01$ level for MAP, MRR and P@5, but no significance difference for the other metrics. In fact, the most striking improvements are found in MRR, where PPS provides results 103% and 159.1% better than BPRMF and WRMF, respectively. In turn, PPS provides no significance difference for MAP, P@5 and P@10 when compared to MP. However, MP provides statistically significant increase at the $p < 0.05$ level in $NDCG$@5 and $NDCG$@10 over PPS. Thus, the most interesting finding here is that MP provides results at least as good as PPS except for MRR.

In the Epinions dataset, PPS provides statistically significant increase at the $p < 0.01$ level over BPRMF, MP, P^3_α and RP^3_β for all metrics. PPS respectively provides results at least 54.7%, 270.4% and 635.6% better than BPRMF, P^3_α and RP^3_β. One can see that MP provides better results than P^3_α and RP^3_β for all metrics. When compared to WRMF, PPS provides statistically significant increase at the $p < 0.01$ level for MAP, MRR, P@5 and P@10, where PPS shows results at most 38.2% superior. However, WRMF provides statistically significant increase at the $p < 0.05$ and the $p < 0.01$ in $NDCG$@5 and $NDCG$@10, respectively, over PPS.

Among the methods we propose in this work, PPS clearly outperforms PIS and PORS in all datasets and metrics, except for P@10 in the FilmTrust dataset where there is no statistically significant difference. PIS systematically provides results as good as PORS in all datasets and metrics except for the FilmTrust dataset where the latter outperforms the former. However, PIS is the most computationally demanding method we propose as it makes use of numerical integration.

In summary, PPS provides the best results in the BookCrossing and all Amazon datasets. In the MovieLens dataset, WRMF provides the best results. In the FilmTrust dataset, due to its simplicity, we advocate that MP is the best choice even though PPS provides a significant improvement in MRR. Finally, in the Epinions dataset, PPS provides the best results for all metrics except for $NDCG$.

6. CONCLUSION

In this paper, we studied ranking algorithms for personalized recommendation in the context of explicit feedback. We have presented three scoring methods based on the Bayesian paradigm that take advantage of the set of ratings given by the users. Also, we have proposed a strategy to cope with the memory and computational burden incurred from graph-based approaches in the literature, and analytically showed that our strategy is more efficient. The method we propose exploit three-step paths starting from the target user in the user-item graph to score items by means of the proposed scoring methods.

In our evaluation, we carried out experiments in seven datasets to assess the performance of the recommendation methods. We empirically showed that our PPS method clearly outperforms the graph-based approaches considered in all datasets and metrics. We also compared our methods against two representative matrix factorization methods and the results show that PPS clearly outperforms those methods in four datasets and is competitive with them in the other two; among the methods we propose, PPS clearly outperforms PIS and PORS. Thus, we confirm our hypothesis that graph-based approaches can take advantage of the set of ratings for improving the user satisfaction in recommender systems.

We plan to investigate the relaxation of the assumption $R = \{0, 1\}$, so that our method is able to deal with a multiple scale rating system. In this way, we can have a better idea of how effective our scoring methods are in such contexts and,

Table 2: Performance comparison of our method and baselines on datasets used for evaluation.

