Recommendations using Absorbing Random Walks

Ajit P. Singh AJIT@CS.CMU.EDU
Machine Learning Department, 5000 Forbes Ave, Pittsburgh PA 15213

Asela Gunawardana ASELAG@MICROSOFT.COM
Chris Meek MEEK@MICROSOFT.COM
Microsoft Research, One Microsoft Way, Redmond WA 98052

Arun C. Surendran ACSUREN@MICROSOFT.COM
Microsoft Adcenter Labs, One Microsoft Way, Redmond WA 98052

Abstract

Collaborative filtering attempts to find items of interest for a user by utilizing the preferences of other users. In this paper we describe an approach to filtering that explicitly uses social relationships, such as friendship, to find items of interest to a user. Modeling user-item relations as a bipartite graph we augment it with user-user (social) links and propose an absorbing random walk that induces a set of stationary distributions, one per user, over all items. These distributions can be interpreted as personalized rankings for each user. We exploit sparsity of both user-item and user-user relationships to improve the efficiency of our algorithm.

1. Introduction

Information filtering, finding items of interest, has long been categorized into three approaches (Malone et al., 1987; Resnick et al., 1994):

- *Content filtering*, which uses features of the items to find similar content. If the items are documents, this also called information retrieval.
- *Collaborative filtering*, which uses the preferences of other users to select items. Most commonly, these preferences take the form of item ratings (Resnick et al., 1994; Linden et al., 2003; Breese et al., 1998).
- *Social filtering*, which explicitly uses relations between users to find other items of interest.

There are variations and combinations of the above. For example, content filtering can exploit relationships between items - PageRank and related methods uses links between web pages (Page et al., 1998; Kleinberg, 1999; Lempel & Moran, 2001). Some methods combine both content and collaborative filtering (Goldberg et al., 1992; Cohn & Hofmann, 2000; Schein et al., 2002; Melville et al., 2002; Basilico & Hofmann, 2004). Others allow users to query social networks to focus filtering (Kautz et al., 1997).

Most works have focused on content and collaborative filtering, mainly due to the availability of data. Large document corpora (*e.g.*, TREC), and user rating data (McJones, 1997; Resnick et al., 1994) are readily available; large social networks are a relatively recent phenomenon and large data sets are difficult to come by. There has been an explosion of social Internet services in recent years: online gaming, social networking, instant messaging, blogs, etc. Such services hold the promise of augmenting existing user-item data sets with explicit relationships between users, e.g. buddy lists and blog-rolls. In this paper we specifically address the problem of using such social networking data to augment recommendation.

Collaborative vs Social Filtering: Collaborative filtering works on the principle of *homophilous diffusion*, where recommendations for a user are constructed from the preferences of similar users (Canny, 2002). Social filtering on the other hand uses association between users; there is some level of implicit assumption that people usually associate with others who are similar to them, but this is not necessarily true. There is a subtle difference between how links are

Presented at *North East Student Colloquium on Artificial Intelligence (NESCAI)*, 2007.

treated in collaborative filtering and in social filtering. While the existence of social links between users represents interaction or similarity, the absence of a social link does not mean that the two users are dissimilar. Similarly, while there are item ratings for each user it does not mean that unrated items are rated poorly. Were this the case we could induce a total ordering on the items for each user, which would greatly simplify evaluation (see §6.1).

Vector Space vs Graph methods: We can see from the above discussion that the definition of "similarity" is crucial to the idea of recommendation, and algorithms can be further characterized based on the way in which they compute this similarity. For example, similarities between users can be computed using vector space models (Goldberg et al., 2001; Hofmann, 2003; Rennie & Srebro, 2005) where the items form an m-dimensional space in which users are embedded. The data is usually available as an $n \times m$ matrix, where the rows correspond to users and a column corresponds to items. Vector-space models are computationally expensive, e.g. computing similarities between all pairs of users is infeasible when $n \gg 10^3$. Moreover, vector-space models do not readily allow the addition of links between users or links between items.

As an alternative to vector-space methods, the user-item matrix can be represented as a graph; specifically, it can be represented as a bipartite graph. In these methods similarity between users is based on statistics such as commute or hitting times between their nodes (Kubica et al., 2003; Fouss et al., 2004; Brand, 2005). Under some conditions, the graph can be treated as a Markov chain, and its long-term stationary distribution (Page et al., 1998; Kleinberg, 1999; Lempel & Moran, 2001) can be used to induce a global ranking. However, this global ranking is inconvenient for direct use in recommendation systems, since the recommendations would not depend on the user. In contrast to vector space methods, graph methods are representationally flexible. However, this flexibility is often associated with an increased computational cost.

Value of Recommendations: Many previous approaches ignore the fact that the utility of a recommendation depends on the recommendation being made. In the real-world online gaming application we describe in §5.1, the most popular 1% of items are owned by over 50% of users, and account for over 15% of all items owned. Similarly the most popular 7.5% of items are owned by over 10% of users, and account for over 60% of items owned. Because of the highly skewed nature of item ownership, the usual practice of randomly choosing a set of test items and users and then measuring how well a system performs on this test set is misleading. This is because a small number of popular items will dominate the test set. Thus, a system that predicts the most popular items will fare well in this evaluation. However, the actual utility of such a system is minimal – users are typically aware of the most popular items, and do not need a recommendation system to find out about them.

