Attributed Signed Network Embedding

Suhang Wang
Arizona State University
suhang.wang@asu.edu

Charu Aggarwal
IBM T. J. Watson Research Center
charu@us.ibm.com

Jiliang Tang
Michigan State University
tangjili@cse.msu.edu

Huan Liu
Arizona State University
huan.liu@asu.edu

ABSTRACT

The major task of network embedding is to learn low-dimensional vector representations of social-network nodes. It facilitates many analytical tasks such as link prediction and node clustering and thus has attracted increasing attention. The majority of existing embedding algorithms are designed for unsigned social networks. However, many social media networks have both positive and negative links, for which unsigned algorithms have little utility. Recent findings in signed network analysis suggest that negative links have distinct properties and added value over positive links. This brings about both challenges and opportunities for signed network embedding. In addition, user attributes, which encode properties and interests of users, provide complementary information to network structures and have the potential to improve signed network embedding. Therefore, in this paper, we study the novel problem of signed social network embedding with attributes. We propose a novel framework SNEA, which exploits the network structure and user attributes simultaneously for network representation learning. Experimental results on link prediction and node clustering with real-world datasets demonstrate the effectiveness of SNEA.

CCS CONCEPTS

•Information systems →Data mining;

KEYWORDS

Signed Social Networks, Network Embedding, Node Attributes

ACM Reference format:
Suhang Wang, Charu Aggarwal, Jiliang Tang, and Huan Liu. 2017. Attributed Signed Network Embedding. In *Proceedings of CIKM'17, November 6–10, 2017, Singapore.*, , 10 pages.
DOI: https://doi.org/10.1145/3132847.3132905

1 INTRODUCTION

The increasing availability of large-scale social media networks has greatly advanced the ability to perform various mining tasks. An important task is that of network embedding, which aims at learning low-dimensional vector representations of nodes. Network embedding is one of the most central tasks in data mining, and it has been proven to be useful in many social network mining tasks such as link prediction [18], community detection [21], node classification [4] and visualization [32]. The majority of existing network embedding algorithms have been dedicated to social networks with only positive links. However, social networks can contain both positive and negative links, and these signed social networks are present on a variety of social media sites, such as Epinions with trust and distrust links, and Slashdot with friend and foe links. In addition to existing signed social networks, many algorithms are proposed to construct signed networks from positive and negative interactions between users or documents [9, 19].

The availability of negative links in signed networks causes problems in leveraging the basic principles that are commonly used for mining unsigned social networks. This is because the principles of mining signed social networks can be substantially different from those of unsigned networks [17, 28]. For example, homophily effects and social influence for unsigned networks may not be applicable to signed networks in their original form [30]. These present challenges in extending existing algorithms from the unsigned case. In a similar vein, signed network embedding cannot be easily carried out by simply extending the existing embedding algorithms for unsigned social networks. Recent research on mining signed social networks suggests that negative links have added value over positive links in various analytical tasks. For example, a small number of negative links can significantly improve positive link prediction performance [16], and they can also improve recommendation performance in social media [33]. While signed network embedding is challenging, its research results can potentially advance signed network mining tasks such as link prediction. However, the existing work on signed network embedding is rather limited. In addition, node attributes, which reveal users interests and/or properties, have been proven to be effective for learning better representations for unsigned networks [7, 46]. Thus, we are curious if node attributes can help to improve the quality of signed network embedding.

In this paper, we investigate the novel problem of signed network embedding with attributes in social media by studying the following two questions: (1) What's the relationship between user links and user attributes; and (2) how to model signed links and attributes simultaneously for learning network embeddings. To answer these two questions, we conduct data analysis on signed network with attributes and based on the findings of the analysis we propose a novel framework for *S*igned *N*etwork *E*mbedding with *A*ttributes, SNEA, which models both signed links and user attributes in a unified framework. The major contributions of the paper are summarized next:

CIKM'17, November 6–10, 2017, Singapore.

DOI: https://doi.org/10.1145/3132847.3132905

- Providing a principled way to analyze the relationship between signed links and the similarity of user attributes;
- Proposing a novel framework SNEA, which leverages both signed links and user attributes for learning network embedding; and
- Conducting experiments on real-world signed social network datasets to assess the effectiveness of SNEA.

The remainder of the paper is organized as follows. In Section 2, we review related work. In Section 3, we give a preliminary analysis of signed social networks with attributes, which lays the groundwork for SNEA. In Section 4, we introduce the details of the proposed framework SNEA. In Section 5, we present a method to solve the optimization problem of SNEA along with the time complexity analysis. In Section 6, we show empirical evaluation with discussion. In Section 7, we give conclusion with future work.

2 RELATED WORK

Network embedding aims at learning low-dimensional vector representations for nodes of a given network. It has been proven to be useful in many tasks of network analysis such as link prediction [18], community detection [5, 21], node classification [4, 37] and visualization [32]. The heterogeneity in data representation, the sparsity of the network, and the varying degrees of various nodes, all play a significant role in making network mining tasks more challenging. To address the sparsity issue, network embedding encodes and represents each node in a unified low-dimensional space, which facilitates a better understanding of semantic relationships and further alleviates the inconveniences caused by sparsity [22]. Network embedding has attracted increasing attention in recent years. The majority of them focuses on unsigned network embedding [2, 10, 18, 20, 22, 24, 31, 32, 34, 39]. For example, in [2], spectral analysis is performed on Laplacian matrix and the top-k eigenvectors are used as the representations of network nodes. t-SNE proposed in [32] embeds the weighted unsigned network to low dimension for visualization by using stochastic neighbor embedding. SDNE [34] studies deep networks for network embedding. DeepWalk [22] introduces the idea of Skip-gram, a word representation model in NLP, to learn node representations from random-walk sequences. Node attributes are also exploited to improve the performance of unsigned network embedding [7, 46]. For example, TADW [46] extends DeepWalk to learn network representation by assuming that each node are associated with rich texts and utilizes these texts to help network embedding. Gong et. al. [7] performed joint link prediction and attribute inference using a social-attribute network.

However, the aforementioned algorithms are designed for unsigned networks and do not take negative links into consideration, while in signed networks there exists both positive and negative links. The majority of network embedding algorithms, such as spectral analysis, t-SNE, DeepWalk, SDNE and TADW, utilize the homophily effects or social influence between pairs of linked nodes. As a result, such pairs are likely to be similar, along with their vector representations. However, this is not true for signed networks due to the existence of negative links; such negative links are used to denote distrust or foe relationships between two nodes. Thus, signed network embedding cannot be carried out by simply extending embedding algorithms for unsigned social networks. In addition, negative links have added value over positive links in various analytical tasks. For example, a small number of negative links can significantly improve positive link prediction performance [16], and they can also improve recommendation performance in social media [33]. Therefore, recently, signed network representation is attracting increasing attention [12, 16, 27, 45, 47]. In [16, 35], degree-based features such as the number of incoming positive and negative links of a node and triad based features that include the structure information of a triad are defined manually and extracted from the network to represent the nodes for sign prediction in networks. Another work in [12] models signed networks using matrix factorization. Zheng et al. [47] extend the spectral embedding to tackle signed network. Song et al. [27] propose two lower bounds of GAUC to put more emphasis on ranking positive links on the top and negative links at the bottom of a ranking list. Wang et al. [36] propose a deep network based model for signed network embedding. As node attributes is helpful for unsigned network work embedding, it has potential to improve the quality of signed network embedding. However, to the best of our knowledge, there's no existing work that exploits node attributes for signed link prediction. Thus, in this paper, we study the novel problem of signed network embedding with attributes. In particular, we propose a novel framework SNEA, which leverages signed social network and user attributes for learning better network representations.

3 PROBLEM STATEMENT

Let $\mathcal{U} = \{u_1, u_2, \ldots, u_n\}$ be a set of users where n is the number of users. A user u_i can have positive or negative links to other users, which results in a signed social network. Let $\mathcal{G} = \{\mathcal{U}, \mathcal{E}\}$ denote the signed social network where $\mathcal{E} \subset \mathcal{U} \times \mathcal{U}$ is a set of edges. We use $\mathbf{A} \in \mathbb{R}^{n \times n}$ to denote the adjacency matrix, where $\mathbf{A}_{ij} = 1$ means positive link from u_i to u_j, $\mathbf{A}_{ij} = -1$ denotes negative link and $\mathbf{A}_{ij} = 0$ means the link is missing. Generally, links on signed social network conveys two social relations such as trust and distrust links in Epinions, and friend and foe links in Slashdot. In addition to the signed social network, each user is also associated with a set of attributes. We use $\mathbf{X} \in \mathbb{R}^{n \times m}$ to denote the user-attribute matrix where m is the number of attributes. With the aforementioned notations and definitions, the problem of signed network embedding can be formally stated as follows:

Given a signed social network $\mathcal{G}$ with adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ and user-attribute matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$, we aim to learn a low dimensional vector representation for each node as

$$f(\mathbf{A}, \mathbf{X}) \rightarrow \mathbf{U}_{\mathcal{G}} \tag{1}$$

where $f(\cdot)$ is the transformation function we want to learn and $\mathbf{U}_{\mathcal{G}} \in \mathbb{R}^{n \times K}$ is the low-dimensional representation of the signed social network with attributes.

