Top-k Link Recommendation in Social Networks

Dongjin Song
Department of ECE
University of California San Diego
La Jolla, USA, 92093-0409
Email: dosong@ucsd.edu

David A. Meyer
Department of Mathematics
University of California San Diego
La Jolla, USA, 92093-0112
Email: dmeyer@math.ucsd.edu

Dacheng Tao
Centre for Quantum Comp. & Intelligent Sys.
Faculty of Engineering and IT
University of Technology, Sydney
81 Broadway Street, Ultimo, NSW 2007, Australia.
Email: dacheng.tao@uts.edu.au

***Abstract*—Inferring potential links is a fundamental problem in social networks. In the link recommendation problem, the aim is to suggest a list of potential people to each user, ordered by the preferences of the user. Although various approaches have been developed to solve this problem, the difficulty of producing a ranking list with high precision at the top—the most important consideration for real world applications—remains largely an open problem. In this work, we propose two top-k link recommendation algorithms which focus on optimizing the top ranked links. For this purpose, we define a cost-sensitive ranking loss which penalizes the mistakes at the top of a ranked list more than the mistakes at the bottom. In particular, we propose a *log* loss, derive its surrogate, and formulate a top-k link recommendation model by optimizing this surrogate loss function based upon latent features. Moreover, we extend this top-k link recommendation model by incorporating both the latent features and explicit features of the network. Finally, an efficient learning scheme to learn the model parameters is provided. We conduct empirical studies based upon four real world datasets, *i.e.*, Wikipedia, CondMat, Epinions, and MovieLens 1M, of which the largest network contains more than 70 thousand nodes and over one million links. Our experiments demonstrate that the proposed algorithms outperform several state-of-the-art methods.**

***Keywords*-Link recommendation, cost-sensitive ranking loss, top-k learning to rank, social networks.**

I. Introduction

In the past decade, social networks, *e.g.*, Facebook, Twitter, LinkedIn, *etc.*, have become increasingly common and have dramatically reshaped people's social lives. Thus a considerable amount of effort has been devoted to investigating their underlying social mechanisms for the purpose of enhancing user experience. Link prediction [15, 16] and link recommendation [4] are two fundamental problems, solutions to which can help users connect to people they are interested in. Specifically, given a set of potential people, link prediction is actually a binary classification problem indicating the presence or absence of links to these people with either explicit network topological structure [9, 1, 15] (*e.g.*, common friends) or latent features [15, 16]. Link recommendation, which casts the same problem as a personalized ranking problem, suggests a list of people to each user with whom the user might create new connections; in the ranked list, people are recommended in decreasing order of ranking scores (which estimate the user's preferences).

In this paper, we focus on link recommendation, which has been studied in many previous works [9, 1, 15, 17, 32, 16, 4, 8]. These approaches, in general, fall into two categories, *i.e.*, explicit network topological feature-based approaches and latent feature-based approaches. Given a partially observed social network, the first recommends links based upon explicit network topological features such as common neighbors, Jaccard coefficients [15], Katz index [9], or Adamic and Adar similarity [1]. These approaches, however, cannot perform well when little topological information is available for nodes. The second recommends links based upon latent features extracted from the network. For instance, Rendle et al. [17] and Menon et al. [16] employ low rank approximation as the underlying model to extract latent features and produce ranking scores by optimizing areas under receiver operating characteristic curves (AUC) which treats each pairwise comparison equally.

Although various approaches have demonstrated their usefulness for link recommendation, few of them are specifically designed for producing a ranking list with high precision at the top—the most important consideration for practical applications. In this paper, we cast this aim as the top-k link recommendation problem. Intuitively, top-k link recommendation is combinatorially difficult because given each user and a positive integer k, we aim to select a subset of k people (in descending order of likelihood) from all people with whom this user may connect, such that the precision at the top k positions is maximized.

To tackle this problem, we propose two top-k link recommendation algorithms which focus on optimizing the top ranks. Rather than minimizing the expected 0-1 loss defined between the top-k predicted and ground truth lists [29], we define a cost-sensitive ranking loss which penalizes the mistakes at the top of a ranked list more than the mistakes at the bottom. In particular, we propose a *log* loss, derive its surrogate form, and formulate a top-k link recommendation algorithm by optimizing this surrogate loss function based upon latent features. Moreover, we extend this top-k link recommendation model by incorporating both the latent features and explicit features of the network. Finally, we employ an efficient learning scheme to learn model param-

eters. To study the effectiveness of the proposed algorithm, we compare the proposed approaches with five different baseline methods based upon four real world datasets, *i.e.*, Wikipedia, CondMat, Epinions, and MovieLens 1M. The largest network has more than 70 thousand nodes and one million links. Our experimental results demonstrate that our proposed top-k link recommendation algorithms outperform state-of-the-art methods.

The rest of this paper is organized as follows: in Section 2, we describe related work. In Section 3, we state the problem we aim to study and present a cost-sensitive ranking loss. In Section 4, we develop two algorithms to perform top-k link recommendation based upon the cost-sensitive ranking loss. We show the experimental results in Section 5, and conclude our work in Section 6.

II. Related Work

Existing methods for link recommendation fall into two main categories, *i.e.*, explicit network topological feature-based approaches and latent feature-based approaches.

Explicit network topological feature-based approaches recommend links based upon explicit network topological structures such as the neighborhoods of nodes or the ensemble of paths. The neighborhoods of nodes-based methods include common neighbors [15] and Jaccard's coefficient [20]; the ensemble of path-based methods, *e.g.*, Katz index [9], PageRank [15], and supervised random walks [4], produce ranking scores by considering the ensemble of all paths between two nodes. These approaches, however, cannot perform well when little topological information is available for nodes (*i.e.*, for example, when a node has few direct connections or high-order connections to other nodes). To handle this situation, latent feature-based approaches are developed.

Latent feature-based approaches recommend links by extracting latent features to recover the values, or the relative ordering of the values, of entries in the (weighted) adjacency matrix associated with a social network. In particular, there are three types of latent feature-based approaches. Pointwise methods [19, 11, 12, 7] treat link recommendation as a matrix completion problem and reconstruct the adjacency matrix of a partially observed social network with a low rank matrix factorization model. Pairwise methods treat link recommendation as a learning to rank problem based upon pairwise comparisons [17, 16, 13] of two triplets (*i.e.*, (i, j, score(i, j)) and (i , k, score(i, k)), where i denotes i-th row (user) of the data matrix, j and k are columns (users) of the data matrix)). Listwise methods [6, 30, 29] were originally developed for information retrieval, but they can be adapted to perform link recommendation [21]. In particular, they aim to learn a ranking loss function by taking individual lists as instances and minimizing the expected 0-1 loss function defined on the predicted list and the ground truth list.