Our Contribution: In this paper we present an approach that combines relational graphs (like social networks) and ownership data (user-item graphs) to make recommendations. Our method is based on a random walk on a graph with absorbing states - we call this an "absorbing random walk". We use this approach for the task of recommendation for online gaming. This is a domain that has both social features and user preference data. Our goal is to use the games a user owns (user preferences) and his network of friends (social links) to suggest other items the user might want to buy. Evaluating recommendation can be tricky for various reasons, including those discussed above. We propose two new evaluation criteria, the population R-score R_p and Recall vs. item popularity curves, for recommendations systems. We compare our approach against standard vector-based approaches as well as a state-of-the-art co-clustering algorithm.

2. Recommendation as a Link Prediction Task

Given a set of users and items, ownership can be represented as a bipartite graph with links between users and items they own. Recommendation can be thought of as the problem of inducing a ranking on the item nodes for each user node. The k top-ranked items are returned as the most likely items that the user should own. Evaluating recommendations requires ground truth, which is only available for the items rated (or owned) by a user. Starting with a graph where all the edges are known, testing can be done by holding out some known links, learning on the rest of the graph and trying to predict the held-out links. For each held out link the more highly ranked the corresponding item is in the recommendations for the corresponding user, the better the prediction. In this paper, we focus on predicting whether a user owns an item (binary voting), which can be viewed as link prediction.

2.1. Augmented Bipartite Graph

We treat relationships between users as augmenting the bipartite graph with social links (Figure 1). Thus we now have the following definition:

Definition 1 *Given sets* $\mathcal{U} = \{u_1, \cdots, u_n\}$ *and* $\mathcal{I} = \{g_1, \cdots, g_m\}$ *of users and items, the* ownership graph $\mathcal{G}$ *is given by* $\mathcal{G} = (\mathcal{U}, \mathcal{I}, \mathcal{O}, \mathcal{S})$ *where*

$$\mathcal{O} = \{(u, g) : (u, g) \in \mathcal{U} \times \mathcal{I}\}$$
$$\mathcal{S} = \{(u, u') : (u, u') \in \mathcal{U} \times \mathcal{U}\}$$

For convenience, we slightly abuse notation and use

$$\mathcal{O}(u) \triangleq \{g : (u, g) \in \mathcal{O}\}$$
$$\mathcal{S}(u) \triangleq \{u : (u, u) \in \mathcal{S}\}$$

to denote the items owned by u and the users socially linked to u respectively.

The adjacency matrix of this augmented bipartite graph can be written as

$$W = \begin{bmatrix} W_{\mathcal{S}} & W_{\mathcal{O}} \\ W_{\mathcal{O}}^T & 0 \end{bmatrix}$$

where $W_{\mathcal{O}}$ and $W_{\mathcal{S}}$ are the adjacency matrices corresponding to the links $\mathcal{O}$ and $\mathcal{S}$.

3. Absorbing Random Walks

Traditional random walk approaches convert W to a Markov chain over nodes defined by the row stochastic matrix $T = D^{-1}W$ where D is a diagonal matrix with entries $D_{ii} = \sum_j W_{ij}$. This Markov chain can be used in different ways to to induce a ranking on the nodes. For example, statistics of the random walks, such as hitting times between nodes can be used. Some popular methods methods allow, with small probability, a transition to any other state to ensure the existence of a unique stationary distribution (Lempel & Moran, 2001; Page et al., 1998). Such a distribution can be used to find the global ranking of nodes, which can be interpreted as the long term probability of landing on any node irrespective of where we start from. Spectral methods that work directly on the bipartite graph can also be used for recommendations (see §4).

Such rankings are problematic for two reasons: lack of personalization and lack of focus. Typical random walk statistics produce the same ranking, regardless of which user wants recommendations. Moreover, we are only interested in ranking over the items – such techniques can expend energy in providing ranking on user nodes which is unnecessary. In PageRank, the first problem is addressed by topic-sensitive (Haveliwala, 2002) or query-dependent (Richardson & Domingos, 2001) variants.

We propose a new method where we perform a random walk on the augmented bipartite graph shown before.

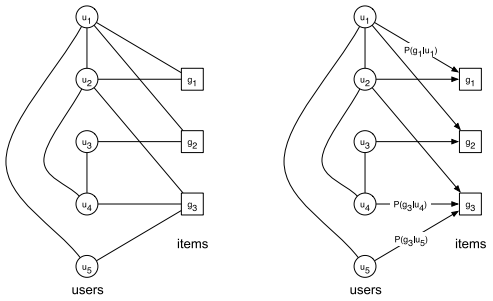


Figure 1. An augmented bipartite graph (left) and the corresponding absorbing random walk (right).

