SHINE: Signed Heterogeneous Information Network Embedding for Sentiment Link Prediction

Hongwei Wang*
Shanghai Jiao Tong University
Shanghai, China
wanghongwei55@gmail.com

Fuzheng Zhang
Microsoft Research Asia
Beijing, China
fuzzhang@microsoft.com

Min Hou
University of Science and Technology of China, Hefei, Anhui, China
hmhoumin@gmail.com

Xing Xie
Microsoft Research Asia
Beijing, China
xingx@microsoft.com

Minyi Guo†
Shanghai Jiao Tong University
Shanghai, China
guo-my@cs.sjtu.edu.cn

Qi Liu
University of Science and Technology of China, Hefei, Anhui, China
qiliuql@ustc.edu.cn

ABSTRACT

In online social networks people often express attitudes towards others, which forms massive sentiment links among users. Predicting the sign of sentiment links is a fundamental task in many areas such as personal advertising and public opinion analysis. Previous works mainly focus on textual sentiment classification, however, text information can only disclose the "tip of the iceberg" about users' true opinions, of which the most are unobserved but implied by other sources of information such as social relation and users' profile. To address this problem, in this paper we investigate how to predict possibly existing sentiment links in the presence of heterogeneous information. First, due to the lack of explicit sentiment links in mainstream social networks, we establish a labeled heterogeneous sentiment dataset which consists of users' sentiment relation, social relation and profile knowledge by entity-level sentiment extraction method. Then we propose a novel and flexible end-to-end *Signed Heterogeneous Information Network Embedding* (SHINE) framework to extract users' latent representations from heterogeneous networks and predict the sign of unobserved sentiment links. SHINE utilizes multiple deep autoencoders to map each user into a low-dimension feature space while preserving the network structure. We demonstrate the superiority of SHINE over state-of-the-art baselines on link prediction and node recommendation in two real-world datasets. The experimental results also prove the efficacy of SHINE in cold start scenario.

ACM Reference Format:
Hongwei Wang, Fuzheng Zhang, Min Hou, Xing Xie, Minyi Guo, and Qi Liu. 2018. SHINE: Signed Heterogeneous Information Network Embedding for Sentiment Link Prediction. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining (WSDM'18).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3159652.3159666

*This work is done while H. Wang and M. Hou are visiting Microsoft Research Asia.
†M. Guo is the corresponding author.

WSDM'18, February 5–9, 2018, Marina Del Rey, CA, USA
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5581-0/18/02…$15.00
https://doi.org/10.1145/3159652.3159666

1 INTRODUCTION

The past decade has witnessed the proliferation of online social networks such as Facebook, Twitter and Weibo. In these social network sites, people often share feelings and express attitudes towards others, e.g., friends, movie stars or politicians, which forms *sentiment links* among these users. Different from explicit social links indicating friend or follow relationship, sentiment links are implied by the semantic content posted by users, and involve different types: *positive* sentiment links express like, trust or support attitudes, while *negative* sentiment links signify dislike or disapproval of others. For example, a tweet saying "*Vote Trump!*" shows a positive sentiment link from the poster to Donald Trump, and "*Trump is mad...*" indicates the opposite case.

For a given sentiment link, we define its *sign* to be positive or negative depending on whether its related content expresses a positive or negative attitude from the generator of the link to the recipient [14], and all such sentiment links form a new network topology called *sentiment network*. Previous work [6, 11, 15] mainly focuses on sentiment classification based on the concrete content posted by users. However, they cannot detect the existence of sentiment links without any prior content information, which greatly limits the number of possible sentiment links that could be found. For example, if a user does not post any word concerning Trump, it is impossible for traditional sentiment classifiers to extract the user's attitude towards him because "one cannot make bricks without straw". Therefore, a fundamental question is, can we predict the sign of a given sentiment link without observing its related content? The solution to this problem will benefit a great many online services such as personalized advertising, new friends recommendation, public opinion analysis, opinion polls, etc.

Despite the great importance, there is little prior work concerning predicting the sign of sentiment links among users in social networks. The challenges are two-fold. On the one hand, lack of explicit sentiment labels makes it difficult to determine the polarity of existing and potential sentiment links. On the other hand, the complexity of sentiment generation and the sparsity of sentiment links make it hard for algorithms to achieve desirable performance. Recently, several studies [12, 14, 31, 35] propose methods to solve the problem of predicting signed links. However, they rely heavily on manually designed features and cannot work well in real-world scenarios. Another promising approach called *network embedding*

[8, 17, 23, 26], which automatically learns features of users in network, seems plausible to solve the task. However, they can only apply to networks with positive-weighted (i.e., unsigned) and single-type (i.e., homogeneous) edges, which limits their power in the task of practical sentiment link prediction.