In this paper, we focus on solving the personalized top-k link recommendation problem in social networks rather than determining the top-k influential nodes/links from the perspective of information diffusion [10, 26, 3]. One related problem is top-k recommendation in collaborative filtering which aims to find a few specific items which are assumed to be most appealing to the user. Recent advances [7, 31] have shown that root mean square error (RMSE) is not a natural fit for evaluating top-k recommendations. This is because pointwise methods do not directly model the relative order of the values in the adjacency matrix. Another related problem is the top-k learning to rank problem for information retrieval. Recent work [29, 2] focuses on optimizing an expected 0-1 loss function defined on the top-k predicted and ground truth lists. In real world applications, however, this loss could be cost-sensitive, *i.e.*, it can depend on the positions of incorrectly ranked objects. In this paper, therefore, we define a cost-sensitive ranking loss function, *i.e.*, *log* loss, which penalizes the mistakes at the top of a ranked list more than the mistakes at the bottom, and derive its surrogate form for real world applications. Then, we formulate two link recommendation algorithms by optimizing the proposed surrogate loss function.

Note that the proposed cost-sensitive ranking loss differs from the ranking error function in [27, 28] since it is a continuous and differentiable objective and thus gradient based approaches such as Newton's method and stochastic gradient method can be utilized for optimization. The ranking error function in [27, 28], however, is a discrete and non-differentiable objective which is difficult to optimize directly. Although similar ideas have been used in completely different contexts, such as information retrieval [33, 34] and image annotation [28], to the best of our knowledge, our work is the first one to conduct link recommendation with a newly developed cost-sensitive ranking loss function.

III. Cost-sensitive ranking loss

In this section, we first state the problem we aim to study. Then, we present the cost-sensitive ranking loss which penalizes the mistakes at the top of a ranked list more than the mistakes at the bottom.

Notations: Let $X \in \mathbb{R}^{n\times n}$ be an n by n matrix; we use $X^i \in \mathbb{R}^{1\times n}$ to denote its i-th row and use $X_j \in \mathbb{R}^n$ to represent its j-th column; we use X_{ij} to denote the entry in its i-th row and j-th column; we utilize $\boldsymbol{g}_{ij} \in \mathbb{R}^d$ to denote a d dimensional explicit feature vector which is extracted from the i-th row and j-th column of the adjacency matrix.

A. Problem statement

Given a partially observed social network, we use $X_{train} \in \{1, ?\}^{n\times n}$ to denote its adjacency matrix, where 1 denotes a known existing link, and ? denotes an unknown status link. In the training stage, we treat ? as zero and aim to learn a mapping function (encoding the preference score) from

user i to user j, *i.e.*, $f(X_{train}, i, j) = \widehat{X}_{ij}$, which produces a ranking score ($\widehat{X}_{ij} \in \mathbb{R}$) to recover the relative ordering of the entries in X_{train}. In the test phase, we evaluate whether the top ranked unknown status links (?) are true links based upon $\widehat{X}$ and the ground truth $X_{test} \in \{1, 0\}^{n \times n}$. We assume X_{train} and X_{test} are sparse matrices in this work because real world networks tend to be very sparse. For simplicity, we use X to denote X_{train} in the following.

B. Cost-sensitive ranking loss

To better explain our newly proposed cost-sensitive ranking loss, we compare it with the top-k true loss in [29]. In particular, let *sorted* X^i denote the ordered X^i with existing links ranked at the top and let *sorted* $\widehat{X}^i$ denote $\widehat{X}^i$ in descending order. The top-k true loss for user i is defined as:

$$L_{true}(\textit{sorted } X^i, \textit{sorted } \widehat{X}^i) = \begin{cases} 0, & \text{if } (\textit{sorted } \widehat{X}^i)_s = (\textit{sorted } X^i)_s \ (\forall\ 1 \le s \le k) \\ 1, & \text{otherwise,} \end{cases} \tag{1}$$

where s is the index of *sorted* $\widehat{X}^i$ (or *sorted* X^i) and k is determined by specific applications. The above true loss indicates that if the permutation of the top-k predicted results is exactly the same as the permutation in the top-k positions of ground truth, then the loss is zero; otherwise the loss is one. For real world applications, the loss can be cost-sensitive, *i.e.*, it can depend on the positions of incorrectly ranked objects. Therefore, to optimize the precision at the top positions of a ranking list, we develop a cost-sensitive ranking loss which penalizes the mistakes at the top of a ranked list more than the mistakes at the bottom.

Given an existing link from user i to person j within a social network X, the *rank* for this link is determined by:

$$R(X_{ij}, \widehat{X}^i) = I(X_{ij} = 1) \cdot \sum_{s=1}^{n} I(\widehat{X}_{ij} \le \widehat{X}_{is}) I(X_{is} = 0), \tag{2}$$

where $I(\cdot)$ is an indicator function which is one if the condition in the parenthesis is satisfied and is zero otherwise. For user i, given any existing link $X_{ij} = 1$, ideally, it should be ranked on top of all unknown status links (*i.e.*, $X_{is} = 0$). Therefore, *rank* measures the number of incorrectly ranked unknown status links before this particular link.

Based upon the *rank* we defined, the cost-sensitive ranking loss over the ground truth ranking list (X^i) and the predicted ranking list ($\widehat{X}^i$) for user i is given as:

$$L\big(R(X_{ij}, \widehat{X}^i)\big) = \log\big(1 + R(X_{ij}, \widehat{X}^i)\big), \tag{3}$$

where $L(\cdot)$ is essentially a transformation function which weights the mistakes at the top of a ranked list more than the mistakes at the bottom as shown in Figure 1. $L^{'}(\textit{rank}) > 0$ preserves the intuitive understanding of *rank*. To capture the greater importance of lower ranks, we should expect $L^{'}(\textit{rank})$ to be decreasing, *i.e.*, $L^{''}(\textit{rank}) \le 0$.

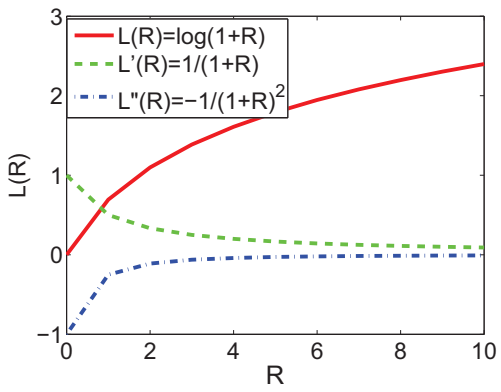


Figure 1: The loss function $L(R)$, its gradient $L'(R)$, and second-order gradient $L''(R)$.

IV. Model and Optimization

In this section, we present two link recommendation approaches, *i.e.*, the top-k link recommendation with latent features and the top-k link recommendation with both latent features and explicit features, based upon the cost-sensitive ranking loss we developed. We also introduce a stochastic gradient method to learn model parameters.