We treat this graph as a special Markov chain which has *absorbing states* i.e. states which once reached cannot be transitioned out of. We call this approach an "absorbing random walk". The walk begins from the user we want to produce recommendations for – either transitioning to a friend with probability α or to an item with probability $1 - \alpha$. If the walker chooses to transition to a friend, one is chosen uniformly at random from the list of people the user is connected to. If the walker chooses to transition to a item, one is chosen uniformly at random from the list of items a user owns. Items are absorbing states, so the walker cannot leave an item. Figure 1 illustrates the Markov chain. Formally, the absorbing random walk is defined by the following transition probabilities:

$$P(u'|u) = \begin{cases} \frac{\alpha}{|\mathcal{S}(u)|} & \text{if } u' \in \mathcal{S}(u) \\ 0 & \text{otherwise,} \end{cases}$$
$$P(g|u) = \begin{cases} \frac{(1-\alpha)}{|\mathcal{O}(g)|} & \text{if } g \in \mathcal{O}(u) \\ 0 & \text{otherwise,} \end{cases} \quad (1)$$
$$P(g'|g) = \begin{cases} 1 & \text{if } g' = g \\ 0 & \text{otherwise,} \end{cases}$$
$$P(u|g) = 0$$

It can be seen that the probability that the walker is still in a user state after t steps is α^t, and that the walk is absorbed with probability $(1-\alpha^t)$ after t steps. The probability that the walk is absorbed after step t is $\alpha^{t-1}(1-\alpha)$. Thus, we have

Property 1 *The expected number of steps before absorption is* $\frac{1}{1-\alpha}$.

For each game, the probability $P^{(t)}(g|u)$ of being at item g after starting at user u and taking t steps is either 0 for all t or monotonically increasing, since g is absorbing. Either $P^{(t)}(g|u)$ has trivially converged

or is bounded above and increasing (in which case, it must converge). Moreover, the walk is reducible (since it has absorbing states), so that it is not guaranteed to have a unique stationary distribution. Therefore

Property 2 *The walk starting at user u converges to an* absorbing distribution *$P_u(g)$ over items g. In general, the absorbing distribution depends on u.*

Since the probability of not being absorbed after t steps is α^t, the most any $P^{(t)}(g|u)$ can change is α^t, giving

Property 3 *The error in approximating $P_u(g)$ by $P^{(t)}(g|u)$ is at most α^t.*

Danglers: In practice, there are users with no social links, or danglers. Possible solutions include (i) backing-off to a vector-space method (ii) connecting the dangler to every other user, averaging all the user's item distributions (iii) picking a random subset of users, and creating artificial social links to the k users with the most similar set of items.

Ranking: Once $P_u(g)$ is computed each item can be ranked in order of decreasing probability. Because the contribution of the t-th step to $P_u(g)$ decays exponentially, the initial step, and thus the items we know the user owns, will dominate. We simply remove those items from the ranking. Another approach is to force the first step of the walk to be to a friend. We use the former approach throughout.

3.1. Biased Absorbing Random Walks

Our assumption in §3 is that social links imply similar interests, which is not always true. To address this problem we bias the random walk. A social link is still chosen with probability α, but instead of choosing a link uniformly at random the choice is biased by the similarity of the destination user. Thus the user-user transition probability in equation (1) is replaced by

$$P(u'|u) = \begin{cases} \alpha \frac{e^{\beta \cos(g(u),g(u'))}}{\sum_{u'' \in \mathcal{S}(u)} e^{\beta \cos(g(u),g(u''))}} & \text{if } u' \in \mathcal{S}(u) \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

where

$$\cos(g, g(u')) = \frac{g(u) \cdot g(u')}{|g(u)||g(u')|}.$$

$g(u)$ is the binary vector of items owned by user u and β is a free parameter that reflects the preference for transitioning to a friend who owns the same items. Since the unbiased random walk is a special case ($\beta = 0$) we refer only to the biased case in the sequel.

3.2. Computing the Absorbing Distribution

The transition probabilities of equations (1) and (2) can be represented by a transition matrix

$$P = \begin{bmatrix} \alpha U & (1-\alpha)G \\ 0 & I \end{bmatrix} \tag{3}$$

where U is an $n \times n$ transition matrix on users, G is an $n \times m$ matrix of user to item transitions, and I is the $m \times m$ identity matrix.

The absorbing distributions $P_u(g)$ can be obtained by reading off the upper right hand $n \times m$ block of P^∞, but doing this directly is computationally expensive. Since the chain is reducible, we cannot compute a stationary distribution by finding the first eigenvector. The naïve approach is to compute a t step walk

$$P^t = \begin{bmatrix} \alpha^t U^t & (1-\alpha)\sum_{k=0}^{t-1} \alpha^k U^k G \\ 0 & I \end{bmatrix} \tag{4}$$

for some value of t and appeal to Property 3. However, in practice, we may be unable to use high enough a value of t to have a good approximation for the values of α we wish to use. One justification for using a t step walk even for larger α is that in many social networks the size of the connected components follows a power law (Albert & Barabsi, 2002). If only a few users are reachable in the walk, then it is often the case that large values of α do not significantly alter rankings.