Based on the above facts, in this paper we investigate the problem of predicting sentiment links in absence of sentiment related content in online social networks. Our work is two-step. First, considering the lack of labeled data, we establish a labeled sentiment dataset from Weibo, one of the most popular social network sites in China. We leverage state-of-the-art entity-level sentiment extraction method to calculate the sentiment of the poster towards the celebrity in each tweet. Besides, to handle the sparsity problem, we collect two additional types of side information: social relationship among users and profile knowledge of users and celebrities. Our choices are enlightened by [27] and [34], respectively, in which [27] demonstrates that the structural information of social networks can greatly affect users' preference towards online items, and [34] proves that information from knowledge base could boost the performance of recommendation. The heterogeneous information networks are illustrated in Fig. 1.

To explore more possible sentiment links from the network, in the second step, we propose a novel end-to-end framework termed as *Signed Heterogeneous Information Network Embedding* (SHINE). Greatly different from existing network embedding approaches, SHINE is able to learn user representation and predict sentiment from signed heterogeneous networks. Specifically, SHINE adopts multiple deep autoencoders [20], a type of deep-learning-based embedding technique, to extract users' highly nonlinear representations from the sentiment network, social network and profile network, respectively. The learned three types of user representations are subsequently fused together by specific aggregation function for further sentiment prediction. In addition to the adaptability to signed heterogeneous networks, the superiority of SHINE also lies in its end-to-end prediction technology and high flexibility of adding or removing modules of side information (i.e., social relationship and profile knowledge), which is discuss in Section 5.

We conduct extensive experiments on two real-world datasets. The results show that SHINE achieves substantial gains compared with baselines. Specifically, SHINE outperforms other strong baselines by 8.8% to 16.8% in the task of link prediction on Accuracy, and by 17.2% to 219.4% in the task of node recommendation on Recall@100 for positive nodes. The results also prove that SHINE is able to utilize the side information efficiently, and maintains a decent performance in cold start scenario.

2 RELATED WORK

2.1 Signed Link Prediction

Our problem of predicting positive and negative sentiment links connects to a large body of work on signed social networks, including trust propagation [9], spectral analysis [13], and social media mining [22]. For the link prediction problem in signed graphs, Leskovec. *et al.* [14] adopt signed triads as features for prediction based on structural balance theory. Ye *et al.* [31] utilize transfer learning to leverage edge sign information from source network and improve prediction accuracy in target network. Tang *et al.* design NeLP framework [21] which exploits positive links in social media to predict negative links. The difference between the above work and ours is that we construct a labeled dataset by entity-level sentiment extraction method, as there is no explicit signed links in mainstream online social networks. Besides, we use state-of-the-art deep learning approach to learn the representation of links.

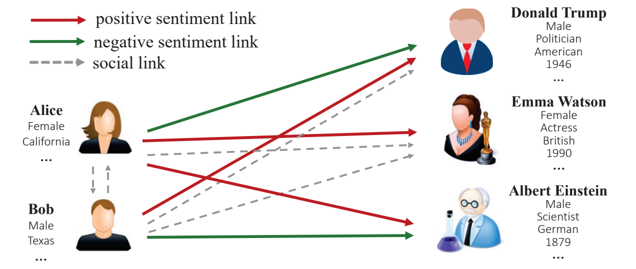


Fig. 1: Illustration of a snippet of heterogeneous networks with sentiment, social relationship and user profile.

2.2 Network Embedding

There is a long history of work on network embedding. Earlier works such as IsoMap [24] and Laplacian Eigenmap [1] first construct the affinity graph of data using the feature vectors and then embed the affinity graph into a low-dimension space. Recently, DeepWalk [17] deploys random walk to learning representations of social network. LINE [23] proposes objective functions that preserve both local and global network structures for network embedding. Node2vec [8] designs a biased random walk procedure to learn a mapping of nodes that maximizes the likelihood of preserving network neighborhoods of nodes. SDNE [26] uses autoencoder to capture first-order and second-order network structures and learn user representation. However, these methods can only address unsigned and homogeneous networks. Additionally, several studies focus on representation learning in the scenario of heterogeneous network [3, 32], attributed network [10], or signed network [29, 33]. However, these methods are specialized in only one particular type of networks, which is not applicable to the problem of sentiment prediction in real-world signed and heterogeneous networks.

3 DATASET ESTABLISHMENT

In this section we introduce the process of collecting data from online social networks, and discuss the details of how to extract sentiment towards celebrities from tweets.

3.1 Data Collection

3.1.1 Weibo Tweets. We select Weibo[1] as the online social networks studied in this work. Weibo is one of the most popular social network sites in China which is akin to a hybrid of Facebook and Twitter. We collected 2.99 billion tweets on Weibo from August 14, 2009 to May 23, 2014 as raw dataset. To filter out useful data which contains sentiment towards celebrities, we first apply Jieba[2], the most popular Chinese text segmentation tool, to tag the part of

[1]http://weibo.com
[2]https://github.com/fxsjy/jieba

Table 1: Statistics of Weibo sentiment datasets. "celebrities v." means the celebrities owning verified accounts on Weibo.