Since *rank* in Eq. 2 is non-continuous and non-differentiable with respect to the ranking score $\widehat{X}$, the loss function in Eq. 3 is also non-continuous and non-differentiable. To address this issue, we use the sigmoid function $g(z) = 1/(1 + \exp(-z))$ to approximate the indicator function in *rank*, *i.e.*,

$$\widehat{R}(X_{ij}, \widehat{X}^i) = I(X_{ij} = 1) \cdot \sum_{s=1}^{n} g(\widehat{X}_{is} - \widehat{X}_{ij}) I(X_{is} = 0) \tag{4}$$

which is defined as the *approximated rank*. By replacing $R(\cdot)$ with $\widehat{R}(\cdot)$, the surrogate loss function of $L(X_{ij}, \widehat{X}^i)$ can be written as $\phi(X_{ij}, \widehat{X}^i)$.

A. Top-k link recommendation with latent features

Based upon the surrogate loss function, *i.e.*, $\phi(X_{ij}, \widehat{X}^i)$, we develop a novel top-k link recommendation algorithm to perform link recommendation in social networks. In real world applications, social networks tend to be very large and sparse, so directly seeking for an $n \times n$ parameter matrix $\widehat{X}$ will be both computationally and memory inefficient. To tackle this issue, we approximate $\widehat{X}$ with two low rank matrices $U \in \mathbb{R}^{r \times n}$ and $V \in \mathbb{R}^{r \times n}$ and our aim can be recast as learning the following ranking score function:

$$\widehat{X}_{ij} = U_i^T V_j, \tag{5}$$

such that the precision of the top k positions of the ranking list is maximized. Note that U_i and V_j are the two latent user feature vectors, respectively. $r \ll n$ is the matrix rank.

When X is a symmetric network, we can set $U = V$ in Eq. 5 for simplicity.

With this underlying ranking score function, the objective function of the top-k link recommendation algorithm can be formulated as

$$\begin{aligned}&\mathcal{O}(U,V)\\ =&\sum_{i=1}^{n}\sum_{j=1}^{n}\phi(X_{ij},\widehat{X}^i)+\frac{\lambda}{2}\sum_{i=1}^{n}U_i^TU_i+\frac{\mu}{2}\sum_{j=1}^{n}V_j^TV_j\\ =&\sum_{i=1}^{n}\sum_{j=1}^{n}\phi\Big(\widehat{R}(X_{ij},\widehat{X}^i)\Big)+\frac{\lambda}{2}\sum_{i=1}^{n}U_i^TU_i+\frac{\mu}{2}\sum_{j=1}^{n}V_j^TV_j,\end{aligned} \tag{6}$$

where the first term is a surrogate loss function and the other two terms are regularization terms to avoid over-fitting. $\lambda > 0$ and $\mu > 0$ are two hyper-parameters for controlling the trade off between the surrogate loss function and the regularization terms.

A local minimum of the objective in Eq. 6 can be achieved by performing gradient descent over U and V iteratively. In particular, the gradient of $\mathcal{O}(U,V)$ with respect to U_i is given as

$$\begin{aligned}\frac{\partial\mathcal{O}(U,V)}{\partial U_i}=&\sum_{j=1}^{n}\frac{1}{1+\widehat{R}(X_{ij},\widehat{X}^i)}\sum_{s=1}^{n}g(\widehat{X}_{is}-\widehat{X}_{ij})\cdot\\ &g(-\widehat{X}_{is}+\widehat{X}_{ij})\cdot(V_s-V_j)\cdot I(X_{ij}=1)\cdot\\ &I(X_{is}=0)+\lambda U_i,\end{aligned} \tag{7}$$

the gradient of $\mathcal{O}(U,V)$ with respect to V_j is given as

$$\begin{aligned}\frac{\partial\mathcal{O}(U,V)}{\partial V_j}=&\sum_{i=1}^{n}\frac{1}{1+\widehat{R}(X_{ij},\widehat{X}^i)}\sum_{s=1}^{n}g(\widehat{X}_{is}-\widehat{X}_{ij})\cdot\\ &g(-\widehat{X}_{is}+\widehat{X}_{ij})\cdot(-U_i)\cdot I(X_{ij}=1)\cdot\\ &I(X_{is}=0)+\mu V_j,\end{aligned} \tag{8}$$

and the gradient of $\mathcal{O}(U,V)$ with respect to V_s is given as

$$\begin{aligned}\frac{\partial\mathcal{O}(U,V)}{\partial V_s}=&\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{1}{1+\widehat{R}(X_{ij},\widehat{X}^i)}g(\widehat{X}_{is}-\widehat{X}_{ij})\cdot\\ &g(-\widehat{X}_{is}+\widehat{X}_{ij})\cdot U_iI(X_{ij}=1)\cdot\\ &I(X_{is}=0)+\mu V_s.\end{aligned} \tag{9}$$

One disadvantage of the top-k link recommendation algorithm is that it does not consider explicit network topological features, *e.g.*, node degree, common neighbors, *etc*. To overcome this issue, we extend the top-k link recommendation algorithm to consider both the latent features and the explicit features of social networks.

B. Top-k link recommendation with both latent features and explicit features

Although various link recommendation algorithms have been developed based upon either latent features or explicit network topological features of social networks, few of them incorporates both of them. With the intuition that both of them provide confidence to recover the relative ordering of the values, of entries in the (weighted) adjacency matrix associated with a social network, we extend top-k link recommendation with the following model:

Algorithm 1 Top-k link recommendation algorithm with latent features.

Input: X, λ, μ, α, T (maximum iteration).
Initialize: set $t = 0$, initialize U and V randomly.
repeat
  $t = t + 1$
  Randomly pick up an observed link $X_{ij} = 1$;
  Fix user i, uniformly draw b unknown status links $X_{is} = 0$;
  if $\exists\ s$ such that $U_i^TV_j < U_i^TV_s$
    Calculate $\frac{\partial\mathcal{O}(U,V)}{\partial U_i}$ based upon Eq. 7;
    Calculate $\frac{\partial\mathcal{O}(U,V)}{\partial V_j}$ with Eq. 8;
    Calculate $\frac{\partial\mathcal{O}(U,V)}{\partial V_s}$ with Eq. 9;
    $U = U - \alpha\frac{\partial\mathcal{O}(U,V)}{\partial U}$;
    $V = V - \alpha\frac{\partial\mathcal{O}(U,V)}{\partial V}$;
  end
until validation error does not improve or $t > T$.

$$\widehat{X}_{ij}=U_i^TV_j+\boldsymbol{g}_{ij}^T\boldsymbol{\theta}, \tag{10}$$

where U_i and V_j are the two latent user feature vectors, respectively; $\boldsymbol{g}_{ij}$ is a 3-dimensional explicit feature vector which encodes the degree of node i, the degree of node j, and the number of common neighbors of node i and node j; $\boldsymbol{\theta} \in \mathbb{R}^3$ is parameter of explicit features.