Computing P^t can be done in $O(tn^3 + n^2m)$ time by taking advantage of the structure of P. To address the issue of computing U^k, we note that often the goal is not to produce recommendations for all users, but only a small subset of users. If there are r users who need recommendations, we can replace each U^k term in equation 4 with $\tilde{U}U^{k-1}$ where $\tilde{U}$ is the $r \times n$ row-minor. In this case, the computation required is $O(trn^2 + rnm)$. While this is the worst case, the average case is in fact much better, since the sparse block structure of the social links ensures the fill-in of $\tilde{U}U^{k-1}$ is manageable. In the online gaming domain, the probability that a user has d social links decreases super-exponentially in d. The savings are significant.

A related approach for exploiting sparsity for computing item-item similarities is used in (Linden et al., 2003). There, a cosine-similarity between two items is computed iff both items were bought by the same customer. Analogously, one could compute user-user similarities for every pair of users who owned the same

item. In our online gaming data, over half the users own the most popular game, limiting the gains to be had by exploiting sparsity of user-user similarities.

The use of absorbing random walks is not restricted to augmented bipartite graphs. (Zhu et al., 2007) uses absorbing random walks to improve ranking diversity. Starting with a non-absorbing walk over a weighted graph, the most highly ranked node is changed to an absorbing state at each iteration. Then, the node visited most frequently before absorption becomes the most highly ranked node. When averaged with a good static ranking, one can balance relevance and diversity.

4. Related Methods

As discussed in Section 1 there are many recommendation systems. Here we discuss a few methods that are popular in the recommendation community. We will later compare our absorbing random walk method against these two.

Cosine similarity based item-item recommendations We follow the Amazon approach (Linden et al., 2003), which computes cosine similarity between items by representing them as vectors of users. The sparsity of the user vectors for all but the most popular items makes this method quite efficient. To make a recommendation for a user, the cosine similarity between a pair of items is extended to a cosine similarity between a user u and an item g through

$$\cos(u, g) = \frac{1}{|\mathcal{O}(u)|} \sum_{g' \in \mathcal{O}(u)} \cos(g', g).$$

Items are then ranked in order of decreasing $\cos(u, g)$, excluding items that are already owned[1].

Cosine similarity based collaborative filtering One very common approach in collaborative filtering is to rank users based on some similarity or distance. The cosine distance is a commonly used in these situations. Then recommendations for user u are produced by averaging the preferences of the N nearest neighbours of $u, \Gamma(u, N)$

$$P_u(g) = \sum_{u \in \Gamma(u,N)} w(u')P(g|u')$$
$$w(u') = \frac{\cos(g(u), g(u'))}{\sum_{u''} \cos(g(u''), g(u'))}.$$

Items are ranked in order of decreasing $P_u(g)$, excluding owned titles.

[1]We also investigated combination using the max operator instead of averaging, but averaging gave better results.

Spectral Co-clustering A different approach to averaging the preferences of other users together is clustering. We use the spectral co-clustering approach of (Dhillon, 2001), which simultaneously embeds users and items onto a low-dimensional space. Applying the idea to our current task, we derive the first s left and right singular vectors of G and embed both users and items into the subspace derived from these singular vectors. In this embedded subspace, the items are ordered according to their distance from a user; recommendations are made based on this ranking.

Alternatively, we follow the same process for the matrix $A = U^T G$. We can interpret this matrix as a 2-step random walk where a user picks another user they are connected to, and then picks a game that the other user owns. The spectral algorithm can be thought of as finding a sub-space embedding of this modified graph that uses both social network as well as ownership information. We compare the absorbing random walk with both these algorithms.

5. Data Sets

5.1. Online Gaming

We evaluate the methods described above using data from an online gaming service. The service allows a few million users to interact with each other and play games. In particular, the users of the service maintain friend lists, and their game consoles report games played, high scores, and other achievements to the service. In addition, the service allows users to download content such as media, and extensions to games.

We view the service as an augmented bipartite graph, with nodes representing users and content. The graph has two types of links: user-user links representing friend relationships, and user-content links representing ownership of items of content. At the time when the data was collected, there were over 400 items of content. We use two subsets of the users: a "small" set containing over 60,000 users and a "large" set containing over 800,000 users. In either case, all users have at least one friend in the set, and own at least one piece of content. User-user links leading outside the "small" or "large" set were omitted.

5.2. NIPS Corpus

The absorbing random walk approach is motivated by a social network on users, but the algorithm can also be used where the links are not explicitly social interactions. A publicly available example of such a data set is the NIPS corpus (Roweis, 2002), a collection of papers from a machine learning conference covering 1987-

1998. The entities in the corpus are papers, authors, and words. As an augmented bipartite graph we view the problem as one of recommending words to papers (treat papers as users and words as items). Social links between papers are induced by co-authorship. An edge of weight k is placed between two papers if they share k authors. While this task is somewhat artificial, we report results on it for the sake of reproducibility of results on an easily available data set.