		MAP	MRR	P@5	P@10	NDCG@5	NDCG@10
BX	BPRMF	▼ 0.024 (190.1)	▼ 0.042 (140.3)	▼ 0.013 (228.1)	▼ 0.011 (167.2)	▼ 0.013 (237.8)	▼ 0.013 (190.7)
	MP	▼ 0.026 (165.9)	▼ 0.052 (94.5)	▼ 0.014 (185.5)	▼ 0.012 (148.0)	▼ 0.015 (186.9)	▼ 0.014 (162.9)
	P^3_α	▼ 0.021 (226.1)	▼ 0.032 (219.8)	▼ 0.011 (290.1)	▼ 0.008 (261.3)	▼ 0.012 (262.1)	▼ 0.011 (250.1)
	PIS	▼ 0.055 (26.0)	▼ 0.080 (25.4)	▼ 0.033 (24.6)	▼ 0.025 (19.3)	▼ 0.035 (25.8)	▼ 0.030 (22.6)
	PORS	▼ 0.039 (80.1)	▼ 0.055 (81.3)	▼ 0.021 (96.9)	▼ 0.017 (80.3)	▼ 0.022 (104.2)	▼ 0.019 (93.3)
	PPS	0.070	0.100	0.041	0.030	0.044	0.037
	RP^3_β	▼ 0.016 (327.5)	▼ 0.023 (336.9)	▼ 0.009 (363.4)	▼ 0.008 (296.5)	▼ 0.010 (363.9)	▼ 0.009 (322.1)
	WRMF	▼ 0.056 (25.0)	▼ 0.078 (28.4)	▼ 0.034 (20.9)	▼ 0.027 (13.5)	▼ 0.037 (20.0)	▼ 0.032 (14.5)
Cds	BPRMF	▼ 0.018 (202.2)	▼ 0.040 (153.5)	▼ 0.014 (177.0)	▼ 0.012 (143.8)	▼ 0.015 (168.3)	▼ 0.014 (145.4)
	MP	▼ 0.007 (662.4)	▼ 0.017 (482.6)	▼ 0.005 (633.6)	▼ 0.005 (507.2)	▼ 0.005 (641.2)	▼ 0.005 (551.5)
	P^3_α	▼ 0.014 (280.6)	▼ 0.026 (282.6)	▼ 0.009 (361.3)	▼ 0.007 (333.1)	▼ 0.009 (345.3)	▼ 0.008 (328.4)
	PIS	▼ 0.046 (17.4)	▼ 0.086 (16.5)	▼ 0.034 (14.7)	▼ 0.026 (13.9)	▼ 0.035 (15.5)	▼ 0.030 (15.3)
	PORS	▼ 0.028 (89.9)	▼ 0.057 (77.4)	▼ 0.021 (91.2)	▼ 0.017 (77.1)	▼ 0.020 (102.6)	▼ 0.018 (93.7)
	PPS	0.054	0.100	0.039	0.030	0.040	0.034
	RP^3_β	▼ 0.007 (626.9)	▼ 0.013 (670.7)	▼ 0.004 (921.9)	▼ 0.004 (724.2)	▼ 0.004 (934.1)	▼ 0.004 (783.3)
	WRMF	▼ 0.031 (72.4)	▼ 0.060 (67.1)	▼ 0.025 (56.7)	▼ 0.020 (48.0)	▼ 0.027 (50.7)	▼ 0.024 (44.8)
Electronics	BPRMF	▼ 0.014 (92.6)	▼ 0.031 (80.4)	▼ 0.009 (98.1)	▼ 0.007 (95.1)	▼ 0.009 (94.1)	▼ 0.008 (92.5)
	MP	▼ 0.017 (58.9)	▼ 0.037 (54.3)	▼ 0.010 (72.3)	▼ 0.008 (65.7)	▼ 0.010 (69.8)	▼ 0.009 (67.3)
	P^3_α	▼ 0.007 (299.7)	▼ 0.011 (406.0)	▼ 0.003 (480.4)	▼ 0.002 (488.5)	▼ 0.003 (420.7)	▼ 0.003 (435.5)
	PIS	▼ 0.025 (9.8)	▼ 0.052 (10.1)	▼ 0.016 (9.6)	▼ 0.012 (9.5)	▼ 0.015 (10.2)	▼ 0.013 (10.5)
	PORS	▼ 0.022 (24.1)	▼ 0.045 (27.1)	▼ 0.013 (28.1)	▼ 0.011 (23.3)	▼ 0.013 (32.6)	▼ 0.011 (29.3)
	PPS	0.027	0.057	0.017	0.014	0.017	0.015
	RP^3_β	▼ 0.003 (829)	▼ 0.004 (1199)	▼ 0.001 (2349)	▼ 0.001 (1726)	▼ 0.001 (2348)	▼ 0.001 (1896)
	WRMF	▼ 0.022 (23.5)	▼ 0.042 (33.8)	▼ 0.014 (26.2)	▼ 0.011 (23.0)	▼ 0.014 (21.6)	▼ 0.012 (20.0)
Kindle	BPRMF	▼ 0.019 (505.4)	▼ 0.039 (291.3)	▼ 0.013 (519.4)	▼ 0.011 (426.0)	▼ 0.013 (551.2)	▼ 0.012 (479.1)
	MP	▼ 0.006 (1679.1)	▼ 0.016 (870.9)	▼ 0.004 (1984.8)	▼ 0.003 (1694.2)	▼ 0.004 (2001.2)	▼ 0.004 (1787.8)
	P^3_α	▼ 0.027 (312.1)	▼ 0.039 (284.4)	▼ 0.017 (373.5)	▼ 0.013 (360.1)	▼ 0.018 (366.1)	▼ 0.015 (359.2)