With the above model, the objective of extended top-k link recommendation can be written as:

$$\begin{aligned}\overline{\mathcal{O}}(U,V,\boldsymbol{\theta})=&\sum_{i=1}^{n}\sum_{j=1}^{n}\phi\Big(\widehat{R}(X_{ij},\widehat{X}^i)\Big)+\frac{\lambda}{2}\sum_{i=1}^{n}U_i^TU_i\\ &+\frac{\mu}{2}\sum_{j=1}^{n}V_j^TV_j+\frac{\gamma}{2}\boldsymbol{\theta}^T\boldsymbol{\theta},\end{aligned} \tag{11}$$

where the first term is the cost-sensitive ranking loss, the second, third, and last terms are regularization terms. $\lambda > 0$, $\mu > 0$, and $\gamma > 0$ are three hyper-parameters to control the trade-off between the surrogate loss function and other terms.

A local minimum of the objective in Eq. 11 can be achieved by performing gradient descent over U, V, and $\boldsymbol{\theta}$, iteratively. Specifically, the gradient of $\overline{\mathcal{O}}(U,V,\boldsymbol{\theta})$ with

respect to U_i is given as

$$\frac{\partial \overline{\mathcal{O}}(U,V,\boldsymbol{\theta})}{\partial U_i} = \sum_{j=1}^{n} \frac{1}{1+\widehat{R}(X_{ij},\widehat{X}^i)} \sum_{s=1}^{n} g(\widehat{X}_{is}-\widehat{X}_{ij}) \cdot g(-\widehat{X}_{is}+\widehat{X}_{ij}) \cdot (V_s - V_j) \cdot I(X_{ij}=1) \cdot I(X_{is}=0) + \lambda U_i, \tag{12}$$

the gradient of $\overline{\mathcal{O}}(U,V,\boldsymbol{\theta})$ with respect to V_j is given as

$$\frac{\partial \overline{\mathcal{O}}(U,V,\boldsymbol{\theta})}{\partial V_j} = \sum_{i=1}^{n} \frac{1}{1+\widehat{R}(X_{ij},\widehat{X}^i)} \sum_{s=1}^{n} g(\widehat{X}_{is}-\widehat{X}_{ij}) \cdot g(-\widehat{X}_{is}+\widehat{X}_{ij}) \cdot (-U_i) \cdot I(X_{ij}=1) \cdot I(X_{is}=0) + \mu V_j, \tag{13}$$

the gradient of $\overline{\mathcal{O}}(U,V,\boldsymbol{\theta})$ with respect to V_s is given as

$$\frac{\partial \overline{\mathcal{O}}(U,V,\boldsymbol{\theta})}{\partial V_s} = \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{1}{1+\widehat{R}(X_{ij},\widehat{X}^i)} g(\widehat{X}_{is}-\widehat{X}_{ij}) \cdot g(-\widehat{X}_{is}+\widehat{X}_{ij}) \cdot U_i I(X_{ij}=1) \cdot I(X_{is}=0) + \mu V_s, \tag{14}$$

and the gradient of $\overline{\mathcal{O}}(U,V,\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ is written as

$$\frac{\partial \overline{\mathcal{O}}(U,V,\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{s=1}^{n} \frac{1}{1+\widehat{R}(X_{ij},\widehat{X}^i)} g(\widehat{X}_{is}-\widehat{X}_{ij}) \cdot g(-\widehat{X}_{is}+\widehat{X}_{ij}) \cdot (\boldsymbol{g}_{is}-\boldsymbol{g}_{ij}) \cdot I(X_{ij}=1) \cdot I(X_{is}=0) + \gamma \boldsymbol{\theta}. \tag{15}$$

C. Optimization

Although a local minimum of the objective functions given by Eq. 6 and Eq. 11 can be found by performing gradient descent over U and V (and $\boldsymbol{\theta}$) iteratively, the computational complexity for a full gradient over U (or V, $\boldsymbol{\theta}$) is around $O(pnr)$ where $p \ll n^2$ is the number of existing links. In real world applications, p can be as large as one million or even one billion. In this case, calculating the full gradient will be too slow or even infeasible. Therefore, a stochastic learning scheme which minimizes $\phi\Big(\widehat{R}(X_{ij},\widehat{X}^i)\Big)$ for each given existing link is used. In this case, the computational complexity for a gradient over U_i (or V_j, $\boldsymbol{\theta}$) for each given existed link is only around $O(nr)$. To further reduce the complexity, given each observed link, we can randomly sample a subset of unknown status links (*e.g.*, $b \ll n$). If at least one violation (*i.e.*, $U_i^T V_j < U_i^T V_s$ or $U_i^T V_j + \boldsymbol{g}_{ij}^T \boldsymbol{\theta} < U_i^T V_s + \boldsymbol{g}_{is}^T \boldsymbol{\theta}$) is found in this subset, we can approximate $\phi\Big(\widehat{R}(X_{ij},\widehat{X}^i)\Big)$ with $\phi\Big(\frac{n}{b} I(X_{ij}=1) \sum_{s=1}^{b} g(\widehat{X}_{is}-\widehat{X}_{ij}) I(X_{is}=0)\Big)$ by assuming that violations are uniformly distributed and conduct gradient descent over U and V (and $\boldsymbol{\theta}$) subsequently.

Algorithm 2 Top-k link recommendation algorithm with both latent features and explicit features.

Input: X, λ, μ, α, T (maximum iteration).
Initialize: set $t=0$, initialize U and V randomly.
repeat
  $t = t+1$
  Randomly pick up an observed link X_{ij};
  Fix user i, uniformly draw b unknown status links $X_{is}=0$;
  if $\exists\ s$ such that $U_i^T V_j + \boldsymbol{g}_{ij}^T \boldsymbol{\theta} < U_i^T V_s + \boldsymbol{g}_{is}^T \boldsymbol{\theta}$
    Calculate $\frac{\partial \overline{\mathcal{O}}(U,V,\boldsymbol{\theta})}{\partial U_i}$ based upon Eq. 12;
    Calculate $\frac{\partial \overline{\mathcal{O}}(U,V,\boldsymbol{\theta})}{\partial V_j}$ with Eq. 13;
    Calculate $\frac{\partial \overline{\mathcal{O}}(U,V,\boldsymbol{\theta})}{\partial V_s}$ with Eq. 14;
    Calculate $\frac{\partial \overline{\mathcal{O}}(U,V,\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$ with Eq. 15;
    $U = U - \alpha \frac{\partial \overline{\mathcal{O}}(U,V,\boldsymbol{\theta})}{\partial U}$;
    $V = V - \alpha \frac{\partial \overline{\mathcal{O}}(U,V,\boldsymbol{\theta})}{\partial V}$;
    $\boldsymbol{\theta} = \boldsymbol{\theta} - \alpha \frac{\partial \overline{\mathcal{O}}(U,V,\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$;
  end
until validation error does not improve or $t > T$.

Table I: The detailed statistics of four datasets. Note that MovieLens 1M is a bipartite network which has 6040 users and 3592 items.