In the online gaming data $n \gg m$; the situation is reversed on the NIPS corpus ($n = 1740$ and $m = 13,649$). Every word is associated with at least six papers, and there are over 15K social links and 900K paper-word links. As with the online gaming data, the chance that a paper has d social links diminishes superexponentially in d.

6. Experiments

6.1. Evaluation criteria

While we view generating recommendations as a link prediction task, it does not change our goal of presenting a ranked list to the user. The problem with zero-one loss is that it assumes the user only reads the first item on his list. This can be relaxed by assuming that a user's likelihood of considering the first j items decreases exponentially in j (Breese et al., 1998):

$$p(u,t) = 2^{-(j_t-1)/(\lambda-1)}$$

where j_t is the position of title t on the list (∞ if the title is not on the list). The half-life λ is the position in the list where a user has a 50-50 chance of stopping. In our experiments $\lambda = 5$.

Another limitation of zero-one loss is that it treats every link as equally hard to predict. Guessing that a user owns a popular item is easy; guessing that a user owns an unpopular or niche item is harder. Define the utility function

$$U(u_i, g_i) = -\log(\tilde{p}(g_i))$$
$$\tilde{p}(g_i) = \frac{1}{n}\sum_{i=1}^{n} \delta(u_i \text{ owns } g_i).$$

The expected utility, or R-score (Breese et al., 1998), given test links from a user, $\{(u, g_1), \ldots, (u, g_\ell)\}$, is

$$R_u = \sum_{i=1}^{\ell} p(u, g_i) U(u, g_i).$$

Because different users can receive different numbers of recommendations, the R-scores are normalized by the maximum possible R-score and then averaged over users. The maximum possible R-score is achieved by placing the held out links at the top of the ranking, sorted in order of decreasing utility.

$$E[R_u] = \frac{1}{n}\sum_{u=1}^{n} \frac{R_u}{R_u^{max}}.$$

Averaging over users obscures the fact that some users own less popular items. For example, in the extreme case where we have two test users, one of whom has only a very rare test item and the other of whom has only a very popular item, predictions of the two test items are weighted equally.

We propose to instead average over links $\{(u_1, g_1), \ldots, (u_\ell, g_\ell)\}$

$$R_p = \frac{\sum_i p(u_i, g_i) U(u_i, g_i)}{\sum_i U(u_i, g_i)}$$

which we refer to as the population R-score.

If we restrict ourselves to holding out at most one test link per user, then recall has a simple interpretation. Following (Sarwar et al., 2000) a set of test links are held out $\{(u_1, g_1), \ldots, (u_\ell, g_\ell)\}$. An algorithm outputs $k \ll m$ recommendations, for a chosen threshold k. Recall is the percentage of links where g_i appears in the top k recommendations for user u_i. If we hold out at most one link per user, recall can be interpreted as the percentage of users who were satisfied by their recommendations (*i.e.*, the algorithm ranked the held out item highly). We report recall for test items binned by popularity. Items are sorted in order of popularity, and binned so that each bin contains approximately the same number of test links. Recall is reported separately for each of these bins. Thus, we can evaluate whether an algorithm performs well because it is good at predicting popular items (not useful in practice) or at predicting less popular items.

6.2. Online Gaming Results

From the small online gaming data set two thousand user-item links were selected at random. Because n is large, it is unlikely that more than one link is removed per user, minimizing perturbations in $\mathcal{G}$.

The item-item approach is denoted Amazon. The cosine similarity approach (§4) using N nearest neighbours is denoted Cosine-N. The biased random walk approach using a t-step approximation is denoted Absorbing-t. The spectral embedding algorithm which is based on only the user-item ownership graph is called Spectral-A. The algorithm which uses both social and ownership networks is called Spectral-B. For

Table 1. Small online gaming data set, using R-scores.

Method	R_p	$E[R_u]$
Amazon	0.429	0.508
Cosine-1	0.194	0.220
Cosine-10	0.423	0.482
Cosine-100	*0.485*	*0.561*
Cosine-1000	0.478	0.559
Absorbing-1	0.351	0.408
Absorbing-7	0.420	0.492
Spectral-A	0.351	0.437
Spectral-B	0.384	0.455

the small online gaming data set, we use all the singular vectors that correspond to the non-zero singular values (there are 435 of them).

Holding out two thousand links for training and using a 7-step walk the learnt parameters were $(\alpha, \beta) = (0.99, 10)$. Because the subsampling of the small data set artificially created many users who connect only to each other and other small connected components, there is little penalty for increasing α without bound. In addition, this minimizes the effective power of the random walk.

On this data set, Cosine-100 wins both in terms of R-scores (Table 1) and in terms of recall (Figure 2). Because n is small, it is possible to compute the cosine similarity between each test user and the $n-1$ other users. Spectral-B (using the additional social information) is able to outperform Spectral-A, the plain co-embedding algorithm. Even with a large number of singular vectors, these two approaches do not outperform Absorbing-7 (Table 1).