4 A PRELIMINARY DATA ANALYSIS FOR ATTRIBUTED SIGNED SOCIAL NETWORKS

Because unsigned network representation learning with attributes strongly depends on the finding that two linked users are more likely to share similar attributes than two users without link, it is

Table 1: Statistics of the Datasets

Dataset	Epinions	Slashdot
# of Users	27,215	33,407
# of Positive Links	326,909	477,176
# of Negative Links	58,695	158,104
# of Reviews/Posts	67,668	94,095
# of Attributes	22,367	19,875
# of Classes	18	22

natural to explore if similar findings exist on signed social networks with attributes [7, 15]. Such an understanding lays the groundwork for a meaningful framework for signed network embedding with attributes. In this section, we will first introduce the datasets and then perform preliminary data analysis to understand the relations between signed links and similarities of user attributes.

4.1 Datasets

For the purpose of this study, we collected two datasets from Epinions and Slashdot [1]. Details about the datasets are described below.

Epinions is a popular product review site. Users in Epinions can create both positive (trust) and negative (distrust) links to other users, which results in signed network $\mathbf{A}$. They can also write reviews about products. Therefore, reviews written by users are used to construct user-attributes matrix $\mathbf{X}$ using bag-of-words. We also collect categories of the products for which they write reviews. For a user who writes reviews for products from multiple categories, we choose the one with most products she writes reviews to as her label and there are 18 categories.

Slashdot is a technology news platform. Users in Slashdot can create friend (positive) and foe (negative) links to other users, which results in the signed network $\mathbf{A}$. They can also write posts about technologies. The posts written by a user are used to construct the user-attributes matrix $\mathbf{A}$ using bag-of-words. We also collect information about the groups that they join. These group identifiers are treated as the class labels and there are 22 classes.

Some additional preprocessing was performed on these datasets by filtering users without any positive and negative links, or with few non-zero entities in the user-attribute matrix $\mathbf{X}$. A number of key statistics of these datasets are illustrated in Table 1. It is evident from the table that (i) positive links are denser than negative links in signed social networks; and (ii) signed networks are very sparse.

4.2 Analysis on Links and Attributes

Previous studies suggest that users in unsigned social networks are likely to share similar attributes with their friends, which serves as the basis of most network representation with attributes [7, 46]. In this subsection, we investigate the attributes similarity of two users in signed social networks.

Let p_i, n_i and r_i denote the number of users with positive, negative and no links with u_i. We construct three sets for each user u_i with the same size of $\min(p_i, n_i, r_i)$. These sets correspond to (i) a friend circle $\mathcal{P}_i$ including randomly selected users who have positive links with u_i; (ii) a foe circle $\mathcal{N}_i$ containing randomly selected users who have negative links with u_i; and (iii) a random circle $\mathcal{R}_i$ including randomly selected users who have no links with u_i.

[1] http://www.epinions.com/ and https://slashdot.org/

Table 2: Average user attributes similarities in $\mathcal{P}$, $\mathcal{N}$ and $\mathcal{R}$

	Epinions		Slashdot	
	CA	COSINE	CA	COSINE
$\mathcal{P}$	75.93	0.0650	21.24	0.0332
$\mathcal{N}$	67.16	0.0540	16.64	0.0289
$\mathcal{R}$	27.30	0.0168	10.34	0.0162

We then create the positive link set $\mathcal{P}$, negative link set $\mathcal{N}$ and the missing link set $\mathcal{R}$ as

$$\begin{aligned} \mathcal{P} &= \{(u_i, u_k) | u_k \in \mathcal{P}_i, i = 1, \ldots, n\} \\ \mathcal{N} &= \{(u_i, u_k) | u_k \in \mathcal{N}_i, i = 1, \ldots, n\} \\ \mathcal{R} &= \{(u_i, u_k) | u_k \in \mathcal{R}_i, i = 1, \ldots, n\} \end{aligned} \tag{2}$$

We can then calculate the similarity for each pair of users (u_i, u_k) in $\mathcal{P}$, $\mathcal{N}$ and $\mathcal{R}$. In this work, we investigate two ways of calculating similarity as follows

- CA: For a pair of users, (u_i, u_k), we compute the similarity $sim(u_i, u_k)$ as the number of common attributes by both u_i and u_k; and
- COSINE: We compute $sim(u_i, u_k)$ as the cosine similarity between the attributes of u_i and attributes of u_k

With these two ways of calculating similarity, we can compute the similarity between each pair of users in $\mathcal{P}$ and we use $\mathbf{s}_p \in \mathbb{R}^{|\mathcal{P}|\times 1}$ to denote the similarity vector. Similarly, we use $\mathbf{s}_n$ and $\mathbf{s}_r$ to denote the similarity vectors for $\mathcal{N}$ and $\mathcal{R}$, respectively. The mean values of $\mathbf{s}_p$, $\mathbf{s}_n$, and $\mathbf{s}_r$ are shown in Table 2. From the table, we make two observations: (i) users are likely to have more similar attributes with their friends than their foes; and (ii) users are likely to have more similar attributes with their foes than users without connections. The second observation is particularly troubling because it shows that natural extensions of homophily do not apply to signed networks. To statistically verify the observations, we conduct a two-sample t-test.

Table 3: P-value of t-test results

	Epinions		Slashdot	
	CI	COSINE	CI	COSINE
$\{\mathbf{s}^p, \mathbf{s}^n\}$	2.31e-31	7.72e-110	3.27e-144	7.85e-148
$\{\mathbf{s}^p, \mathbf{s}^r\}$	8.51e-151	4.52e-194	0	0
$\{\mathbf{s}^n, \mathbf{s}^r\}$	1.65e-130	6.35e-142	5.972e-198	1.32e-126

For two vectors $\{\mathbf{x}, \mathbf{y}\}$, the null hypothesis H_0 and the alternative hypothesis H_1 of the two-sample t-test are defined as follows:

$$H_0 : \mathbf{x} \leq \mathbf{y} \qquad H_1 : \mathbf{x} > \mathbf{y} \tag{3}$$

where the null hypothesis indicates that the mean of $\mathbf{x}$ is less than or equal to that of $\mathbf{y}$. We perform the t-test on $\{\mathbf{s}^p, \mathbf{s}^n\}$, $\{\mathbf{s}^p, \mathbf{s}^r\}$ and $\{\mathbf{s}^n, \mathbf{s}^r\}$, respectively, to substantiate the aforementioned observation. For example, when we perform the t-test on $\{\mathbf{s}^p, \mathbf{s}^n\}$, the null hypothesis is that positively linked users have less common attributes than that of negatively linked users; therefore, if we reject the null hypothesis, then the assumption that positively linked have more common attributes than negatively linked users is verified. The null hypothesis for each test is rejected at significance level $\alpha = 0.01$ with p-value shown in Table 3, which verifies our observations statistically.

The observation can also be explained from a user's perspective. Consider a case that u_i has more common attributes with u_j, which could be explained by their common interest in a particular subject such as *Techonology*. In such a case, it is more likely for u_i to be aware of u_j and have interactions such as comments and replies on u_j's posts. Thus, u_i is more likely to construct positive or negative links with users that share attributes with her. If u_i has negative link with u_j, then they are more likely to hold different opinions with each other on certain products/things and thus u_i's attributes may be less common to u_j than u_i's friends. On the contrary, if u_i has few attributes in common with u_j, then u_i is not likely to know u_j or have interaction with u_j. This implies that attributes are closely related to links and have the potential to help learn better embedding for link prediction or node clustering.

4.3 Extended Structural Balance Theory

Recently, extended structural balance theory [23] is proposed for comparing the closeness of users in signed social networks without considering attributes. The essential idea of extended structural balance theory is that: for four users u_i, u_j, u_k and u_s, with $\mathbf{A}_{ij} = 1, \mathbf{A}_{ik} = -1$ and $\mathbf{A}_{is} = 0$, i.e., u_i has a positive link with u_j, a negative link with u_k and no link with u_s, then $g(u_i, u_j) \leq g(u_i, u_s) \leq g(u_i, u_k)$, where $g(u_i, u_j)$ means distance between u_i and u_j. This can be easily understood from the perspective of the semantic meanings of the links. For example, if a positive link means trust and a negative link means distrust, then u_i should trust u_j most, trust u_k least and be neutral towards u_s. Thus, u_i prefers to be close to u_j and far from u_k. If $\mathbf{U} \in \mathbb{R}^{n\times K}$ is the embedding learned from $\mathbf{A}$, it should also satisfy the closeness measure, i.e., $g(\mathbf{u}_i, \mathbf{u}_j) \leq g(\mathbf{u}_i, \mathbf{u}_s) \leq g(\mathbf{u}_i, \mathbf{u}_k)$, where we use $\mathbf{u}_i$ to represent the i-th row of $\mathbf{U}$. However, from the user-attribute perspective, Table 2 gives $g(u_i, u_j) \leq g(u_i, u_k) \leq g(u_i, u_s)$, where the first part is the same as extended structural balance theory but the second part conflicts with extended structural balance theory. *If we want the embedding to satisfy the manifold structure in terms of attributes and also the extended structural balance theory, then using a single embedding* $\mathbf{U}$ *will result in conflict. However, if we learn two embedding from network and attributes independently, though we can satisfy both manifold structure and extended structural balance theory, it raises two problems: (1) We may not capture the inherent connection between networks and attributes by learning two embeddings and thus result in non-optimal network representation; and (2) It is more natural to learn one unified embedding than two embeddings because a single embedding can be easily used as input to other algorithms for network analysis tasks such as link prediction and node classification.* Therefore, both singe embedding and two embeddings have there their advantage and disadvantages, which brings challenges for signed network embedding.