Dataset	Wikipedia	CondMat	Epinions	MovieLens 1M
Nodes	7,115	23,133	75,879	6040/3592
Links	103,689	186,994	508,837	1,000,209
Links: Non-links	1:488	1:2,929	1:11,317	1:20.70
Average degree	14.57	7.89	6.70	165.59
Type	Directed	Undirected	Directed	Directed

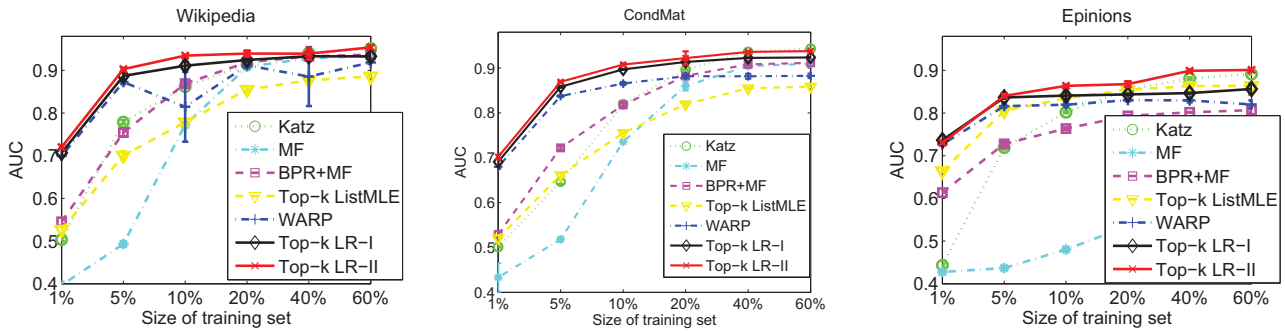
In practical applications, we stop updating U and V (and $\boldsymbol{\theta}$) until it exceeds the maximum iteration or the validation metric (*e.g.*, MAP) does not improve. The detailed optimization procedure for top-k link recommendation with latent features is provided in Algorithm 1 and the detailed optimization procedure for extended top-k link recommendation with both latent features and explicit features is provided in Algorithm 2.

V. Experiments

To demonstrate the effectiveness of the proposed top-k link recommendation algorithms, we compare them with various baseline approaches based upon two different tasks, *i.e.*, link recommendation and collaborative ranking (which can be seen as a generalized link recommendation problem with multiple scales), over four publicly available datasets.

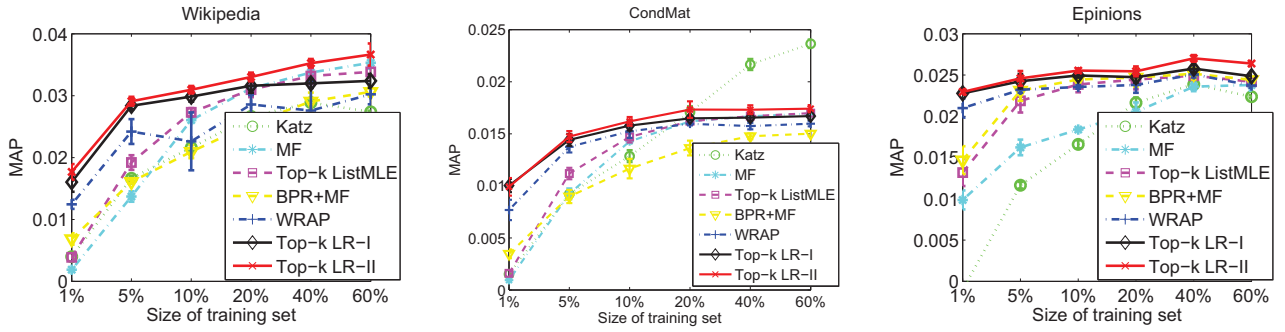
A. Datasets

For the link recommendation task, we consider two well-known directed social networks, *i.e.*, Wikipedia [5] and



(a) AUC on Wikipedia (b) AUC on CondMat (c) AUC on Epinions

Figure 2: (a) to (c) are AUC obtained on Wikipedia, CondMat, and Epinions when the size of training set varies from 1%, 5%, 10%, 20%, 40%, to 60%. Error bars denote the standard deviations.



(a) MAP on Wikipedia (b) MAP on CondMat (c) MAP on Epinions

Figure 3: (a) to (c) are MAP obtained on Wikipedia, CondMat, and Epinions when the size of training set varies from 1%, 5%, 10%, 20%, 40%, to 60%. Error bars denote the standard deviations.

Epinions [18], as well as one undirected social network, *i.e.*, CondMat [14] [1].

The Wikipedia data comprise a voting network for promoting candidates to the role of admin. The voters, half coming from existing admins and another half coming from ordinary Wikipedia users, can indicate a vote with respect to the promotion of a candidate. Epinions, which is a product review website, is a trust network in which users can indicate whether they trust each other based upon their reviews. CondMat is a collaboration network from the e-print arXiv and covers scientific collaborations between authors' papers submitted to Condensed Matter category.

For collaborative ranking, we consider MovieLens 1M dataset[2] which consists of 6040 users as well as 3952 items. In this dataset, each rating value is a integer which ranges from 1 to 5. The detailed statistics of these four datasets are provided in Table I.

[1]These datasets are available online at `http://snap.stanford.edu/data/`.

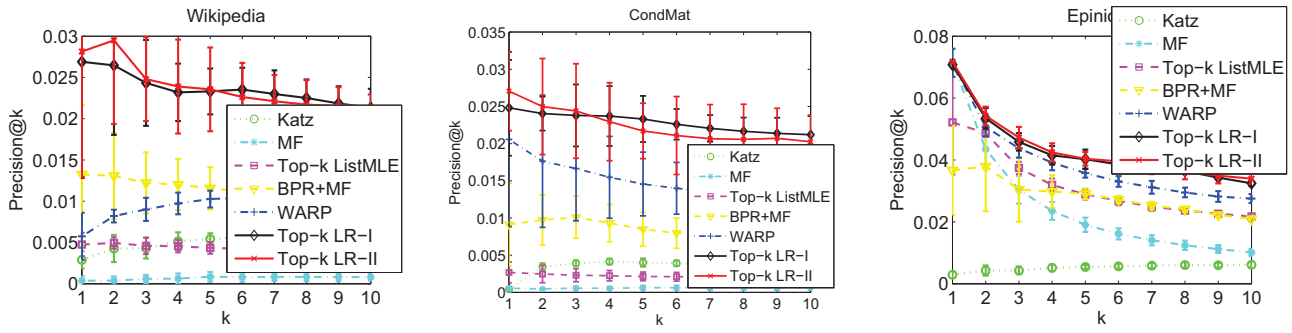
[2]This dataset is available online at `http://grouplens.org/datasets/movielens/`.

B. Evaluation

For link recommendation, given a fully observed social network $X \in \{0,1\}^{n\times n}$, we keep a fraction (*i.e.*, 1%, 5%, 10%, 20%, 40%, and 60%) of observed links (as X_{train}) for training, 10% of the observed links for validation (X_{vali}), and evaluate on a test set (X_{test}) comprising 30% of the observed links. We define the zero entries in X_{train} as unknown status links (?) since each zero entry has the potential to be either a link or non-link.