Breaking down the recall by item popularity (Figure 2) provides a clear illustration of the problem that beguiles recommendation algorithms. We expect a large fraction of the N nearest neighbours to own a popular item, but a much smaller fraction of those neighbours to own a less popular item. In the limit of an item owned by no one, this problem is known as *cold-start* (Maltz & Erlich, 1995). The first seven bins in Figure 2 contain the twenty most popular items. In contrast, the rightmost bin contains almost four hundred items.

6.3. Large Gaming Set

The same procedure was done using the large online gaming data set. On this set, we drew 2500 test links and 2500 training links. Since $n \approx 900,000$ we could not feasibly compute the required $900,000 \times 2500$ cosine similarities necessary for the cosine similarity

Table 2. Large online gaming data set, using R-scores.

Method	R_p	$E[R_u]$
Amazon	0.296	0.384
Absorbing-3	0.312	0.397

Table 3. NIPS corpus, using R-scores.

Method	R_p	$E[R_u]$
Cosine-1	0.024	0.031
Cosine-10	0.015	0.019
Cosine-100	0.019	0.025
Cosine-1000	0.020	0.026
Cosine-All	0.020	0.026
Absorbing-10	*0.039*	*0.048*
Spectral-A	0.029	0.038
Spectral-B	0.027	0.035

based user-user technique. Using a 3-step walk the learnt parameters were $(\alpha, \beta) = (0.6, 14)$, which suggests that treating every friend as equally informative is unwise. The R-scores for the Amazon algorithm are compared with the 3-step walk in Table 2. The results suggest that on the full data set it is significantly harder to produce good recommendation, as confirmed by the recall curve (Figure 3). The absorbing walk outperforms the Amazon algorithm on the larger set, perhaps confirming that the sub-sampling introduced too many small connected components in the smaller set. The large data set was too big for us to compute the singular vectors (memory limitations).

6.4. NIPS Results

Our procedure for the NIPS corpus is analogous to that used for the online gaming data. Two thousand links were held out at random, half for training walk parameters, and words were grouped into bins by popularity. The parameters chosen by maximizing R_p on the training links are $(\alpha, \beta) = (0.475, 10)$.

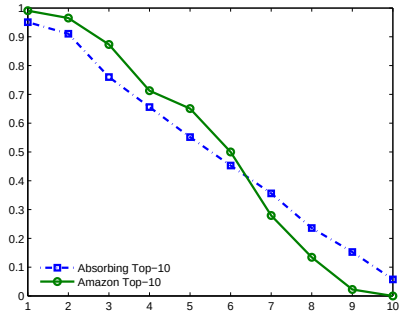
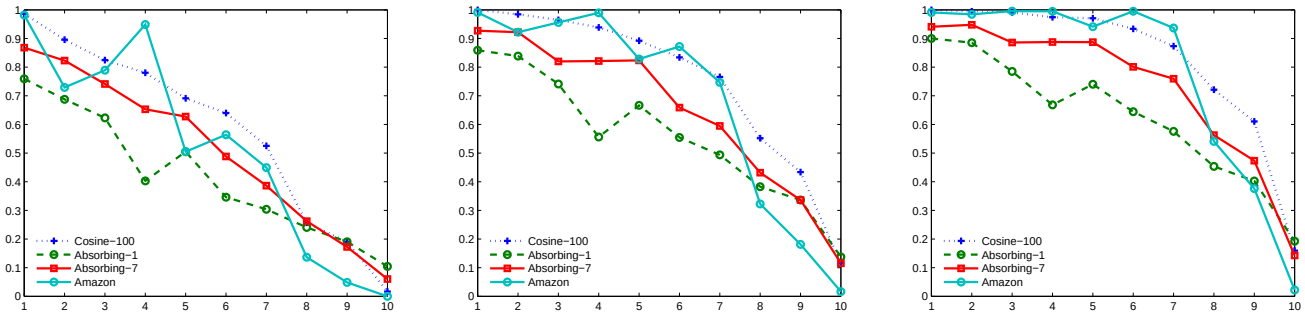


Figure 3. Recall vs. item popularity for the large online gaming data set.



(a) Random Walk. Top-5 (b) Random Walk. Top-10 (c) Random Walk. Top-15

Figure 2. Online gaming (small) – Recall vs. item popularity. The x-axis lists the ten bins the games were divided into. The higher the bin number; the less popular the games in the bin. Top-k: k recommendations were made to each user.