# users	12,814	# social links	71,268
# celebrities	1,723	# tweets	126,380
# celebrities v.	706	# pos. tweets	108,906
# ordinary users	11,091	# neg. tweets	17,474

speech (POS) of each word for each tweet. Then we select those tweets containing words with POS tagging as "person name" which exist in our established celebrity database (detailed in Section 3.1.4). After getting the set of candidate tweets, for each tweet we calculate its sentiment value (-1 to +1) towards the mentioned celebrities, and select those tweets with high absolute sentiment values. The final dataset consists of a set of triples (a, b, s), where a is the user who posts the tweet, b is the certain celebrity mentioned in the tweet, and $s \in \{+1, -1\}$ is the sentiment polarity of user a towards user b. The method of calculating sentiment values is detailed in Section 3.2.

3.1.2 Social Relation. In addition to the sentiment dataset, we also collect the social relation among users from Weibo. The dataset of social relation consists of tuples (a, b), where a is the follower and b is the followee.

3.1.3 Profile of Ordinary Users. The profile of ordinary users are collected from Weibo. For each ordinary user, we extract two of his attributes, *gender* and *location*, as his profile information. The attribute values are represented as one-hot vectors.

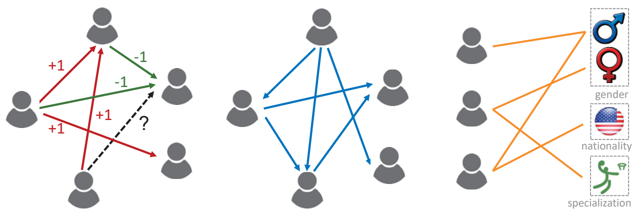
3.1.4 Profile of Celebrities. We use Microsoft Satori[3] knowledge base to extract profile of celebrities. First, we traverse the knowledge base and select terms with object type as "person". Then we filter out popular celebrities with high edit frequency in knowledge base and high appearance frequency in Weibo tweets. For each of these "hot" celebrities, we extract 9 attributes as his profile information: *place of birth*, *date of birth*, *ethnicity*, *nationality*, *specialization*, *gender*, *height*, *weight*, and *astrological sign*. Values of these attributes are discretized so that every celebrity's attribute values can be expressed as one-hot vectors. Furthermore, we remove celebrities with ambiguous names as well as other noises.

3.2 Sentiment Extraction

To extract users' sentiment towards celebrities in tweets, we first generate a sentiment lexicon consisting of words and their *sentiment orientation* (SO) scores. To achieve this, we manually construct a emoticon-sentiment mapping file and map each tweet to positive or negative class according to the label of emoticon appeared in the tweet. For example, *"I love Kobe! [kiss]"* is mapped to positive class if the key-value pair ([kiss], *positive*) exists in the emoticon-sentiment mapping file. Note that the class of emoticon cannot be directly regarded as the sentiment towards celebrities since we found a large number of mismatch cases, e.g., *"Miss you Taylor Swift [cry][cry]"*. Afterwards, for each word (segmented by Jieba) with occurrence frequency from 2,000 to 10,000,000 in the raw tweets datasets, similar to [2], we calculate its SO score as

$$SO(word) = PMI(word, pos) - PMI(word, neg), \qquad (1)$$

[3]http://searchengineland.com/library/bing/bing-satori



(a) Sentiment network (b) Social network (c) Profile network

Fig. 2: Illustration of the three studied networks.

where PMI is the *point-wise mutual information* [25] defined as $PMI(x, y) = \log \frac{p(x,y)}{p(x)p(y)}$, *pos* and *neg* are the tweets of positive and negative class, respectively. SO scores are subsequently normalized to $[-1, 1]$.

After getting the lexicon, we use SentiCircle [19] to calculate sentiment towards celebrities in each tweet. Given a piece of tweet as well as the mentioned celebrity, we represent the contextual semantics of the celebrity as a polar coordinate space, where the celebrity is situated in the origin and other terms in the tweet are scattered around. Specifically, for celebrity term c, the coordinate of term t_i is (r_i, θ_i), where r_i is the inverse of distance between c and t_i in syntax dependence graph generated by LTP [4], and $\theta_i = SO(t_i) \cdot \pi$. The overall sentiment towards the celebrity c is, therefore, approximated as the geometric center of all terms c_i. We take the projection of the geometric center on y-axis as final sentiment value towards the celebrity.

To validate the effectiveness of sentiment extraction, we randomly select 1,000 tweets (500 positive and 500 negative tagged by our method) in Weibo sentiment dataset, and manually label each one of them. The result shows that the precision is 95.2% for positive class and 91.0% for negative class, which we believe is accurate enough for subsequent experiments. The basic statistics of Weibo sentiment datasets is presented in Table 1.