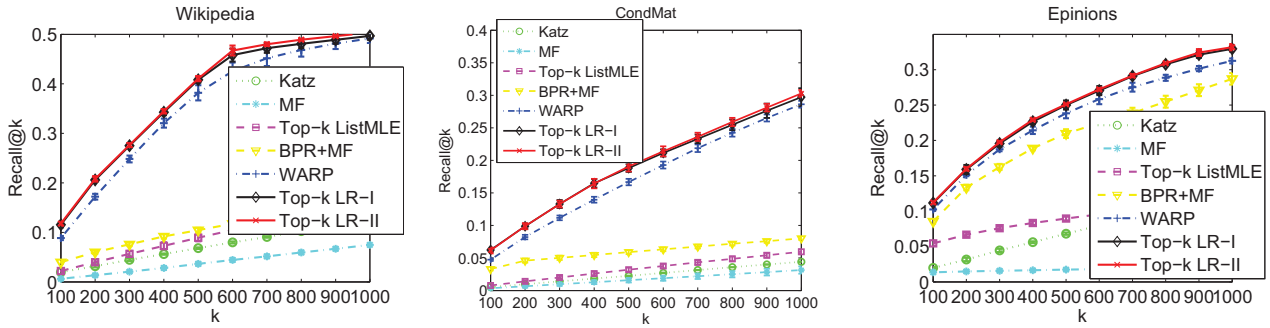
For each user with minimum node degree 3, we evaluate the link recommendation performance based upon area under the receiver operating characteristic (ROC) curve (AUC), average precision (AP), the precision of unknown existing links at the top-k positions (*i.e.*, Precision@k (P@k)), and the recall of unknown existing links at the top-k positions (*i.e.*, Recall@k). For all users with minimum node degree 3, average AUC, mean average precision (MAP), average P@k, and average Recall@k are employed for comparison.

Similarly, for collaborative ranking, given a fully observed bipartite network $X \in \{0,1,2,3,4,5\}^{m\times n}$, we keep a fraction (*i.e.*, 20%, 40%, and 60%) of observed non-zero entries (as X_{train}) for training, 10% of the observed non-zero entries for validation (X_{vali}), and evaluate on a test set (X_{test}) comprising 30% of the observed non-zero entries. The



(a) Precision@k on Wiki (1%) (b) Precision@k CondMat (1%) (c) Precision@k Epinions (1%)

Figure 4: Precision@k for Wikipedia, CondMat, and Epinions when the size of training set is 1%. Error bars denote the standard deviations.



(a) Recall@k on Wiki (1%) (b) Recall@k CondMat (1%) (c) Recall@k Epinions (1%)

Figure 5: Recall@k for Wikipedia, CondMat, and Epinions when the size of training set is 1%. Error bars denote the standard deviations.

difference is that we evaluate how the non-zero entries are ranked based upon normalized discounted cumulative gain (NDCG) at top 10 positions, *i.e.*, NDCG@10.

C. Baseline algorithms and parameter setting

To demonstrate the effectiveness of the proposed top-k link recommendation algorithms, we compare the following seven methods:

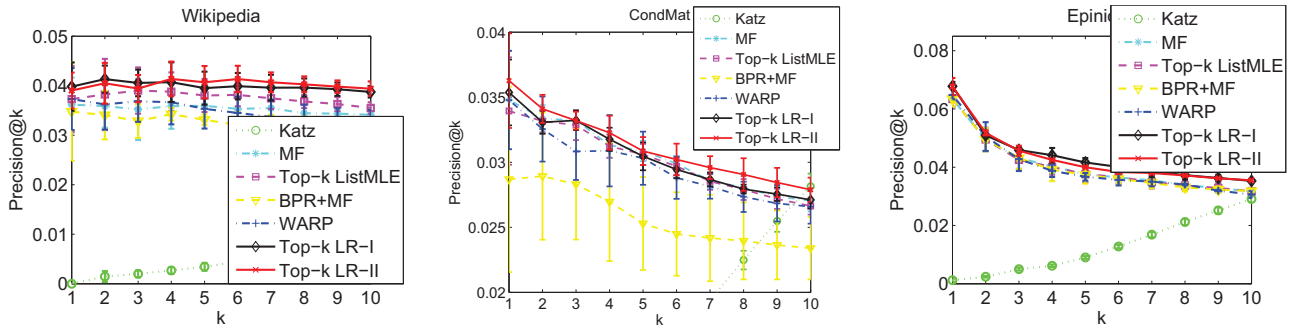
- Katz index [9] is a representative network topological approach; the parameter β is determined by search over the grid $\{0.001, 0.005, 0.01, 0.05, 0.1, 0.5\}$;
- Matrix factorization (MF) [12] is a pointwise method;
- Bayesian personalized ranking based matrix factorization (BPR+MF) [17] is a representative pairwise method;
- Adapted top-k listwise maximum likelihood estimation (top-k ListMLE) based upon the top-k likelihood loss [29] and matrix factorization;
- Weighted approximated-rank approximation (WARP) [28] based matrix factorization;
- Top-k LR-I: top-k link recommendation with latent features;
- Top-k LR-II: top-k link recommendation with both latent features and explicit features.

For MF, BPR+MF, and top-k ListMLE, the hyper-parameters for regularization terms are determined by searching over the grid $\{1, 5, 10, 20, 50, 100, 200\}$ and the matrix rank r is determined by searching over the grid $\{5, 10, 30, 50, 70, 90\}$. For WARP, top-$k$ LR-I, and top-k LR-II, we set $\lambda = \mu = \gamma$ for simplicity and it is optimized by searching over the grid $\{10^{-5}, 5 \times 10^{-5}, 10^{-4}, 5 \times 10^{-4}, 10^{-3}, 5 \times 10^{-3}, 10^{-2}, 5 \times 10^{-2}, 10^{-1}\}$ and the matrix rank r is determined over the same set as for MF, BPR+MF, and top-k ListMLE for fair comparison; the parameters which achieve the best performance over the validation set (X_{vali}) are used for test. We conduct five trials for each dataset and report the average performance and their associated standard deviations.

For collaborative ranking, Katz index cannot work because MovieLens 1M is a bipartite network. The parameter settings for MF, WARP, top-k ListMLE, BPR+MF, and top-k LR-I are identical as in link recommendation.