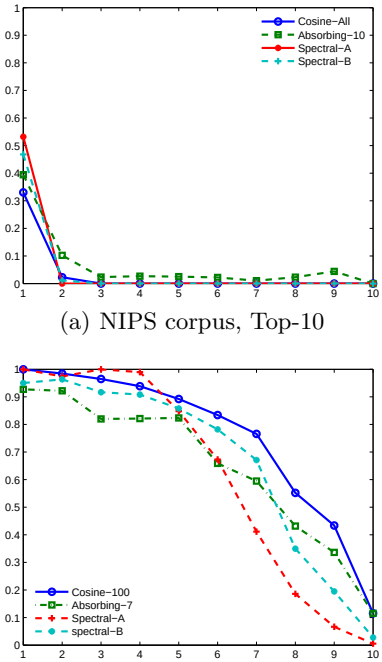
Since the number of papers is small, we can compute the cosine similarity between all pairs of users (Cosine-All). It is also possible to compute a 10-step walk. The network on papers lends itself to a random walk model, the social network is less sparse than the online gaming data. Table 3 shows that the absorbing random walk does almost twice as well as the best cosine method. The two spectral algorithms best the cosine method, but are poorer than the absorbing random walk approach. For the NIPS corpus, the spectral algorithms use $s = 1000$ singular values. Since m is large in this task, and word occurrence quite dense, we were unable to use the Amazon approach.

More importantly from the perspective of recommendation algorithm is Figure 4. If we ask each algorithm to choose 10 or 20 words for each paper, the absorbing random walk dominates all the cosine methods. Even when a cosine method does better on popular words, the absorbing random walk is better at matching rare words to papers.

6.5. Learning Random Walks

An absorbing random walk has free parameters, α and β. Both parameters have compelling interpretations – α measures how far out influence should extend on the social network (social filtering) while β measures the degree of homophilous diffusion (collaborative filtering). However, it is not obvious what these values should be for a particular data set. As a proof of concept, we propose an alternating maximization procedure for learning (α, β).

Our goal is to maximize R_p, which corresponds best to our view of recommendation as link prediction. Training links are held out, and we can evaluate R_p on the training lists, $\hat{R}_p$. Fixing β, maximize α by grid search; then fixing α maximize β by grid search (a loose upper bound on β is chosen). Alternating maximization of α and β leads to a local optima. Computing ∇R_p using equation 4 is a topic of future research.



(a) NIPS corpus, Top-10

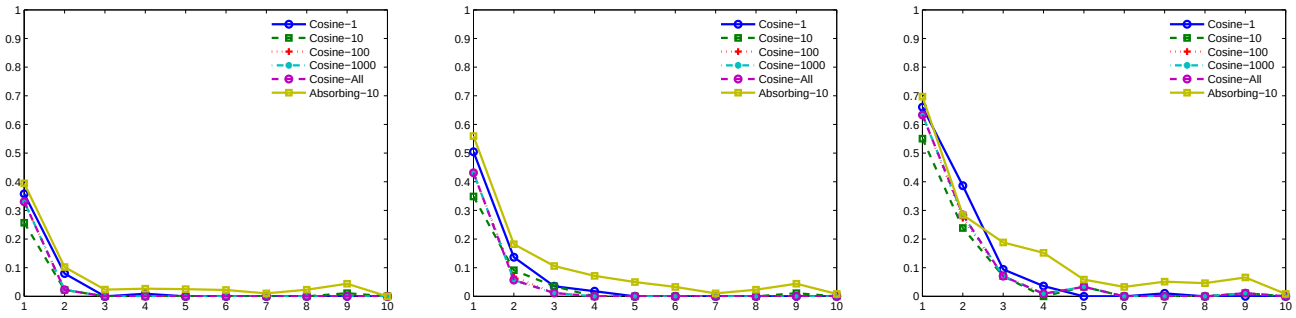
(b) Online gaming (small), Top-10

Figure 5. Comparison of spectral co-clustering on both the online gaming and NIPS problems. Recall vs. item popularity.

7. Discussion

There are three areas for advancement on this work: computational, user-user transition models, and generalization to arbitrary relational domains.

While our approach represents a more efficient



(a) Top-10 (b) Top-20 (c) Top-50

Figure 4. (NIPS corpus) Recall vs. item popularity. The x-axis lists the 10 bins the words were divided into. The higher the bin number, the less popular the words in the bin. Top-k is the number of recommendations each algorithm made.

memory-based model, which can exploit sparse matrix operations, it still does not scale to $n \gg 10^6$. The most obvious way to avoid matrix-matrix multiplies is to simulate the random walk – *i.e.*, Monte Carlo estimation of $P_u(g)$.

The only feature of our user-user transition model is the item ownership vector. More features may be useful. For example, on the online gaming data the age of each user is available; papers are categorized by topic in the NIPS corpus. To deal with the problem of very sparse social structure (*e.g.*, danglers) incorporating a small chance of transition between any pair of users may help.

The class of problems we have dealt with have a fixed relational structure. There are two types of entities (users, items) and fixed types of relationships (user-user, user-item). Exploring whether this algorithm generalizes to arbitrary instances of relational databases is an open question.

References

Albert, R., & Barabsi, A.-L. (2002). Statistical mechanics of complex networks. *Rev. Mod. Phys.*, *74*, 47–97.

Basilico, J., & Hofmann, T. (2004). Unifying collaborative and content-based filtering. *Proc. 21st Intl. Conf. Machine Learning* (p. 9). New York, NY, USA: ACM Press.