4 PROBLEM FORMULATION

In this section we formulate the problem of predicting sentiment links in heterogeneous information networks. For better illustration, we split the original heterogeneous network into the following three single-type networks:

Sentiment network. The directed *sentiment network* is denoted as $G_s = (V, S)$, where $V = \{1, ..., |V|\}$ represents the set of users (either ordinary users or celebrities) and $S = \{s_{ij} \mid i \in V, j \in V\}$ represents sentiment links among users. Each s_{ij} can take the value of $+1$, -1 or 0, representing that user i holds a positive, negative, or unobserved sentiment towards user j, respectively.

Social network. The directed *social network* is denoted as $G_r = (V, R)$, where $R = \{r_{ij} \mid i \in V, j \in V\}$ represents social links among users. Each r_{ij} can take the value of 1 or 0, representing that user i follows user j or not in the social network.

Profile network. We denote $\mathcal{A} = \{A_1, ..., A_{|\mathcal{A}|}\}$ the set of user's attributes, and $a_{kl} \in A_k$ the l-th possible value of attribute A_k. We take the union of all possible values of attributes and renumber them as $U = \bigcup A_k = \{a_j \mid j = 1, ..., \sum_k |A_k|\}$. Then the undirected bipartite *profile network* can be denoted as $G_p = (V, U, P)$, where $P = \{p_{ij} \mid i \in V, a_j \in U\}$ represents profile links between

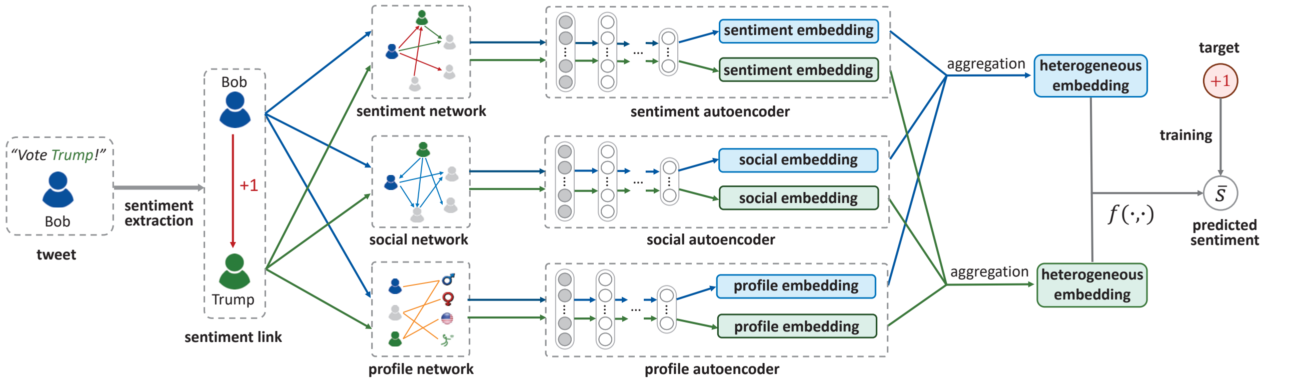


Fig. 3: Framework of the end-to-end SHINE model. To clearly demonstrate the model, we only show the encoder part of all the three autoencoders and leave out the decoder part in this figure.

users and attribute values. Each p_{ij} can take the value of 1 or 0, representing that user i possesses attribute value j or not.

The three networks are illustrated in Fig. 2.

Sentiment links prediction. We define the problem of predicting sentiment links in heterogeneous information networks as follows: Given the sentiment network G_s, social network G_r and profile network G_p, we aim to predict the sentiment of unobserved links between users in G_s.

5 SIGNED HETEROGENEOUS INFORMATION NETWORK EMBEDDING

In this section we introduce the proposed SHINE model. We first show the whole framework of SHINE. Then we present the details of the SHINE model, including how to extract user representation jointly from the three networks as well as the learning algorithm. At last we give some discussions on the model.

5.1 Framework

In this paper we propose an end-to-end SHINE model to predict sentiment links. The framework of SHINE is shown in Fig. 3. In general, the whole framework consists of three major components: sentiment extraction and heterogeneous networks construction (the left part), user representation extraction (the middle part), as well as representation aggregation and sentiment prediction (the right part). For each tweet mentioning a specific celebrity, we first calculate the associated sentiment (discussed in Section 3.1), and represent the user and the celebrity in this sentiment link by using their neighborhood information from the three constructed networks (introduced in Section 4). We then design three distinct autoencoders to extract short and dense embeddings from original sparse neighborhood-based representation respectively, and aggregate these three kinds of embeddings into final heterogeneous embedding. The predicted sentiment can thus be calculated by applying specific similarity measurement function (e.g., inner product or logistic regression) to the two heterogeneous embeddings, and the whole model can be trained based on the predicted sentiment and the target (i.e., the ground truth obtained in sentiment extraction step). In the following subsections we will introduce SHINE model in detail.