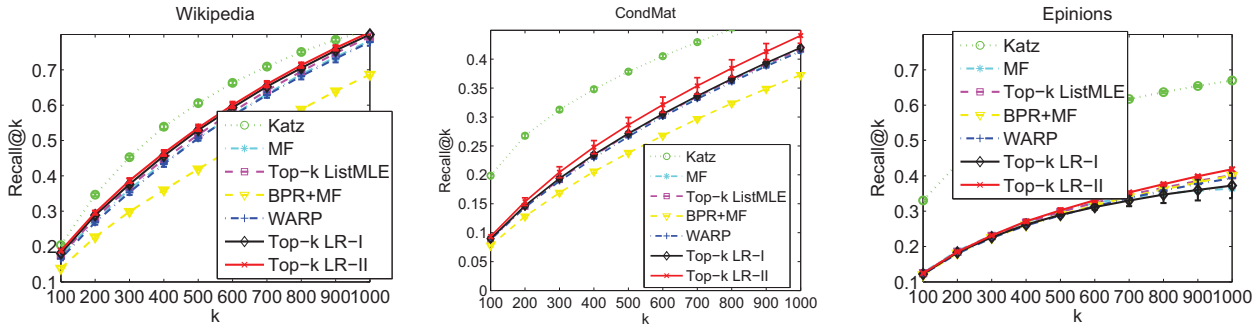
D. Task-I: link recommendation

As shown in Figure 2(a) to 2(c) and Figure 3(a) to 3(c), link recommendation performance is evaluated with AUC and MAP when the size of each dataset varies from 1%, 5%, 10%, 20%, 40% to 60%. We observe that top-k LR-I



(a) P@k on Wiki (20%) (b) P@k CondMat (20%) (c) P@k Epinions (20%)

Figure 6: Precision@k over Wikipedia, CondMat, and Epinions when the size of training set is 20%. Error bars denote the standard deviations.



(a) Recall@k on Wiki (20%) (b) Recall@k CondMat (20%) (c) Recall@k Epinions (20%)

Figure 7: Recall@k over Wikipedia, CondMat, and Epinions when the size of training set is 20%. Error bars denote the standard deviations.

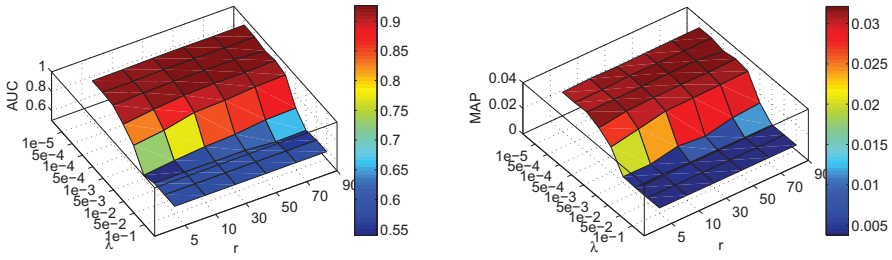
and top-k LR-II generally achieve better AUC and MAP than baseline algorithms for Wikipedia, CondMat, and Epinions. This is because both top-k LR-I and top-k LR-II focus on optimizing the precisions at the top positions of a ranking list. In particular, top-k LR-II consistently outperforms top-k LR-I since it incorporates explicit features of the networks which is extremely helpful when networks become less sparse (*i.e.*, the size of training set increases). Besides, we observe that Katz index outperforms other latent feature based approaches over CondMat and Epinions (regarding MAP) when the size of training set is over 20%. This is because CondMat is a collaboration network and Epinions is a trust network, both of them contain relatively rich high-order relationships (when the networks are relatively dense) which can be captured by the Katz index. Latent feature approaches, however, do not explicitly model high-order relationships in these two networks.

To further investigate the top-k link recommendation performance, we report the AUC and MAP associated Precision@k and Recall@k in Figure 5 and 6, 7 and 8, respectively. On these three datasets, we observe that top-k LR-I and top-k LR-II consistently outperform baseline approaches for Precision@k and Recall@k when k varies. This is because: (1) little high-order relationship is available for Katz; (2) BPR+MF is optimizing AUC, rather than the precision@k; (3) Top-k ListMLE is optimizing the expected 0-1 loss defined between the top-k predicted and ground truth lists, rather than a cost-sensitive loss; (4) the objective of WRAP is non-smooth and the optimization is limited for only one violation per step which may make the gradient direction inaccurate.

We test the parameter sensitivity of top-k LR-I when $\lambda = \mu \in \{10^{-5}, 5\times10^{-5}, 10^{-4}, 5\times10^{-4}, 10^{-3}, 5\times10^{-3}, 10^{-2}, 5\times10^{-2}, 10^{-1}\}$ and matrix rank $r \in \{5, 10, 30, 50, 70, 90\}$. In Figure 8(a) and Figure 8(b), we observe that the AUC and MAP values are very stable when k varies from 5 to 90 and $\lambda = \mu$ varies from 0.00001 to 0.005. We also evaluate the parameter sensitivity of top-k LR-II in Figure 9(a) and Figure 9(b), we notice that top-k LR-II is even more stable than top-k LR-I since it can achieve better AUC and MAP in a wider range of parameter space.

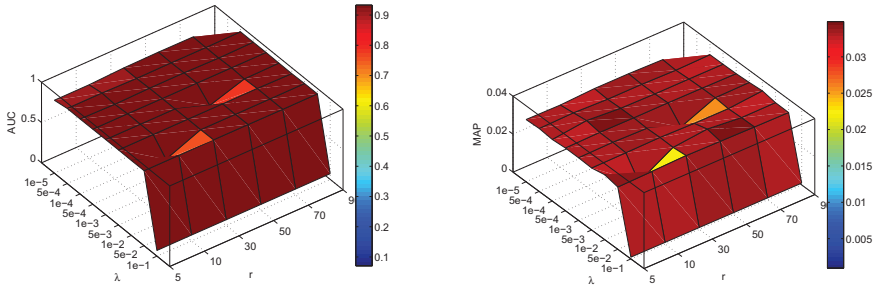
E. Task-II: collaborative ranking

Collaborative ranking can be seen as a generalized link recommendation problem to perform link recommendation at each scale (level). For instance, given a rating value 5, we would like to rank it above rating values of 1 to 4, and given



(a) Top-k LR-I: AUC on Wikipedia (20%) (b) Top-k LR-I: MAP on Wikipedia (20%)

Figure 8: AUC and MAP of Top-k LR-I for Wikipedia when the size of training set is 20% and the hyper-parameter λ and r varies.



(a) Top-k LR-II:AUC on Wikipedia (20%) (b) Top-k LR-II: MAP on Wikipedia (20%)

Figure 9: AUC and MAP of Top-k LR-II for Wikipedia when the size of training set is 20% and the hyper-parameter λ and r varies.

Table II: Average NDCG@10 and associated standard deviations for MovieLens 1M dataset when 20%, 40%, and 60% non-zero entries are used for training

Methods/Setting	20%	40%	60%
MF	0.7112($\pm$0.0013)	0.7262($\pm$0.0007)	0.7456($\pm$0.0014)
Top-k ListMLE	0.6906($\pm$0.0210)	0.6990($\pm$0.0008)	0.7204($\pm$0.0102)
BPR+MF	0.6864($\pm$0.0014)	0.6930($\pm$0.0003)	0.7164($\pm$0.0008)
WRAP	0.6854($\pm$0.0073)	0.6874($\pm$0.0016)	0.7082($\pm$0.0023)
Top-k LR-I	**0.7234($\pm$0.0022)**	**0.7356($\pm$0.0026)**	**0.7582($\pm$0.0038)**

a rating value 4, we expect to rank it above rating values 1 to 3, *etc*. We compare top-k LR-I against MF, MAP, BPR+MF, and top-k ListMLE for collaborative ranking in which we aim to rank non-zero entries based upon NDCG@k. To achieve this purpose, top-k LR-I, WRAP, and BPR+MF are adapted to aggregate binary ranking at different levels (scales). In Table II, we observe that Top-k LR-I achieves the best performance when the size of training set varies from 20%, 40% to 60%. This is because aggregated Top-k LR-I aims to optimize the top precisions at each level. Top-k LR-II is not used here since explicit features (common neighbors) are unavailable.