This observation suggests that the optimal embedding should be one unified embedding matrix that captures the inherent relationship between signed network and also satisfies both extended social balance theory from networks and manifold constraint from attributes. Therefore, we propose the dual embedding called SNEA, which learns the embedding $\mathbf{U} \in \mathbb{R}^{n\times K}$ capturing the properties from signed network and a project matrix $\mathbf{P}$ such that $\mathbf{UP}$ encodes the properties from user attributes. With this setting, for extended structural balance theory, we have $g(\mathbf{u}_i, \mathbf{u}_j) \leq g(\mathbf{u}_i, \mathbf{u}_s) \leq g(\mathbf{u}_i, \mathbf{u}_k)$. In terms of user attributes, we have $g(\mathbf{u}_i, \mathbf{u}_j\mathbf{P}) \leq g(\mathbf{u}_i, \mathbf{u}_k\mathbf{P}) \leq g(\mathbf{u}_i, \mathbf{u}_s\mathbf{P})$. In the next section, we will give details of the proposed SNEA.

5 THE PROPOSED FRAMEWORK

In this section, we give the details of the proposed framework SNEA for modeling signed social network with attributes. Since signed network with attributes has both $\mathbf{A}$ and $\mathbf{U}$, we will introduce how to capture properties of signed network with $\mathbf{U}$ and how to capture the properties of attributes with $\mathbf{UP}$.

5.1 Modeling Signed Social Network

To model signed social networks, one of the most popular and effective methods is to use the low-rank modeling to decompose $\mathbf{A}$ into low-rank matrices that can be used to reconstruct $\mathbf{A}$ [1, 11, 12]. As discussed in the previous section, we use $\mathbf{U}$ to denote the latent user feature matrix for $\mathbf{A}$. Then, the decomposition of $\mathbf{A}$ is as follows:

$$\min_{\mathbf{U},\mathbf{H}} \|\mathbf{W} \odot (\mathbf{A} - \mathbf{U}\mathbf{H}\mathbf{U}^T)\|_F^2 \tag{4}$$

where $\mathbf{W} \in \{0,1\}^{n\times n}$ is the indicator matrix with $\mathbf{W}_{ij} = 1$ if $\mathbf{A}_{ij} \neq 0$ and $\mathbf{W}_{ij} = 0$ otherwise. $\odot$ is the hadamard operation. $\mathbf{H} \in \mathbb{R}^{K\times K}$ is the interaction matrix to capture similarity of user latent features. $\mathbf{A}_{ij}$ is reconstructed as $\mathbf{u}_i\mathbf{H}\mathbf{u}_j^T$.

The extended structural balance theory suggests that a user should sit closer to her friends than non-linked users and sit far away from foes. Among various ways to measure closeness, we choose the Euclidean distance, which is a popular and effective way to measure distance. With Euclidean distance, $g(\mathbf{u}_i, \mathbf{u}_j) = \|\mathbf{u}_i - \mathbf{u}_j\|_2^2$. Then, for three users (u_i, u_j, u_k) with $\mathbf{A}_{ij} = 1$ and $\mathbf{A}_{ik} = 0$, we would prefer $\|\mathbf{u}_i - \mathbf{u}_j\|_2^2 \leq \|\mathbf{u}_i - \mathbf{u}_k\|_2^2$, i.e., the distance of embedding satisfies extended structural balance theory. Similarly, for $\mathbf{A}_{is} = 0$ and $\mathbf{A}_{ik} = -1$, we would want $\|\mathbf{u}_i - \mathbf{u}_s\|_2^2 \leq \|\mathbf{u}_i - \mathbf{u}_k\|_2^2$. For each user u_i, we randomly select T triplets (u_i, u_j, u_k) with $\mathbf{A}_{ij} = 1$ and $\mathbf{A}_{ik} = 0$, and T triplets triplets (u_i, u_j, u_k) with $\mathbf{A}_{ij} = 0$ and $\mathbf{A}_{ik} = -1$. All these triplets form the set $\mathcal{H}$. We then model extended structural balance theory as follows:

$$\min_{\mathbf{U}} \sum_{(u_i,u_ju_k)\in\mathcal{H}} \max(0, \|\mathbf{u}_i - \mathbf{u}_j\|_2^2 - \|\mathbf{u}_i - \mathbf{u}_k\|_2^2) \tag{5}$$

For any triplet $(u_i, u_ju_k) \in \mathcal{H}$, we would like to minimize the violation of extended structural balance, which occurs when we have $\|\mathbf{u}_i - \mathbf{u}_j\|_2^2 > \|\mathbf{u}_i - \mathbf{u}_k\|_2^2$. In other words, we would like to minimize $\|\mathbf{u}_i - \mathbf{u}_j\|_2^2 - \|\mathbf{u}_i - \mathbf{u}_k\|_2^2$. There are two cases:

- Case 1: if $\|\mathbf{u}_i - \mathbf{u}_j\|_2^2 \leq \|\mathbf{u}_i - \mathbf{u}_k\|_2^2$, then extended structural balance is satisfied. $\max(0, \|\mathbf{u}_i - \mathbf{u}_j\|_2^2 - \|\mathbf{u}_i - \mathbf{u}_k\|_2^2)$ reduces to 0, which doesn't penalize w.r.t $\mathbf{U}$ and $\mathbf{P}$.
- Case 2: if $\|\mathbf{u}_i - \mathbf{u}_j\|_2^2 > \|\mathbf{u}_i - \mathbf{u}_k\|_2^2$, then extended structural balance theory is violated. $\max(0, \|\mathbf{u}_i - \mathbf{u}_j\|_2^2 - \|\mathbf{u}_i - \mathbf{u}_k\|_2^2)$ becomes $\|\mathbf{u}_i - \mathbf{u}_j\|_2^2 - \|\mathbf{u}_i - \mathbf{u}_k\|_2^2$, which can be written as $Tr(\mathbf{M}^{ijk}\mathbf{U}\mathbf{U}^T)$, where $\mathbf{M}^{ijk}$ is a sparse $n \times n$ matrix with $\mathbf{M}^{ijk}_{ij} = \mathbf{M}^{ijk}_{ji} = \mathbf{M}^{ijk}_{kk} = -1$, $\mathbf{M}^{ijk}_{ik} = \mathbf{M}^{ijk}_{ki} = \mathbf{M}^{ijk}_{jj} = 1$ and the other entries being 0.

Combining these two cases, we can rewrite Eq.(5) as

$$Tr\Big(\sum_{(u_i,u_j,u_k)\in\mathcal{H}} I_{ijk}\mathbf{M}^{ijk}\mathbf{U}\mathbf{U}^T\Big) \tag{6}$$

where I_{ijk} is 1 if $\|\mathbf{u}_i-\mathbf{u}_j\|_2^2 > \|\mathbf{u}_i-\mathbf{u}_k\|_2^2$ and 0, otherwise. Putting Eq.(4) and Eq.(6) together, we model signed network with social balance theory as follows:

$$\min_{\mathbf{U},\mathbf{H}} \|\mathbf{W}\odot(\mathbf{A}-\mathbf{U}\mathbf{H}\mathbf{U}^T)\|_F^2 + \alpha Tr\Big(\sum_{(u_i,u_j,u_k)\in\mathcal{H}} I_{ijk}\mathbf{M}^{ijk}\mathbf{U}\mathbf{U}^T\Big) \tag{7}$$

where α is a scalar to control the contribution of the extended structural balance theory.