5.2 Sentiment Network Embedding

Given the sentiment graph $G_s = (V, S)$, for each user $i \in V$, we define its sentiment adjacency vector $\mathbf{x}_i = \{s_{ij} \mid j \in V\} \cup \{s_{ji} \mid j \in V\}$. Note that $\mathbf{x}_i$ fully contains the global incoming and outgoing sentiment information of user i. However, it is impractical to take $\mathbf{x}_i$ directly as the sentiment representation of user i, as the adjacency vector is too long and sparse for further processing. Recently, a lot of network embedding models [8, 17, 23, 26] are proposed, which aim to learn low-dimension representations of vertices while preserving the network structure. Among those models, deep autoencoder is proved to be one of state-of-the-art solutions, as it is able to capture highly nonlinear network structure by using deep models [26]. In general, autoencoder [20] is an unsupervised neural network model of codings aiming to learn a representation of a set of data. Autoencoder consists of two parts, the encoder and the decoder, which contains multiple nonlinear functions (layers) for mapping the input data to representation space and reconstructing original input from representation, respectively. In our SHINE model, we propose to use autoencoders for efficiently user representation learning.

Fig. 4 illustrates the autoencoder for sentiment network embedding. As shown in Fig. 4, the sentiment autoencoder maps each user to a low-dimension latent representation space and recover original information from latent representation by using multiple fully-connected layers. Given the input $\mathbf{x}_i$, the hidden representations for each layer are

$$\mathbf{x}_i^k = \sigma\left(\mathbf{W}_s^k \mathbf{x}_i^{k-1} + \mathbf{b}_s^k\right), \ k = 1, 2, ..., K_s, \tag{2}$$

where $\mathbf{W}_s^k$ and $\mathbf{b}_s^k$ are weight and bias parameters of layer k in the sentiment autoencoder, respectively, $\sigma(\cdot)$ is the nonlinear activation function, K_s is the number of layers of sentiment autoencoder, and $\mathbf{x}_i^0 = \mathbf{x}_i$. For simplicity, we denote $\mathbf{x}_i' = \mathbf{x}_i^{K_s}$ the reconstruction of $\mathbf{x}_i$.

The basic goal of the autoencoder is to minimize the reconstruction loss between input and output representations. Similar to [26], in SHINE model the reconstruction loss term of sentiment autoencoder is defined as

$$\mathcal{L}_s = \sum_{i \in V} \left\|(\mathbf{x}_i - \mathbf{x}_i') \odot \mathbf{l}_i\right\|_2^2, \tag{3}$$

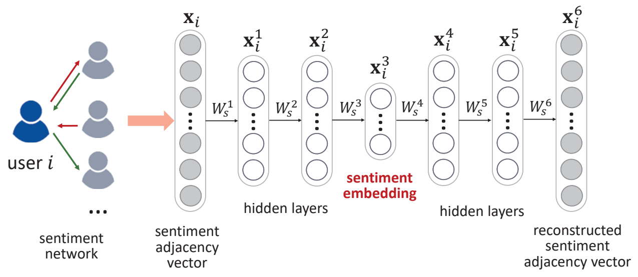


Fig. 4: Illustration of a 6-layer autoencoder for sentiment network embedding.

where $\odot$ denotes the Hadamard product, and $\mathbf{l}_i = (l_{i,1}, l_{i,2}, ..., l_{i,2|V|})$ is the sentiment reconstruction weight vector in which

$$l_{i,j} = \begin{cases} \alpha > 1, & if\ s_{ij} = \pm 1; \\ 1, & if\ s_{ij} = 0. \end{cases} \tag{4}$$

The meaning of the above loss term lies in that we impose more penalty to the reconstruction error of the non-zero elements than that of zero elements in input $\mathbf{x_i}$, as a non-zero s_{ij} carries more explicit sentiment information than an implicit zero s_{ij}. Note that the sentiment embedding of user i can be obtained from the layer $K_s/2$ in the sentiment autoencoder, and we denote $\widehat{\mathbf{x}}_i = \mathbf{x}_i^{K_s/2}$ the sentiment embedding of user i for simplicity.

5.3 Social Network Embedding

Similar to previous sentiment network embedding, we apply autoencoder to extract user representation from the social network. Given the social network $G_r = (V, R)$, for each user $i \in V$, we define its social adjacency vector $\mathbf{y}_i = \{r_{ij} \mid j \in V\} \cup \{r_{ji} \mid j \in V\}$, which fully contains the structural information of user i in the social network. The hidden representations of each layer in the social autoencoder are

$$\mathbf{y}_i^k = \sigma\left(\mathbf{W}_r^k \mathbf{y}_i^{k-1} + \mathbf{b}_r^k\right),\ k = 1, 2, ..., K_r, \tag{5}$$

where the meaning of notations are similar to those in Eq. (2). We also denote $\mathbf{y}_i' = \mathbf{y}_i^{K_r}$ the reconstruction of $\mathbf{y}_i$. Similarly, the reconstruction loss term of social autoencoder is

$$\mathcal{L}_r = \sum\nolimits_{i \in V} \left\|(\mathbf{y}_i - \mathbf{y}_i') \odot \mathbf{m}_i\right\|_2^2, \tag{6}$$

where $\mathbf{m}_i = (m_{i,1}, m_{i,2}, ..., m_{i,2|V|})$ is the social reconstruction weight vector in which if $r_{ij} = 1$, $m_{i,j} = \alpha > 1$, else $m_{i,j} = 1$. The social embedding of user i is denoted as $\widehat{\mathbf{y}}_i = \mathbf{y}_i^{K_r/2}$.