Brand, M. (2005). *A random walks perspective on maximizing satisfaction and profit* (Technical Report TR2005-050). MERL.

Breese, J. S., Heckerman, D., & Kadie, C. (1998). *Empirical analysis of predictive algorithms for collaborative filtering*Technical report MSR-TR-98-12). Microsoft Research.

Canny, J. (2002). Collaborative filtering with privacy via factor analysis. *Proc. 25th Intl. ACM SIGIR Conf. Inf. Retr.*.

Cohn, D., & Hofmann, T. (2000). The missing link - a probabilistic model of document content and hypertext connectivity. *Advances in Neural Information Processing Systems 13*.

Dhillon, I. S. (2001). Co-clustering documents and words using bipartite spectral graph partitioning. *Proc. 7th ACM SIGKDD Intl. Conf. Knowledge Discovery and Data Mining* (pp. 269–274). San Francisco, California: ACM Press.

Fouss, F., Pirotte, A., Renders, J.-M., & Saerens, M. (2004). A novel way of computing dissimilarities between nodes of a graph, with application to collaborative filtering. *Proc. ECML/PKDD Workshop on Statistical Approaches to Web Mining*.

Goldberg, D., Nichols, D., Oki, B. M., & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Commun. ACM*, *35*, 61–70.

Goldberg, K., Roeder, T., Gupta, D., & Perkins, C. (2001). Eigentaste: A constant time collaborative filtering algorithm. *Inf. Retr.*, *4*, 133–151.

Haveliwala, T. H. (2002). Topic-sensitive pagerank. *Proc. 11th World Wide Web Conf.*.

Hofmann, T. (2003). Collaborative filtering via Gaussian probabilistic latent semantic analysis. *Proc. 26th Intl. ACM SIGIR Conf. Inf. Retr.* (pp. 259–266). New York, NY, USA: ACM Press.

Kautz, H., Selman, B., & Shah, M. (1997). Referral web: combining social networks and collaborative filtering. *Commun. ACM*, *40*, 63–65.

Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. *J. ACM*, *46*, 604–632.

Kubica, J., Moore, A., Cohn, D., & Schneider, J. (2003). Finding underlying connections: A fast graph-based method for link analysis and collaboration queries. *Proc. 20th Intl. Conf. Machine Learning*.

Lempel, R., & Moran, S. (2001). SALSA: the stochastic approach for link-structure analysis. *ACM Trans. Inf. Syst.*, *19*, 131–160.

Linden, G., Smith, B., & York, J. (2003). Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, *7*, 76–80.

Malone, T. W., Grant, K. R., Turbak, F. A., Brobst, S. A., & Cohen, M. D. (1987). Intelligent information-sharing systems. *Commun. ACM*, *30*, 390–402.

Maltz, D. A., & Erlich, K. (1995). Pointing the way: Active collaborative filtering. *Proc. CHI 95: Human Factors in Computing Systems*.

McJones, P. (1997). EachMovie collaborative filtering data set. DEC Systems Research Center. http://www.research.digital.com/SRC/eachmovie/.

Melville, P., Mooney, R. J., & Nagarajan, R. (2002). Content-boosted collaborative filtering for improved recommendations. *18th Natl. Conf. Artificial Intelligence* (pp. 187–192). Menlo Park, CA, USA: American Association for Artificial Intelligence.

Page, L., Brin, S., Motwani, R., & Winograd, T. (1998). *The PageRank citation ranking: Bringing order to the web* (Technical Report). Stanford University.

Rennie, J. D. M., & Srebro, N. (2005). Fast maximum margin matrix factorization for collaborative prediction. *Proc. 22nd Intl. Conf. Machine Learning* (pp. 713–719). New York, NY, USA: ACM Press.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). Grouplens: an open architecture for collaborative filtering of netnews. *Proc. ACM Conf. Computer Supported Cooperative Work (CSCW)* (pp. 175–186). New York, NY, USA: ACM Press.

Richardson, M., & Domingos, P. (2001). The intelligent surfer: Probabilistic combination of link and content information in PageRank. *Advances in Neural Information Processing Systems 14*. MIT Press.

Roweis, S. (2002). NIPS conference papers vols. 0-12. http://www.cs.toronto.edu/ roweis/data.html. Data Set.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2000). Analysis of recommendation algorithms for e-commerce. *Proc. 2nd ACM Conf. Electronic Commerce* (pp. 158–167). New York, NY, USA: ACM Press.

Schein, A. I., Popescul, A., Ungar, L. H., & Pennock, D. M. (2002). Methods and metrics for cold-start recommendations. *Proc. 25th Intl. ACM SIGIT Conf. Inf. Retr.* (pp. 253–260). New York, NY, USA: ACM Press.

Zhu, X., Goldberg, A., Gael, J. V., & Andrzejewski, D. (2007). Improving diversity in ranking using absorbing random walks. *Human Language Technologies: Ann. Conf. North American Chapter Assoc. Comp. Linguistics (NAACL-HLT)*.