5.2 Modeling User Attributes

Similarly, matrix factorization is a popular method to learn embedding from attributes. To learn the representations from user attributes, we project $\mathbf{U}$ into the user attribute space with the orthogonal projection matrix $\mathbf{P}$, which is popularly used to transform the latent features from one space to another [8, 25]. With projection matrix, we have $\mathbf{UP}$ as the user latent features for the user-attributes matrix. Then, the decomposition is written as follows:

$$\min_{\mathbf{U},\mathbf{V},\mathbf{P}^T\mathbf{P}=\mathbf{I}} \|\mathbf{X}-\mathbf{U}\mathbf{P}\mathbf{V}^T\|_F^2 \tag{8}$$

where $\mathbf{V}\in\mathbb{R}^{m\times K}$ is the latent feature matrix of attributes. Two users with similar attributes are more likely to be interested in the same topic or within the same group, which implies that their latent representations $\mathbf{u}_i\mathbf{P}$ and $\mathbf{u}_j\mathbf{P}$ should be more similar. This can be formally written as:

$$\min_{\mathbf{U},\mathbf{P}} \sum_{i=1}^{n}\sum_{j=1}^{n}\frac{1}{2}\mathbf{S}_{ij}\|\mathbf{u}_i\mathbf{P}-\mathbf{u}_j\mathbf{P}\|_2^2 = \min_{\mathbf{U},\mathbf{P}} \mathrm{Tr}(\mathbf{P}^T\mathbf{U}^T\mathbf{L}\mathbf{U}\mathbf{P}) \tag{9}$$

where $\mathbf{S}$ is the similarity matrix with each element defined as $\mathbf{S}_{ij} = \exp(-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{\sigma})$ and σ is a scalar to control the similarity. $\mathbf{L}=\mathbf{D}-\mathbf{S}$ is the Laplacian matrix, and $\mathbf{D}$ is a diagonal matrix with the on diagonal element $\mathbf{D}_{ii}=\sum_j \mathbf{S}_{ij}$. Thus, if two nodes are more similar, (i.e., $\mathbf{S}_{ij}$ is large), we penalize more with $\mathbf{S}_{ij}\|\mathbf{u}_i\mathbf{P}-\mathbf{u}_j\mathbf{P}\|_2^2$ and thus $\mathbf{u}_i\mathbf{P}$ and $\mathbf{u}_j\mathbf{P}$ are closer. With Eq.(8) and Eq.(9), we model user-attributes with graph regularization as follows:

$$\min_{\mathbf{U},\mathbf{V},\mathbf{P}^T\mathbf{P}=\mathbf{I}} \|\mathbf{X}-\mathbf{U}\mathbf{P}\mathbf{V}^T\|_F^2 + \gamma\mathrm{Tr}(\mathbf{P}^T\mathbf{U}^T\mathbf{L}\mathbf{U}\mathbf{P}) \tag{10}$$

where γ is a scalar to control the contribution of graph regularizer.

5.3 Proposed Model – SNEA

We have introduced our approaches to model network and attributes separately. With these two components, we propose the signed network embedding framework SNEA, which exploits both network and attributes for network representation. The proposed SNEA framework solves the following optimization problem:

$$\begin{aligned}\min_{\mathbf{U},\mathbf{P},\mathbf{V},\mathbf{H}} &\|\mathbf{W}\odot(\mathbf{A}-\mathbf{U}\mathbf{H}\mathbf{U}^T)\|_F^2 + \alpha Tr\Big(\sum_{(u_i,u_j,u_k)\in\mathcal{H}} I_{ijk}\mathbf{M}^{ijk}\mathbf{U}\mathbf{U}^T\Big)\\ &+\beta\|\mathbf{X}-\mathbf{U}\mathbf{P}\mathbf{V}^T\|_F^2 + \gamma\mathrm{Tr}(\mathbf{P}^T\mathbf{U}^T\mathbf{L}\mathbf{U}\mathbf{P}) + \lambda(\|\mathbf{U}\|_F^2+\|\mathbf{V}\|_F^2+\|\mathbf{H}\|_F^2)\\ s.t.\quad &\mathbf{P}^T\mathbf{P}=\mathbf{I}\end{aligned} \tag{11}$$

Algorithm 1 Update $\mathbf{P}$

Input: Initial feasible $\mathbf{P}$, $0<\mu<1, 0<\rho_1<\rho_2<1$
Output: Updated $\mathbf{P}$

```
Compute G and F
Compute L'_τ(S(0)) as L'_τ(S(0)) = −½‖F‖²_F
set τ = 1
for s = 1 to S do
    Compute S(τ) via Eq.(13)
    Computer L'_τ(S(τ)) via Eq.(16)
    if Armijio-Wolfe conditions in Eq.(14) are satisfied then break-out
    τ = μτ
end for
Update P as P = S(τ)
return P
```

where the first term tries to reconstruct the adjacency matrix $\mathbf{A}$; the second term captures the local information from signed network, and α controls the contribution of this term; the third term models the user-attribute matrix and β controls the importance of this term; the fourth term accounts for the manifold structure based on attributes. The last (regularization) term avoids over-fitting and $\lambda>0$ is a scalar to control the contribution of that term.

Discussion One thing worth noting is the orthogonal constraint on $\mathbf{P}$. Consider Eq.(11) without the constraint $\mathbf{P}^T\mathbf{P}=\mathbf{I}$ and replace $\mathbf{UP}$ with $\mathbf{F}$. We can then optimize the new equation w.r.t $\mathbf{U}$ and $\mathbf{F}$ separately. After learning $\mathbf{U}$ and $\mathbf{F}$, $\mathbf{P}$ can be get as $\mathbf{U}^+\mathbf{F}$, where $\mathbf{U}^+$ is the Moore-Penrose pseudoinverse of $\mathbf{U}$. In other words, without the constraint, we are actually learning two separate embeddings and $\mathbf{U}$ only models $\mathbf{A}$. With the constraint, we avoid the trivial solution as $(\mathbf{U}^+\mathbf{F})^T(\mathbf{U}^+\mathbf{F})\neq\mathbf{I}$.

6 AN OPTIMIZATION FRAMEWORK

The objective function in Eq.(11) is not convex, so we cannot update all the variables jointly. To optimize the objective function, we use alternating optimization, which is a popular method used in matrix factorization [12]. Specifically, we optimize one set of variables in the factors, by fixing other variables. Next, we give the details to optimize the objective function followed by complexity analysis.

6.1 Update Rules

For simplicity, we denote the objective function in Eq.(11) as $\mathcal{L}$.

6.1.1 Update Rule for $\mathbf{P}$. It is generally difficult to optimize w.r.t $\mathbf{P}$ due to the orthogonal constraint. In this work, we use a gradient descent optimization procedure with curvilinear search [44] to solve it. In each iteration of the gradient descent procedure, given the current feasible point $\mathbf{P}$, we define $\mathbf{G}$ as

$$\mathbf{G}=\frac{\partial\mathcal{L}}{\partial\mathbf{P}} = 2(\beta\mathbf{U}^T\mathbf{U}\mathbf{P}\mathbf{V}^T\mathbf{V}-\beta\mathbf{U}^T\mathbf{X}\mathbf{V}+\gamma\mathbf{U}^T\mathbf{L}\mathbf{U}\mathbf{P}) \tag{12}$$

Let $\mathbf{F}=\mathbf{G}\mathbf{P}^T-\mathbf{P}\mathbf{G}^T$. It is easy to verify that $\mathbf{F}^T=-\mathbf{F}$ and thus $\mathbf{F}$ is skew-symmetric. The next new point can be searched as a

curvilinear function of a step size variable τ, such that

$$\mathbf{S}(\tau) = (\mathbf{I} + \frac{\tau}{2}\mathbf{F})^{-1}(\mathbf{I} - \frac{\tau}{2}\mathbf{F})\mathbf{Q} \tag{13}$$

Given that $\mathbf{F}$ is skew-symmetric, it is easy to prove that $\mathbf{S}(\tau)^T\mathbf{S}(\tau) = \mathbf{I}$ using the Cayley transformation [14]. Thus we can stay in the feasible region along the curve defined by τ. We thus apply a similar strategy as the standard back-tracking line search to find a proper step size τ using curvilinear search, while guaranteeing the iterations to converge to a stationary point. We determine a proper step size τ as one satisfying the following Armijo-Wolfe conditions:

$$\mathcal{L}(\mathbf{S}(\tau)) \leq \mathcal{L}(\mathbf{S}(0)) + \rho_1\tau\mathcal{L}'_\tau(\mathbf{S}(0)),\ \mathcal{L}'_\tau(\mathbf{S}(\tau)) \geq \rho_2\mathcal{L}'_\tau(\mathbf{S}(0)) \tag{14}$$

Here $\mathcal{L}'_\tau(\mathbf{S}(\tau))$ is the derivative of $\mathcal{L}'_\tau(\mathbf{S}(\tau))$ w.r.t τ,

$$\mathcal{L}'_\tau(\mathbf{S}(\tau)) = -Tr\Big(\mathbf{R}(\tau)^T(\mathbf{I} + \frac{\tau}{2}\mathbf{F})^{-1}\mathbf{F}\frac{\mathbf{P} + \mathbf{S}(\tau)}{2}\Big) \tag{15}$$

where $\mathbf{R}(\tau) = \frac{\partial\mathcal{L}(\mathbf{S}(\tau))}{\partial\mathbf{S}(\tau)}$. Obviously, $\mathbf{S}(0) = \mathbf{P}$ and thus

$$\mathbf{R}(0) = \frac{\partial\mathcal{L}(\mathbf{P})}{\partial\mathbf{P}} = \mathbf{G} \tag{16}$$

Therefore, $\mathcal{L}'_\tau(\mathbf{S}(0)) = -Tr\left(\mathbf{R}(0)^T\mathbf{F}\frac{\mathbf{Q}+\mathbf{S}(0)}{2}\right) = -\frac{1}{2}\|\mathbf{F}\|_F^2$. We summarize the update rule for $\mathbf{P}$ in Algorithm 1, where S is the maximal iterations for the loop.