5.4 Profile Network Embedding

The profile network $G_p = (V, U, P)$ is an undirected bipartite graph which consists of two disjoint sets of users and attribute values. For each user $i \in V$, its profile adjacency vector is defined as $\mathbf{z}_i = \{p_{ij} \mid j \in U\}$. User i's hidden representations of each layer in the profile autoencoder are

$$\mathbf{z}_i^k = \sigma\left(\mathbf{W}_p^k \mathbf{z}_i^{k-1} + \mathbf{b}_p^k\right),\ k = 1, 2, ..., K_p, \tag{7}$$

where the meaning of notations are similar to those in Eq. (2). We also use the notation $\mathbf{z}_i'$ to denote the reconstruction of $\mathbf{z}_i$. Therefore, the reconstruction loss term of profile autoencoder is

$$\mathcal{L}_p = \sum\nolimits_{i \in V} \left\|(\mathbf{z}_i - \mathbf{z}_i') \odot \mathbf{n}_i\right\|_2^2, \tag{8}$$

where $\mathbf{n}_i$ is the profile reconstruction weight vector defined similarly to $\mathbf{m}_i$ in the previous subsection. The profile embedding of user i is denoted as $\widehat{\mathbf{z}}_i = \mathbf{z}_i^{K_p/2}$.

5.5 Representation Aggregation and Sentiment Prediction

Once we obtain the sentiment embedding $\widehat{\mathbf{x}}_i$, social embedding $\widehat{\mathbf{y}}_i$, and profile embedding $\widehat{\mathbf{z}}_i$ of user i, we can aggregate these embeddings into final heterogeneous embedding $\mathbf{e}_i$ by specific aggregation function $g(\cdot,\cdot,\cdot)$. We list some of the available aggregation functions as follows:

- *Summation* [34], i.e., $\mathbf{e}_i = \widehat{\mathbf{x}}_i + \widehat{\mathbf{y}}_i + \widehat{\mathbf{z}}_i$;
- *Max pooling* [28], i.e., $\mathbf{e}_i = element\text{-}wise\text{-}max(\widehat{\mathbf{x}}_i, \widehat{\mathbf{y}}_i, \widehat{\mathbf{z}}_i)$;
- *Concatenation* [23], i.e., $\mathbf{e}_i = \langle \widehat{\mathbf{x}}_i, \widehat{\mathbf{y}}_i, \widehat{\mathbf{z}}_i \rangle$.

Finally, given two users i and j as well as their heterogeneous embedding $\mathbf{e}_i$ and $\mathbf{e}_j$, the predicted sentiment $\bar{s}_{ij}$ can be calculated as $\bar{s}_{ij} = f(i, j)$, where $f(\cdot,\cdot)$ is specific similarity measurement function. For example:

- *Inner product* [3, 5], i.e., $\bar{s}_{ij} = \mathbf{e}_i^{\mathrm{T}} \mathbf{e}_j + b$, where b is a trainable bias parameter;
- *Euclidean distance* [26], i.e., $\bar{s}_{ij} = -\|\mathbf{e}_i - \mathbf{e}_j\|_2 + b$, where b is a trainable bias parameter;
- *Logistic regression* [17], i.e., $\bar{s}_{ij} = \mathbf{W}^{\mathrm{T}} \langle \mathbf{e}_i, \mathbf{e}_j \rangle + b$, where $\mathbf{W}$ and b are trainable weights and bias parameters.

We will study the choices of f and g in the experimental part.

5.6 Optimization

The complete objective function of SHINE model is as follows:

$$\begin{aligned}\mathcal{L} = &\sum\nolimits_{i \in V} \left\|(\mathbf{x}_i - \mathbf{x}_i') \odot \mathbf{l}_i\right\|_2^2 + \lambda_1 \sum\nolimits_{i \in V} \left\|(\mathbf{y}_i - \mathbf{y}_i') \odot \mathbf{m}_i\right\|_2^2 \\ &+ \lambda_2 \sum\nolimits_{i \in V} \left\|(\mathbf{z}_i - \mathbf{z}_i') \odot \mathbf{n}_i\right\|_2^2 + \lambda_3 \sum\nolimits_{s_{ij} = \pm 1} \left(f(\mathbf{e}_i, \mathbf{e}_j) - s_{ij}\right)^2 \\ &+ \lambda_4 \mathcal{L}_{reg},\end{aligned} \tag{9}$$