	PIS	▼ 0.089 (27.3)	▼ 0.115 (31.2)	▼ 0.065 (25.9)	▼ 0.047 (24.4)	▼ 0.065 (27.5)	▼ 0.054 (26.4)
	PORS	▼ 0.065 (73.2)	▼ 0.087 (72.9)	▼ 0.046 (77.8)	▼ 0.036 (65.2)	▼ 0.044 (88.1)	▼ 0.038 (78.8)
	PPS	0.113	0.151	0.081	0.059	0.083	0.069
	RP^3_β	▼ 0.020 (471.3)	▼ 0.028 (441.0)	▼ 0.012 (568.1)	▼ 0.011 (459.9)	▼ 0.012 (605.4)	▼ 0.011 (523.5)
	WRMF	▼ 0.026 (342.2)	▼ 0.049 (209.1)	▼ 0.018 (350.8)	▼ 0.015 (293.8)	▼ 0.019 (339.0)	▼ 0.017 (303.6)
ML 1M	BPRMF	▲ 0.165 (-29.4)	○ 0.250 (4.3)	▲ 0.250 (-24.1)	▲ 0.213 (-25.2)	▲ 0.253 (-25.9)	▲ 0.236 (-27.2)
	MP	▼ 0.097 (19.2)	▼ 0.207 (25.8)	▼ 0.164 (15.4)	▼ 0.141 (12.9)	▼ 0.164 (14.1)	▼ 0.153 (12.5)
	P^3_α	▼ 0.071 (62.6)	▼ 0.173 (50.3)	▼ 0.114 (66.9)	▼ 0.091 (74.8)	▼ 0.121 (54.8)	▼ 0.108 (59.1)
	PIS	▼ 0.111 (4.5)	▽ 0.243 (7.0)	▼ 0.184 (3.2)	▼ 0.155 (2.5)	▼ 0.181 (3.3)	▼ 0.166 (3.3)
	PORS	▼ 0.096 (20.9)	▼ 0.205 (27.3)	▼ 0.152 (24.8)	▼ 0.136 (17.0)	▼ 0.143 (31.1)	▼ 0.138 (24.6)
	PPS	0.116	0.260	0.189	0.159	0.187	0.172
	RP^3_β	▼ 0.020 (492.1)	▼ 0.018 (1331.2)	▼ 0.003 (6916.0)	▼ 0.004 (4358.3)	▼ 0.004 (4774.6)	▼ 0.005 (3383.9)
	WRMF	▲ 0.190 (-38.8)	○ 0.261 (-0.3)	▲ 0.288 (-34.3)	▲ 0.241 (-34.0)	▲ 0.290 (-35.5)	▲ 0.268 (-35.9)
FilmTrust	BPRMF	▼ 0.439 (3.0)	▼ 0.161 (103.0)	▼ 0.307 (5.6)	○ 0.263 (1.8)	○ 0.362 (0.5)	○ 0.342 (0.6)
	MP	○ 0.449 (0.9)	▼ 0.266 (22.9)	○ 0.320 (1.2)	○ 0.269 (-0.6)	△ 0.370 (-1.8)	△ 0.349 (-1.5)
	P^3_α	▼ 0.349 (30.2)	▼ 0.230 (42.9)	▼ 0.249 (30.8)	▼ 0.182 (48.2)	▼ 0.296 (23.3)	▼ 0.251 (37.3)
	PIS	▼ 0.428 (5.8)	▼ 0.290 (12.5)	▼ 0.309 (4.9)	▽ 0.257 (4.2)	▼ 0.346 (5.2)	▼ 0.326 (5.3)
	PORS	▼ 0.434 (4.2)	▼ 0.290 (12.8)	▽ 0.314 (3.2)	○ 0.263 (1.6)	▼ 0.349 (4.1)	▼ 0.334 (3.0)
	PPS	0.453	0.327	0.324	0.268	0.364	0.344
	RP^3_β	▼ 0.189 (143.2)	▼ 0.113 (198.4)	▼ 0.059 (615.4)	▼ 0.119 (128.8)	▼ 0.066 (639.4)	▼ 0.120 (194.3)
	WRMF	▼ 0.435 (4.0)	▼ 0.126 (159.1)	▼ 0.303 (7.1)	○ 0.263 (1.6)	○ 0.363 (0.1)	○ 0.342 (0.4)
Epinions	BPRMF	▼ 0.018 (100.9)	▼ 0.042 (92.4)	▼ 0.014 (94.3)	▼ 0.013 (71.4)	▼ 0.017 (64.8)	▼ 0.016 (54.7)
	MP	▼ 0.015 (141.7)	▼ 0.034 (136.4)	▼ 0.011 (139.7)	▼ 0.010 (120.4)	▼ 0.015 (88.4)	▼ 0.013 (83.0)
	P^3_α	▼ 0.010 (270.4)	▼ 0.020 (306.0)	▼ 0.006 (350.4)	▼ 0.005 (341.0)	▼ 0.007 (296.8)	▼ 0.006 (295.8)
	PIS	▼ 0.033 (8.7)	▼ 0.074 (10.2)	▽ 0.026 (6.9)	▼ 0.020 (5.3)	▼ 0.026 (8.7)	▼ 0.023 (7.6)
	PORS	▼ 0.022 (65.7)	▼ 0.050 (63.2)	▼ 0.016 (76.3)	▼ 0.014 (53.7)	▼ 0.014 (100.9)	▼ 0.014 (79.9)
	PPS	0.036	0.081	0.027	0.022	0.028	0.025
	RP^3_β	▼ 0.005 (635.6)	▼ 0.008 (902.4)	▼ 0.002 (1073.0)	▼ 0.002 (901.4)	▼ 0.003 (985.4)	▼ 0.002 (892.8)
	WRMF	▼ 0.031 (15.6)	▼ 0.059 (38.2)	▼ 0.025 (11.4)	▼ 0.020 (6.6)	△ 0.030 (-4.8)	▲ 0.027 (-7.3)