6.1.2 Update Rules for $\mathbf{U}$, $\mathbf{H}$ *and* $\mathbf{V}$. The gradient of $\mathcal{L}$ w.r.t $\mathbf{U}, \mathbf{P}, \mathbf{H}$ and $\mathbf{V}$ are given as follows

$$\frac{1}{2}\frac{\partial\mathcal{L}}{\partial\mathbf{U}} = -(\mathbf{W}\odot\mathbf{W}\odot\mathbf{A})\mathbf{U}\mathbf{H}^T + \beta\mathbf{U}\mathbf{P}\mathbf{V}^T\mathbf{V}\mathbf{P}^T - \beta\mathbf{X}\mathbf{V}\mathbf{P}^T \tag{17}$$

$$-(\mathbf{W}\odot\mathbf{W}\odot\mathbf{A})^T\mathbf{U}\mathbf{H} + (\mathbf{W}\odot\mathbf{W}\odot\mathbf{U}\mathbf{H}\mathbf{U}^T)\mathbf{U}\mathbf{H}^T \tag{18}$$

$$+(\mathbf{W}\odot\mathbf{W}\odot\mathbf{U}\mathbf{H}\mathbf{U}^T)^T\mathbf{U}\mathbf{H} + \alpha\mathbf{M}\mathbf{U} + \gamma\mathbf{L}\mathbf{U}\mathbf{P}\mathbf{P}^T + \lambda\mathbf{U} \tag{19}$$

$$\frac{1}{2}\frac{\partial\mathcal{L}}{\partial\mathbf{H}} = -\mathbf{U}^T(\mathbf{W}\odot\mathbf{W}\odot\mathbf{A})\mathbf{U} + \lambda\mathbf{H} + \mathbf{U}^T(\mathbf{W}\odot\mathbf{W}\odot\mathbf{U}\mathbf{H}\mathbf{U}^T)\mathbf{U} \tag{20}$$

$$\frac{1}{2}\frac{\partial\mathcal{L}}{\partial\mathbf{V}} = \beta\mathbf{V}\mathbf{P}^T\mathbf{U}^T\mathbf{U}\mathbf{P} - \beta\mathbf{X}^T\mathbf{U}\mathbf{P} + \lambda\mathbf{V} \tag{21}$$

where $\mathbf{M}$ is defined as follows:

$$\mathbf{M} = \sum_{(u_i,u_j,u_k)\in\mathcal{H}} I_{ijk}\mathbf{M}^{ijk} \tag{22}$$

Then the parameters can be updated $\theta \leftarrow \theta - \epsilon\frac{\partial\mathcal{L}}{\partial\theta}$, where $\theta = \{\mathbf{U}, \mathbf{H}, \mathbf{Q}\}$ and ϵ is the learning rate

6.2 Learning Algorithm of SNEA

With the update rules of $\mathbf{U}, \mathbf{P}, \mathbf{H}$ and $\mathbf{V}$ given above, the optimization algorithm for SNEA is shown in Algorithm 2. Next we briefly review Algorithm 2. In Line 1, we first randomly initialize the parameters $\mathbf{U}, \mathbf{P}, \mathbf{H}$ and $\mathbf{V}$. With $\mathbf{U}$, we can calculate $\mathbf{M}$. After initialization, from Line 2 to Line 9, we update these parameters sequentially until convergence. Finally, the embedding of the network is given as $\mathbf{U}$, which can facilitate other network mining tasks such as signed link prediction and node clustering.

The calculation of $\mathbf{M}$ using Eq.(22) involves $|\mathcal{H}|$ sparse matrices. To save memory, we can first initialize $\mathbf{M}$ to be a all zero matrix. We then update $\mathbf{M}$ as : if $(u_i, u_j, u_k) \in \mathcal{H}$ gives $I_{ijk} = 1$, we decrease values of $\mathbf{M}_{ij}, \mathbf{M}_{ji}, \mathbf{M}_{kk}$ by 1 and increase values $\mathbf{M}_{ik}, \mathbf{M}_{ki}, \mathbf{M}_{jj}$ by 1 without actually creating $\mathbf{M}^{ijk}$. Thus, we don't need to create $|\mathcal{H}|$ matrices.

Algorithm 2 Signed Network Embedding with Attributes

Input: $\mathbf{A} \in \mathbb{R}^{n\times n}, \mathbf{X} \in \mathbb{R}^{n\times m}, \alpha, \beta, \gamma, \lambda, K$
Output: $\mathbf{U} \in \mathbb{R}^{n\times K}, \mathbf{P} \in \mathbb{R}^{K\times K}$

1: Initialize $\mathbf{U}, \mathbf{P}, \mathbf{H}$ and $\mathbf{V}$
2: **repeat**
3: calculate $\mathbf{M}$ using Eq.(22)
4: update $\mathbf{U}$ as $\mathbf{U} \leftarrow \mathbf{U} - \epsilon\frac{\partial\mathcal{L}}{\partial\mathbf{U}}$ using Eq.(17)
5: update $\mathbf{P}$ as with Algorithm 1
6: update $\mathbf{H}$ as $\mathbf{H} \leftarrow \mathbf{H} - \epsilon\frac{\partial\mathcal{L}}{\partial\mathbf{H}}$ using using Eq.(20)
7: update $\mathbf{V}$ as $\mathbf{V} \leftarrow \mathbf{V} - \epsilon\frac{\partial\mathcal{L}}{\partial\mathbf{V}}$ using Eq.(21)
8: **until** Convergence
9: return $\mathbf{U}, \mathbf{P}, \mathbf{Q}, \mathbf{H}$

The convergence of the algorithm using alternating optimization is guaranteed [3]. This is because each time we use gradient descent to update the parameters, we monotonically reduce the value of the objective function. Since the value of objective function in Eq.(11) is non-negative, the algorithm will converge and we will arrive at a local optimum.

6.3 Time Complexity Analysis

The algorithm is composed of two parts, i.e., initialization and updating parameters. The most time-consuming part is the updating part. Thus, our analysis will focus on updating parameters. First, the cost of calculating $\mathbf{M}$ using Eq.(22) is $O(K|\mathcal{H}|)$. Considering the fact that $\mathbf{W}$ and $\mathbf{A}$ are very sparse, it is evident that $\mathbf{W}\odot\mathbf{W}\odot\mathbf{A}$ and $\mathbf{W}\odot\mathbf{W}\odot\mathbf{U}\mathbf{H}\mathbf{U}^T$ are also very sparse. Thus, in Eq.(17), the computational cost of $(\mathbf{W}\odot\mathbf{W}\odot\mathbf{A})\mathbf{U}\mathbf{H}^T$ is about $O(nK^2)$ and the computational cost of $(\mathbf{W}\odot\mathbf{W}\odot\mathbf{U}\mathbf{H}\mathbf{U}^T)\mathbf{U}\mathbf{H}^T$ is $O(nK^2 + n^2K)$. The costs of calculating $\mathbf{U}\mathbf{P}\mathbf{V}^T\mathbf{V}\mathbf{P}$ and $\mathbf{X}\mathbf{V}\mathbf{P}^T$ are both $O(nmK + nK^2)$ and cost of $\mathbf{M}\mathbf{U}$ is $O(|\mathcal{H}|K)$ as $\mathbf{M}$ is sparse. Therefore, the cost of calculating $\frac{\partial\mathcal{L}}{\partial\mathbf{U}}$ is $O(nK^2 + n^2K + nmK + K|\mathcal{H}|)$ and updating $\mathbf{U}$ is $O(nK)$. With a similar analysis procedure, we can get the computational cost of updating the other parameters. We omit the details here and directly give the cost. The cost of updating $\mathcal{P}$ Algorithm 1 is $O(nK^2 + n^2K + nmK + SK^3)$. The cost of calculating $\frac{\partial\mathcal{L}}{\partial\mathbf{H}}$ is $O(nK^2 + n^2K)$. And the costs of calculating $\frac{\partial\mathcal{L}}{\partial\mathbf{Q}}$ is $O(nmK + mK^2)$. Thus, the cost of the algorithm in one iteration is $O(nmK + nK^2 + n^2K + mK^2 + SK^3 + K|\mathcal{H}|)$. Since K is usually much smaller than n and m and we set S to be 10 in practice, the cost nK^2, mK^2 and SK^3 can be ignored compared to nmK and n^2K. Since $\mathcal{H}$ is formed by $2T$ triplets for each user, we have $|\mathcal{H}| = O(nT)$. Therefore, the total cost of the algorithm is $O(t(nmK + n^2K + nTK))$, where t is the number of iterations.

7 EXPERIMENTAL RESULTS

In this section, we conduct experiments to evaluate the effectiveness of the proposed framework SNEA and factors that could affect the performance of SNEA. To measure the quality of the embedding learned by SNEA, following the common way [2, 47], we use the embedding for two popular tasks of mining signed social networks, i.e., signed link perdition and node clustering, with comparisons

to state-of-the-art baseline methods. Further experiments are conducted to study parameter sensitivity.