where λ_1, λ_2, λ_3 and λ_4 are balancing parameters. The first three terms in Eq. (9) are the reconstruction loss terms of sentiment autoencoder, social autoencoder, and profile autoencoder, respectively. The fourth term in Eq. (9) is the supervised loss term for penalizing the divergence between predicted sentiment and ground truth. The last term in Eq. (9) is the regularization term that prevents over-fitting, i.e.,

$$\mathcal{L}_{reg} = \sum_{k=1}^{K_s} \|\mathbf{W}_s^k\|_2^2 + \sum_{k=1}^{K_r} \|\mathbf{W}_r^k\|_2^2 + \sum_{k=1}^{K_p} \|\mathbf{W}_p^k\|_2^2 + \|f\|_2^2, \tag{10}$$

where $\mathbf{W}_s^k$, $\mathbf{W}_r^k$, $\mathbf{W}_p^k$ are the weight parameters of layer k in the sentiment autoencoder, social autoencoder, and profile autoencoder, respectively, and $\|f\|_2^2$ is the regularization penalty for similarity measurement function $f(\cdot,\cdot)$ (if appropriate).

We employ the AdaGrad [7] algorithm to minimize the objective functions in Eq. (9). In each iteration, we randomly select a batch of sentiment links from training dataset and compute the gradient of the objective function with respect to each trainable parameter respectively. Then we update each trainable parameter according to the AdaGrad algorithm till convergence.

5.7 Discussions

5.7.1 Asymmetry. Many real-world networks are directed, which implies that for two nodes i and j in the network, edges (i, j) and (j, i) may coexist and their values are not necessarily identical. A few recent studies have focused on this asymmetry issue [16, 36]. In this work, whether the basic SHINE model can characterize asymmetry depends on the choice of similarity measurement function f. Specifically, SHINE is capable of dealing with the direction of a link if and only if $f(i, j) \neq f(j, i)$ (e.g., logistic regression). However (and fortunately), even if we choose a symmetric function (e.g., inner product or Euclidean distance) as f, we can still easily extend the basic SHINE model to asymmetry-aware version by setting two distinct sets of autoencoders to extract representation of source node and target node respectively. From this point of view, in basic SHINE model the parameters of autoencoders are actually *shared* for source node and target node to alleviate over-fitting, and we can choose to explicitly distinguish the two sets of autoencoders for asymmetry reasons.

5.7.2 Cold start problem. A practical issue for network embedding is how to learn representations for newly arrived node, which is the cold start problem. Almost all existing models cannot work well in cold start scenario because they only use the information from the target network (e.g., sentiment network in this paper), which is not applicable for the newly arrived node who has little interaction with the existing target network. However, SHINE is free of the cold start problem, as it makes full use of side information and incorporate it naturally into the target network when learning user representations. We will further study the performance of SHINE in cold start scenario in the experiment part.

5.7.3 Flexibility. It is worth noticing that SHINE is also a framework with high flexibility. For any other new available side information of users (e.g., users' browsing history), we can easily design a new parallel processing component and "plug" it in the original SHINE framework to assist learning representation. Contrarily, we can also "pull out" social autoencoder or profile autoencoder from SHINE framework if such side information is unavailable. Besides, the flexibility of SHINE also lies in that one can choose different aggregation functions g and similarity measurement functions f, as discussed in Section 5.5.

6 EXPERIMENTS

In this section, we evaluate the performance of our proposed SHINE on real-world datasets. We first introduce the datasets, baselines, and parameter settings for experiments, then present the experimental results of SHINE and baselines.

6.1 Datasets

To comprehensively demonstrate the effectiveness of SHINE framework, we use the following two datasets for experiments:

- **Weibo-STC**: Our proposed *Weibo Sentiment Towards Celebrities* dataset consists of three heterogeneous networks with 12,814 users, 126,380 tweets, 71,268 social links and 37,689 profile values, of which the detail is presented in Section 3.
- **Wiki-RfA**: *Wikipedia Requests for Adminship* [30] is a signed network with 10,835 nodes and 159,388 edges, corresponding to votes cast by Wikipedia uses in election for promoting individuals to the role of administrator. A signed link indicates a positive or negative vote by one user on the promotion of another. Note that Wiki-RfA does not contain any side information of nodes, therefore, this dataset is used to validate the efficacy of the basic sentiment autoencoder in SHINE.