eventually, propose further improvements. Also, we plan to carry out experiments to explain why our approach fails in improving results in MovieLens dataset when compared to the matrix factorization methods. Furthermore, we intend to assess how the path length affects the performance of our method. Finally, we are currently investigating the customization of the scoring methods so that probabilities are conditioned on the target user.

Acknowledgments

This work was partially funded by projects InWeb (MCT/ CNPq 573871/2008-6) and MASWeb (FAPEMIG/PRONEX APQ-01400-14), and by the authors' individual grants from CNPq and FAPEMIG.

References

[1] F. Christoffel, B. Paudel, C. Newell, and A. Bernstein. Blockbusters and wallflowers: Accurate, diverse, and scalable recommendations with random walks. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 163–170, New York, NY, USA, 2015. ACM.

[2] J. Cook. Exact calculation of beta inequalities. Technical report, Department of Biostatistics, November 2005.

[3] C. Cooper, S. H. Lee, T. Radzik, and Y. Siantos. Random walks in recommender systems: Exact computation and simulations. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 811–816, New York, NY, USA, 2014. ACM.

[4] F. Fouss, A. Pirotte, and M. Saerens. A novel way of computing similarities between nodes of a graph, with application to collaborative recommendation. In *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 550–556, Washington, DC, USA, 2005. IEEE Computer Society.

[5] 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 Transactions on Knowledge and Data Engineering*, 19(3):355–369, 2007.

[6] M. Galassi et al. *GNU Scientific Library Reference Manual*. Network Theory Ltd., 3rd edition, 2009.

[7] Z. Gantner, S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Mymedialite: A free recommender system library. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11, pages 305–308, New York, NY, USA, 2011. ACM.

[8] J. Golbeck and J. Hendler. Filmtrust: movie recommendations using trust in web-based social networks. In *Proceedings of Consumer the 3rd IEEE Communications and Networking Conference*, pages 282–286. IEEE, 2006.

[9] M. Gori and A. Pucci. Itemrank: A random-walk based scoring algorithm for recommender engines. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 2766–2771, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[10] F. M. Harper and J. A. Konstan. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems*, 5(4):19:1–19:19, Dec. 2015.

[11] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, pages 263–272, Washington, DC, USA, 2008. IEEE Computer Society.

[12] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems*, 20(4):422–446, Oct. 2002.

[13] P. Massa and P. Avesani. Trust-aware recommender systems. In *Proceedings of the 2007 ACM Conference on Recommender Systems*, RecSys '07, pages 17–24, New York, NY, USA, 2007. ACM.

[14] J. McAuley, R. Pandey, and J. Leskovec. Inferring networks of substitutable and complementary products. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794. ACM, 2015.

[15] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999.

[16] Y. Park and A. Tuzhilin. The long tail of recommender systems and how to leverage it. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, pages 11–18, New York, NY, USA, 2008. ACM.

[17] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, pages 452–461, Arlington, Virginia, United States, 2009. AUAI Press.

[18] G. Shani and A. Gunawardana. *Recommender Systems Handbook*, chapter Evaluating Recommendation Systems, pages 257–297. Springer US, Boston, MA, 2011.

[19] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, N. Oliver, and A. Hanjalic. Climf: Learning to maximize reciprocal rank with collaborative less-is-more filtering. In *Proceedings of the Sixth ACM Conference on Recommender Systems*, RecSys '12, pages 139–146, New York, NY, USA, 2012. ACM.

[20] A. P. Singh, A. Gunawardana, C. Meek, and A. C. Sudendran. Recommendations using absorbing random walks. In *North East Student Colloquium on Artificial Intelligence*, 2007.

[21] C. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In *Proceedings of the 14th International Conference on World Wide Web*, pages 22–32, New York, NY, USA, 2005. ACM.