VI. Conclusion

We proposed two top-k link recommendation algorithms based upon a newly defined cost-sensitive ranking loss which penalizes the mistakes at the top of a ranked list more than the mistakes at the bottom. In particular, we defined *log* loss, derived its surrogate form, and formulated our algorithms by optimizing the proposed surrogate losses with latent features and explicit features of the networks. Our empirical studies demonstrated the effectiveness of our approach over Wikipedia, CondMat, Epinions, and MovieLens 1M.

In the future, one potential problem is to investigate how to model social circles in social networks appropriately and how to leverage this information to conduct top-k link recommendation. Another interesting problem is to study how to utilize *log* loss to perform link recommendation in signed networks [23, 24, 25, 22].

Acknowledgement

The work of Dongjin Song and David A. Meyer was supported by the U.S. Department of Defense Minerva Research Initiative/Army under Grant W911NF-09-1-0081 and in part by the National Science Foundation-Division of Mathematical Sciences under Grant 1223137. The work of Dacheng Tao was supported by the Australian Research Council under Projects FT-130101457 and LP-140100569.

REFERENCES

[1] L. A. Adamic and E. Adar. Friends and neighbors on the web. *Social Networks*, 25:211–230, 2003.

[2] D. Agarwal and M. Gurenvich. Fast top-k retrieval for model based recommendation. In *Proceedings of the 5th ACM international conference on Web search and data mining*, pages 483–492, 2012.

[3] Y. Aytas. *Top-k Link Recommendation for Development of P2P Social Networks*. Bilkent University, Master Thesis, 2014.

[4] L. Backstrom and J. Leskovec. Supervised random walks: predicting and recommending links in social networks. In *Proceedings of the 4th ACM international conference on Web search and data mining*, pages 635–644, Hong Kong, China, 2011.

[5] M. Burke and R. Kraul. Mopping up: modeling wikipedia promotion decisions. In *Proceedings of the 2008 ACM conference on Computer supported cooperative work*, pages 27–36, 2008.

[6] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on Machine Learning*, pages 129–136, 2007.

[7] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the 4th ACM conference on Recommender systems*, pages 39–46, 2010.

[8] Y. Dong, J. Tang, S. Wu, J. Tian, N. V. Chawla, J. Rao, and H. Cao. Link prediction and recommendation across heterogeneous social networks. In *Proceedings of the 12th IEEE International Conference on Data Mining*, 2012.

[9] L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.

[10] D. Kempe, J. Kleinberg, and E. Tardos. Influential nodes in a diffusion model for social networks. In *Proceedings of International Colloquium on Automata, Languages, and Programming*, pages 1127–1138, 2005.

[11] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434, 2008.

[12] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009.

[13] J. Lee, S. Bengio, S. Kim, G. Lebanon, and Y. Singer. Local collaborative ranking. In *Proceedings of the 23th international conference on World Wide Web*, pages 85–96, 2014.

[14] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data*, 1(1), 2007.

[15] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. *Journal of American Society for Information Science and Technology*, 58(7):1019–1031, 2007.

[16] A. K. Menon and C. Elkan. Link prediction via matrix factorization. In *Proceedings of the European Conference on Machine Learning, Part II, LNAI2912*, pages 437–452, 2011.

[17] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, pages 452–461, Arlington, Virginia, United States, 2009.

[18] M. Richardson, R. Agrawal, and P. Domingos. Trust management for the semantic web. In *ISWC*, pages 351–368, 2003.

[19] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *Proceedings of the Neural Information Processing Systems*, pages 1257–1264, 2007.

[20] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.

[21] Y. Shi, M. Larson, and A. Hanjalic. Listwise learning to rank with matrix factorization for collaborative filtering. In *Proceedings of the 4th ACM conference on Recommender systems*, pages 269–272, 2010.

[22] D. Song and D. A. Meyer. Link sign prediction and ranking in signed directed social networks. *To appear in Social Networks Analysis and Mining*.

[23] D. Song and D. A. Meyer. A model of consistent node types in signed directed social networks. In *Proceedings of IEEE/ACM International Conference on Advances in Social Network Analysis and Mining*, pages 82–90, Beijing, China, 2014.

[24] D. Song and D. A. Meyer. Recommending positive links in signed social networks by optimizing a generalized AUC. In *Proceedings of 29th AAAI Conference on Artificial Intelligence*, pages 290–296, Austin, Texas, USA, 2015.

[25] D. Song, D. A. Meyer, and Dacheng Tao. Efficient latent link recommendation in signed networks. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1105–1114, Sydney, Australia, 2015.

[26] N. R. Suri and Y. Narahari. Determining the top-k nodes in social networks using the Shapley value. In *Proceedings of 7th International Conference on Autonomous Agents and Multiagent Systems*, pages 1509–1512, 2008.

[27] N. Usunier, D. Buffoni, and P. Gallinari. Ranking with ordered weighted pairwise classification. In *Proceedings of the 26th International Conference on Machine Learning*, pages 1057–1064, 2009.

[28] J. Weston, S. Bengio, and N. Usunier. WSABIE: Scaling up to large vocalulary image annotation. In *Proceedings of International Joint Conference of Artificial Intelligence*, pages 2764–2770, 2011.

[29] F. Xia, T.-Y. Liu, and H. Li. Statistical consistency of top-k ranking. In *Proceedings of Neural Information Processing Systems*, pages 2098–2106, 2009.

[30] F. Xia, T.-Y. Liu, J. Wang, W. Zhang, and H. Li. Listwise approach learning to rank - theory and algorithm. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1192–1199, 2008.

[31] X. Yang, H. Steck, Y. Guo, and Y. Liu. On top-k recommendation using social networks. In *Proceedings of the 6th ACM conference on Recommender systems*, pages 67–74, Dublin, Ireland, September, 2012.

[32] Z. Yin, M. Gupta, T. Weninger, and J. Han. A unified framework for link recommendation using random walks. In *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 152–159, 2010.

[33] Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *Proceedings of the 30th ACM SIGIR conference on Research and development in information retrieval*, pages 271–278, 2007.

[34] H. Yun, P. Raman, and S.V.N. Vishwanathan. Ranking via robust binary classification. In *Proceedings of Neural Information Processing Systems*, pages 2582–2590, 2014.