7.1 Signed Link Prediction on Signed Network

In this subsection, we check whether the learned embedding can improve the performance of signed link prediction. We begin by introducing the experimental settings.

7.1.1 Experimental Settings. Let $\mathcal{T} = \{< u_i, u_j > |\mathbf{A}_{ij} = 1\}$ be the set of users having positive links and $\mathcal{D} = \{< u_i, u_j > |\mathbf{A}_{ij} = -1\}$ be the set of users having negative links. For both Epinions and Slashdot datasets, we randomly select 20% positive and negative links from $\mathcal{T}$ and $\mathcal{D}$, respectively, which are used as testing set. We then remove the 20% selected links from $\mathcal{T}$ and $\mathcal{D}$. For the remaining links in $\mathcal{T}$ and $\mathcal{D}$, we choose x% positive and negative links from $\mathcal{T}$ and $\mathcal{D}$, respectively, as training set. We vary x as $\{20, 40, 60, 80, 100\}$. The purpose of varying the values of x is to investigate the performance of the proposed framework on the two datasets with different statistics. In real-world signed social networks such as Epinions and Slashdot, positive links are often much denser than negative links; hence positive and negative links are imbalanced in both training and testing sets. Therefore, following the common way to evaluate the signed link prediction problem, we use AUC and F1 instead of accuracy to assess the performance [16]. AUC [6] measures the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one; a higher AUC would indicate a better predictive performance. F1 score accounts for the trade-off between precision and recall; and higher F1 score indicates higher predictive power.

7.1.2 Performance Comparison of Signed Link Prediction. We compare the proposed framework SNEA with the state-of-the-art signed link prediction methods, where MF, SESN and ELLR only use signed network while FExtra, CMF and ESiNE use both signed network and attribute information. The details of the compared methods are:

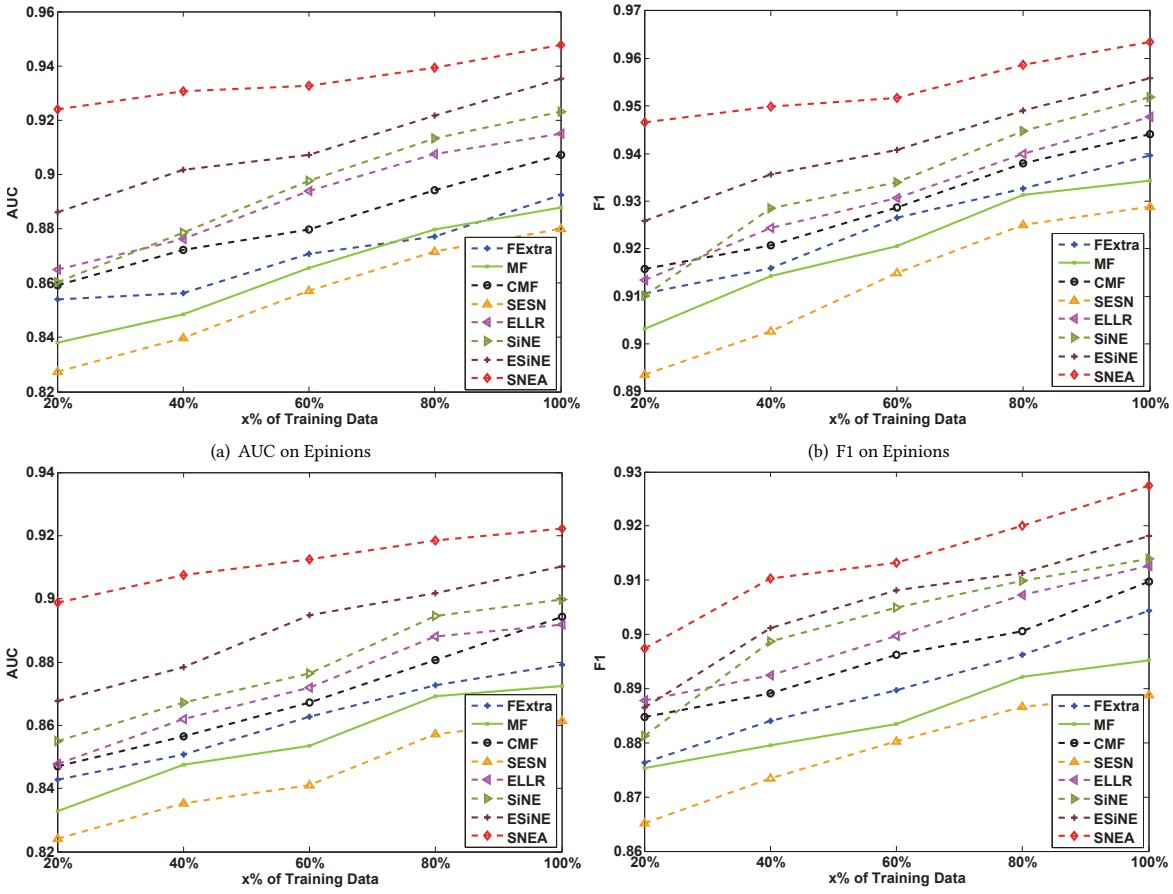
- FExtra: FExtra is a variant of the feature based method proposed in [16]. The original method in [16] extracts 23 features for each pair of nodes from signed network for link prediction. Specifically, for each pair of users (u_i, u_j), it extracts two types of features, i.e., degree based and triad based features. Degree based features contain the degree information such as the number of incoming positive and negative links of u_i, the number of outgoing positive and negative links of u_j and so on; while triad based features include structure information of triads that contains u_i and u_j, such as number of common users. For each pair of users, FExtra uses both the 23 features and their attributes to train a logistic regression classifier for singed link prediction. Note that this feature based method uses *both signed links and node attributes*.
- MF: Matrix factorization [12] based method which factorizes the adjacency matrix $\mathbf{A}$ into two low rank latent matrices and predicts the links by the matrix reconstructed by the two low rank matrices. This can be seen as variant of SNEA without considering attributes.
- CMF: Collective matrix factorization [26] is a matrix factorization model that jointly utilizes signed network $\mathbf{A}$ and user attributes $\mathbf{X}$ as

$$\begin{aligned}\min_{\mathbf{U},\mathbf{V}_1,\mathbf{V}_2} &\|\mathbf{W} \odot (\mathbf{A} - \mathbf{U}\mathbf{V}_1^{\mathbf{T}})\|_F^2 + \alpha\|\mathbf{X} - \mathbf{U}\mathbf{V}_2^T\|_F^2 \\ &+ \lambda(\|\mathbf{U}\|_F^2 + \|\mathbf{V}_1\|_F^2 + \|\mathbf{V}_2\|_F^2)\end{aligned} \tag{23}$$

  Signed links are then predicted as $\mathbf{U}\mathbf{V}_1^T$. It is a naive way of modeling *both links and attributes* without considering extend social balance theory and attributes similarity.
- SESN: Spectral embedding of signed network [47] is a normalized spectral analysis method for signed graphs. It defines signed Laplacian via Rayleigh quotients and computes the top-k eigenvectors as node representation.
- ELLR: ELLR [27] is the state-of-the-art ranking based signed network embedding algorithm which put more emphasis on ranking positive links on the top and negative links at the bottom of a ranking list.
- SiNE: SiNE [36] is a deep network based method for signed network embedding. It tries to push a user closer to his friends but faraway from his foes. No attributes are used.
- ESiNE: Enhanced SiNE which aggregates the embedding of SiNE and the embedding of attributes using matrix factorization. This method also uses *both link and attributes*.

Since distrust (negative link) is not the negation of trust (positive link), the trust/distrust prediction problem cannot be successfully carried out by directly applying trust predictors [30]. Therefore we do not compare SNEA with traditional trust predictors such as [13, 29]. Note that we use cross-validation to determine parameters for all baseline methods. For SNEA, α, γ and λ are set to 0.1, β is set to 1, $K = 20$ and $T = 30$. More details about parameter selection will be discussed in the following subsections. We use $\mathbf{U}$ to reconstruct $\mathbf{A}$ as $\tilde{\mathbf{A}} = \mathbf{U}\mathbf{H}\mathbf{U}^T$ for signed link prediction. Each experiments are conducted 5 times and the average performance are reported in Figure 1. From these figures, we make the following observations:

- In general, with the increase of the training data, the performance of all methods increases.
- Among MF, SESN, ELLR and SiNE, generally, SiNE outperforms the other three. This is because SiNE is a deep network based method which tries to learning embeddings such that a user is more closer to his friends than his foes, which is more efficient. However, SNEA outperforms SiNE, which is because (i) SNEA uses extended social balance theory to guide the representation learning process; and (2) SNEA incorporate attributes, which provides complementary information.
- By introducing attributes, ESiNE outperforms SiNE, which indicates attributes are useful for signed link prediction.
- The proposed framework SNEA always obtain the best performance. Though, FExtra, CMF ESiNE also exploit signed network and attributes for signed link prediction, SNEA outperforms them, which suggests that SNEA is more effective in modeling signed network and attributes simultaneously. This is because SNEA takes extended social balance theory and attribute similarity into consideration and thus can learn better embedding; while ESiNE simply aggregates two embeddings.



(a) AUC on Epinions

(b) F1 on Epinions

(c) AUC on Slashdot

(d) F1 on Slashdot

Figure 1: Performance comparison of signed link prediction on Epinions and Slashdot in terms of AUC and F1

We conduct t-test on all performance comparisons and it is evident from t-test that all improvements are significant. In summary, SNEA obtains significant performance improvement in signed link prediction by leveraging both signed network and user attributes and considering the attributes similarity and semantic meanings of signed links.

7.2 Node Clustering on Signed Social Network

In this subsection, we further check whether the learned signed network embedding with attributes can improve the performance of node clustering. We first introduce experimental settings.

7.2.1 Experimental Settings. In this experiment, we also use Epinions and Slashdot for node clustering. Specifically, for both datasets, we first learn the signed network embedding. With the learned embedding, K-means is applied on the embedding to cluster the nodes into C clusters. Following the common way to measure the quality of clusters [38, 40], two widely used evaluation metrics, accuracy (ACC) and normalized mutual information (NMI), are employed. The larger ACC and NMI are, the better performance is. Since K-means depends on initialization, following previous work, we repeat the experiments 20 times and the average results with standard deviation are reported. As shown in Table 1, Epinions has 18 classes and Slashdot has 22 classes. One thing to note is that both datasets are imbalanced. Some classes have more than 2,000 users, while some only has around 500 hundred users.

7.2.2 Performance Comparison of Node Clustering. We compare the proposed framework with state-of-the-art signed network embedding and feature based methods. The compared algorithms are FExtra, MF, CMF, SNSE, ELLR, SiNE and ESiNE, which are introduced in section 7.1.2. *Note that FExtra, CMF and ESiNE utilizes both the singed network and attributes.* For FExtra, we want the representation of each node instead of features for each pair for clustering. Thus, we only extract degree based features from signed network as triad based features are designed for pairs of nodes. These manually extracted features and attributes are used as node representation for node clustering. For the other methods, we first learn embedding and then use embedding for node clustering.

There are some parameters to be set. For MF, CMF and SNEA, we need to set the latent dimension K. We tune K for these methods by a "grid-search" strategy from $\{20, 40, 60, 100\}$. For SiNE, we use the default setting as in [36]. For SNEA, we empirically set α, γ and λ to 0.1, β to 1 and $T = 30$. More details about parameter selection will be discussed in the following subsections. The average experimental results of different methods on the datasets are summarized in Table 4 and 5. From these two tables, we make the following observations:

Table 4: Node clustering results (ACC±std) of different algorithms on different Epinions and Slashdot.

Dataset	MF	CMF	FExtra	SESN	ELLR	SiNE	ESiNE	SNEA
Epinions	0.1835±0.0149	0.2129±0.0187	0.1868±0.0159	0.1829±0.0156	0.2059±0.0164	0.2097±0.0274	0.2189±0.0241	**0.2453**±0.0148
Slashdot	0.2274±0.0197	0.2585±0.0205	0.2398±0.0176	0.2301±0.0216	0.2503±0.0201	0.2598±0.0225	0.2686±0.0196	**0.2869**±0.0197

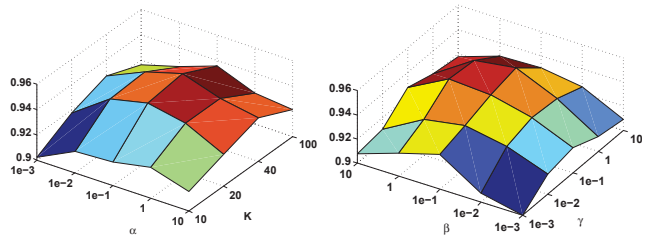
Table 5: Node clustering results (NMI±std) of different algorithms on different Epinions and Slashdot.

Dataset	MF	CMF	FExtra	SESN	ELLR	SiNE	ESiNE	SNEA
Epinions	0.1329±0.0058	0.1627±0.0079	0.1359±0.0048	0.1314±0.0052	0.1549±0.0062	0.1628±0.0074	0.1701±0.0068	**0.1825**±0.0049
Slashdot	0.1576±0.0087	0.1845±0.0089	0.1692±0.0078	0.1602±0.0093	0.1739±0.0084	0.1796±0.0092	0.1879±0.0082	**0.2065**±0.0088

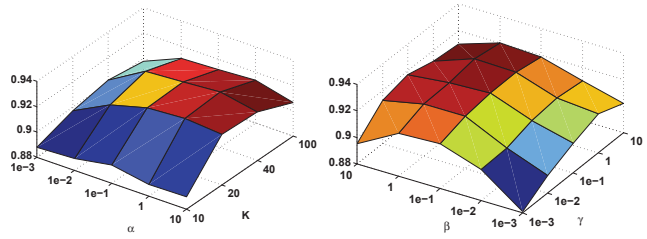
- For both datasets, CMF outperforms MF, and similarly, ESiNE outperforms SiNE. CMF and ESiNE exploit both signed network and attributes while their variants MF and SiNE only utilize signed network. This suggests that attributes are helpful for node clustering. There are two reasons: (1) The signed social network is very sparse, which makes it difficult for node clustering; and (2) Users attributes provides complementary information such as users interests and properties, which are not available in signed social networks and can mitigate network sparsity problem.
- For the algorithms that don't consider node attributes, i.e., SESN, MF, ELLR and SiNE, SiNE outperforms the other three. However, SNEA outperforms SiNE, which implies that by considering attributes, we can learn better node representation for clustering.
- The proposed framework SNEA outperforms all the baseline methods. We also conduct t-test on all performance comparisons and it is evident that improvements are significant, which demonstrates the quality of the learned network embedding. In particular, SNEA, CMF, ESiNE and FExtra use network and attributes for constructing node representation. SNEA achieves better performance than the other three. This is because FExtra extracts features separately from network and attributes; CMF doesn't consider extended structural balance theory and attributes manifold structure; ESiNE simply aggregates network embedding from SiNE and features from MF on attributes; while SNEA leverages both network and attributes into a unified framework by considering extended structural balance theory and attributes manifold structure.

7.3 Parameter Analysis for SNEA

The proposed framework has four importance parameters, i.e., α controlling the contribution of extended social balance theory; β and γ controls the contribution of attributes; and K controls the number of dimension. In this section, we investigate the impact of these parameters on the link prediction and node clustering performances of SNEA. We use the same experimental settings as previous section. For signed link prediction, we only show the results for $x = 100$ in terms of AUC as we have similar observations for $x = \{20, 40, 60, 80\}$ and F1 metric. Similarly, for node clustering, we only show results in terms of ACC. We first fix β to be 1, γ to be 0.1 and vary the values of α as $\{0.001, 0.01, 0.1, 1, 10\}$, the values of K as $\{10, 20, 40, 100\}$. We then fix α to be 0.1, K to be 20 and vary the values of β as $\{0.001, 0.01, 0.1, 1, 10\}$ and the values of γ as $\{0.001, 0.01, 0.1, 1, 10\}$. The results for signed link prediction and node clustering are shown in Figure 2 and 3, respectively. From these figures, we make the following observations: (1) Generally, as the increase of K, the performance first increase and then decrease after K reaches certain value. The same holds for α, β and γ, which is because when β and γ are small, the contribution of attributes becomes small; and (2) The performance is generally better and more stable when the value of K is in $[20, 40]$, and the value of α is in $[0.01, 10]$. Similarly, a value of α (β) within $[0.01, 10]$ gives better performance. This observation eases the parameter selection.



(a) AUC on Epinions with $\beta = 1$ and $\gamma = 0.1$ (b) AUC on Epinions with $K = 20$ and $\alpha = 0.1$

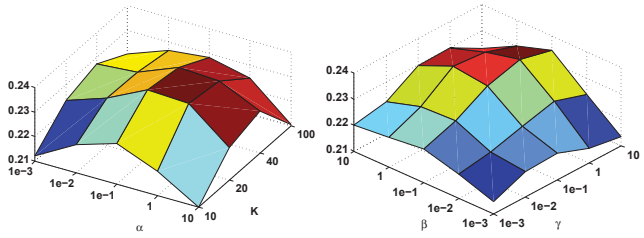


(c) AUC on Slashdot with $\beta = 1$ and $\gamma = 0.1$ (d) AUC on Slashdot with $K = 20$ and $\alpha = 0.1$

Figure 2: Parameter Sensitivity for SNEA on Link Prediction

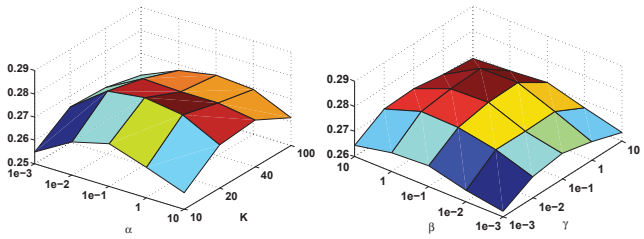
8 CONCLUSION

In this paper, we investigate the problem of signed social network embedding with attributes. We first study the attributes similarity of users with positive, negative and no links, which lays the foundation of the proposed framework SNEA. We then leverage both signed social network and user attributes into a unified framework SNEA by incorporating the extended structure balance theory and the relationship between user links and user attributes. Experiments on real-world datasets demonstrate that the learned embedding by leveraging signed network and user attributes outperforms signed network embedding without attributes in signed network mining tasks such as signed link prediction and node clustering.



(a) ACC on Epinions with $\beta = 1$ and $\gamma = 0.1$

(b) ACC on Epinions with $K = 20$ and $\alpha = 0.1$



(c) ACC for Slashdot with $\beta = 1$ and $\gamma = 0.1$

(d) ACC on Slashdot with $K = 20$ and $\alpha = 0.1$

Figure 3: Parameter Sensitivity for SNEA on Node Clustering

There are several interesting directions need further investigation. First, in this work, we only consider signed link prediction and node clustering. One future work is to use the learned embedding for other network mining tasks such as signed network visualization. Second, user links and user attributes have close relationship and user attributes are helpful for link prediction. Thus, another direction is to investigate if signed links are useful for user attributes prediction. In addition, visual features have been demonstrated to be very effective for data mining [41–43]. Therefore, we also want to exploit visual features for signed network embedding.

9 ACKNOWLEDGEMENTS

This material is based upon work supported by, or in part by, the National Science Foundation (NSF) under the grant #1614576 and Office of Naval Research (ONR) under the grant N00014-16-1-2257.

REFERENCES

[1] Ghazaleh Beigi, Jiliang Tang, Suhang Wang, and Huan Liu. 2016. Exploiting emotional information for trust/distrust prediction. In *SDM*.
[2] Mikhail Belkin and Partha Niyogi. 2001. Laplacian eigenmaps and spectral techniques for embedding and clustering.. In *NIPS*, Vol. 14. 585–591.
[3] James C Bezdek and Richard J Hathaway. 2003. Convergence of alternating optimization. *Neural, Parallel & Scientific Computations* (2003).
[4] Smriti Bhagat, Graham Cormode, and S Muthukrishnan. 2011. Node classification in social networks. In *Social network data analytics*. Springer, 115–148.
[5] Yuxiao Dong, Jing Zhang, Jie Tang, Nitesh V Chawla, and Bai Wang. 2015. CoupledLP: Link Prediction in Coupled Networks. In *SIGKDD*. ACM, 199–208.
[6] Tom Fawcett. 2006. An introduction to ROC analysis. *Pattern recognition letters* 27, 8 (2006), 861–874.
[7] Neil Zhenqiang Gong, Ameet Talwalkar, Lester Mackey, Ling Huang, Eui Chul Richard Shin, Emil Stefanov, Elaine Runting Shi, and Dawn Song. 2014. Joint link prediction and attribute inference using a social-attribute network. *ACM Transactions on Intelligent Systems and Technology (TIST)* 5, 2 (2014), 27.
[8] Yuhong Guo and Min Xiao. 2012. Cross language text classification via subspace co-regularized multi-view learning. In *ICML*.
[9] Ahmed Hassan, Amjad Abu-Jbara, and Dragomir Radev. 2012. Extracting signed social networks from text. In *ACL Workshop on Graph-based Methods for NLP*.
[10] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*.
[11] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast matrix factorization for online recommendation with implicit feedback. In *SIGIR*.
[12] Cho-Jui Hsieh, Kai-Yang Chiang, and Inderjit S Dhillon. 2012. Low rank modeling of signed networks. In *SIGKDD*. ACM, 507–515.
[13] Jin Huang, Feiping Nie, Heng Huang, and Yi-Cheng Tu. 2012. Trust prediction via aggregating heterogeneous social networks. In *CIKM*. ACM.
[14] Alan Jennings and John J McKeown. 1992. *Matrix computation*. John Wiley & Sons Inc.
[15] Timothy La Fond and Jennifer Neville. 2010. Randomization tests for distinguishing social influence and homophily effects. In *WWW*. ACM, 601–610.
[16] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. 2010. Predicting positive and negative links in online social networks. In *WWW*. ACM, 641–650.
[17] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. 2010. Signed networks in social media. In *SIGCHI*. ACM, 1361–1370.
[18] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *Journal of the American society for information science and technology* 58, 7 (2007), 1019–1031.
[19] Silviu Maniu, Bogdan Cautis, and Talel Abdessalem. 2011. Building a signed network from interactions in Wikipedia. In *Databases and Social Networks*.
[20] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. SIGKDD. 1105–1114.
[21] Symeon Papadopoulos, Yiannis Kompatsiaris, Athena Vakali, and Ploutarchos Spyridonos. 2012. Community detection in social media. *Data Mining and Knowledge Discovery* 24, 3 (2012), 515–554.
[22] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *SIGKDD*. ACM, 701–710.
[23] Yi Qian and Sibel Adali. 2013. Extended structural balance theory for modeling trust in social networks. In *PST*. IEEE, 283–290.
[24] Sam T Roweis and Lawrence K Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science* 290, 5500 (2000), 2323–2326.
[25] Xiaoxiao Shi, Qi Liu, Wei Fan, S Yu Philip, and Ruixin Zhu. 2010. Transfer learning on heterogenous feature spaces via spectral transformation. In *ICDM*. IEEE, 1049–1054.
[26] Ajit P Singh and Geoffrey J Gordon. 2008. Relational learning via collective matrix factorization. In *SIGKDD*. ACM, 650–658.
[27] Dongjin Song, David A Meyer, and Dacheng Tao. 2015. Efficient latent link recommendation in signed networks. In *SIGKDD*. ACM, 1105–1114.
[28] Jiliang Tang, Yi Chang, Charu Aggarwal, and Huan Liu. 2016. A Survey of Signed Network Mining in Social Media. *ACM Comput. Surv.* 49, 3 (Aug. 2016).
[29] Jiliang Tang, Huiji Gao, Xia Hu, and Huan Liu. 2013. Exploiting homophily effect for trust prediction. In *WSDM*. ACM, 53–62.
[30] Jiliang Tang, Xia Hu, and Huan Liu. 2014. Is distrust the negation of trust?: the value of distrust in social media. In *Proceedings of the 25th ACM conference on Hypertext and social media*. ACM, 148–157.
[31] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *WWW*. 1067–1077.
[32] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9, 2579-2605 (2008), 85.
[33] Patricia Victor, Chris Cornelis, Martine De Cock, and Ankur Teredesai. 2009. Trust-and distrust-based recommendations for controversial reviews. (2009).
[34] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural Deep Network Embedding. In *SIGKDD*. 1225–1234.
[35] Jing Wang, Jie Shen, Ping Li, and Huan Xu. 2017. Online Matrix Completion for Signed Link Prediction. In *WSDM*.
[36] Suhang Wang, Jiliang Tang, Charu Aggarwal, Yi Chang, and Huan Liu. 2017. Signed Network Embedding in Social Media. In *SDM*.
[37] Suhang Wang, Jiliang Tang, Charu Aggarwal, and Huan Liu. 2016. Linked document embedding for classification. In *CIKM*. ACM.
[38] Suhang Wang, Jiliang Tang, and Huan Liu. 2015. Embedded Unsupervised Feature Selection.. In *AAAI*. Citeseer, 470–476.
[39] Suhang Wang, Jiliang Tang, Fred Morstatter, and Huan Liu. 2016. Paired restricted boltzmann machine for linked data. In *CIKM*. ACM.
[40] Suhang Wang, Yilin Wang, Jiliang Tang, Charu Aggarwal, Suhas Ranganath, and Huan Liu. 2017. Exploiting hierarchical structures for unsupervised feature selection. In *SDM*. SIAM.
[41] Suhang Wang, Yilin Wang, Jiliang Tang, Kai Shu, Suhas Ranganath, and Huan Liu. 2017. What your images reveal: Exploiting visual contents for point-of-interest recommendation. In *WWW*. 391–400.
[42] Yilin Wang, Suhang Wang, Jiliang Tang, Huan Liu, and Baoxin Li. 2015. Unsupervised Sentiment Analysis for Social Media Images.. In *IJCAI*. 2378–2379.
[43] Yilin Wang, Suhang Wang, Jiliang Tang, Huan Liu, and Baoxin Li. 2016. Ppp: Joint pointwise and pairwise image label prediction. In *CVPR*.
[44] Zaiwen Wen and Wotao Yin. 2013. A feasible method for optimization with orthogonality constraints. *Mathematical Programming* 142, 1-2 (2013), 397–434.
[45] Bo Yang, Xuehua Zhao, and Xueyan Liu. 2015. Bayesian Approach to Modeling and Detecting Communities in Signed Network.. In *AAAI*. 1952–1958.
[46] Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Y Chang. 2015. Network representation learning with rich text information. In *IJCAI*. 2111–2117.
[47] Q Zheng and DB Skillicorn. 2015. Spectral embedding of signed networks. In *SDM*. SIAM, 55–63.