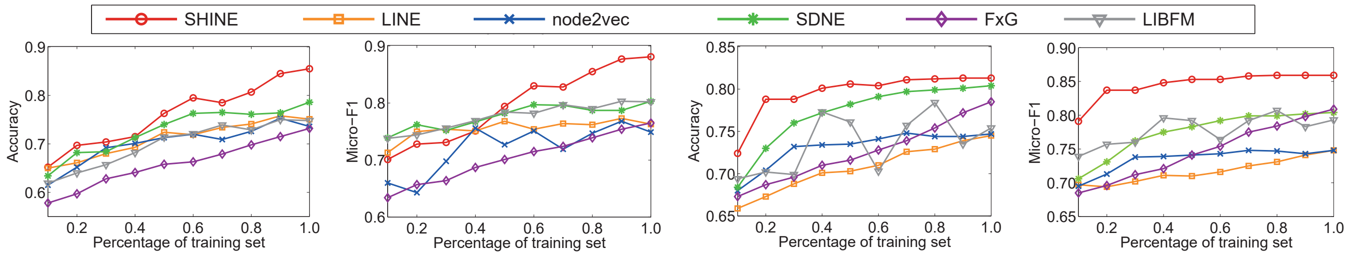
6.2 Baselines

We use the following five methods as baselines, in which the first three are network embedding methods, FxG is a signed link prediction approach, and LIBFM is a generic classification model. Note that the first three methods are not directly applicable to signed heterogeneous networks, so we use them to learn user representations from positive and negative part of each network respectively, and concatenate them to form the final embeddings. For FxG on Weibo-STC dataset, we only use the sentiment network as input because the FxG model cannot utilize the side information of nodes.

- **LINE**: Large-scale Information Network Embedding [23] defines loss functions to preserve the first-order and second-order proximity and learns representations of vertices.
- **Node2vec**: Node2vec [8] designs a biased random walk procedure to learn a mapping of nodes that maximizes the likelihood of preserving network neighborhoods of nodes.
- **SDNE**: Structural Deep Network Embedding [26] is a semi-supervised network embedding model using autoencoder to capture local and global structure of target networks.
- **FxG**: Fairness and Goodness [12] predicts the weights of edges in weighted signed networks by introducing two measures of node behavior: goodness (i.e., how much the node is liked by other nodes) and fairness (i.e., how fair the node is in rating other nodes' likeability).
- **LIBFM**: LIBFM [18] is a state-of-the-art feature based factorization model. In this paper, we use the concatenated one-hot vectors of users in three networks as input to feed LIBFM.

6.3 Parameter Setttings

We design a 4-layer autoencoder in SHINE for each network, in which the hidden layer is with 1,000 units and the embedding layer is with 100 units. Deeper architectures cannot further improve the performance but incur heavier computational overhead according to our experimental results. We choose concatenation as the aggregation function g and inner product as the similarity measurement function f. Besides, we set the reconstruction weight of non-zero elements $\alpha = 10$, the balancing parameters $\lambda_1 = 1$, $\lambda_2 = 1$, $\lambda_3 = 20$, and $\lambda_4 = 0.01$ for SHINE. We will study the sensitivity of these parameters in Section 6.6. For LINE, we concatenate the first-order and second-order representations to form the final 100-dimension embeddings for each node, and the total number of samples is 100 million. For node2vec, the number of embedding dimension is set as 100. For SDNE, the reconstruction weight of non-zero elements is 10 and the weight of first-order term is 0.05. For LIBFM, the dimensionality of the factorization machine is set as $\{1, 1, 0\}$ and



(a) Accuracy on Weibo-STC (b) Micro-F1 on Weibo-STC (c) Accuracy on Wiki-RfA (d) Micro-F1 on Wiki-RfA

Fig. 5: Accuracy and micro-F1 on Weibo-STC and Wiki-RfA for link prediction.

we use SGD method for training with learning rate of 0.5 and 200 iterations. Other parameters in these baselines are set as default.

In the following subsections, we conduct experiments on two tasks: link prediction and node recommendation.

6.4 Link Prediction

In link prediction setting, our task is to predict the sign of an unobserved link between two given nodes. As the existing links in the original network are known and can serve as the ground truth, we randomly hide 20% of links in the sentiment network and select a balanced test set (i.e., the number of positive links is the same as negative links) out of them, while use the remaining network to train SHINE as well as all baselines. We use *Accuracy* and *Micro-F1* as the evaluation metrics in link prediction task. For a more fine-grained analysis, we compare the performance while varying the percentage of training set from 10% to 100%. The result is presented in Fig. 5, from which we have the following observations:

- Fig. 5 shows that our methods SHINE achieves significant improvements in Accuracy and Micro-F1 over the baselines in both datasets. Specifically, in Weibo-STC, SHINE outperforms LINE, node2vec, and SDNE by 13.8%, 16.2%, and 8.78% respectively on Accuracy, and achieves 15.5%, 17.6%, 9.71% gains respectively on Micro-F1.
- Among the three state-of-the-art network embedding methods, SDNE performs best while LINE and node2vec show relatively poor performance. Note that SDNE also uses autoencoder to learning the embedding of nodes, which proves the superiority of autoencoder in extracting highly nonlinear representations of networks from a side.
- FxG performs much better in Wiki-RfA than in Weibo-STC. This is probably due to the following two reasons: 1) Unlike other methods, FxG cannot utilize the side information in Weibo-STC dataset. 2) Weibo-STC is sparser than Wiki-RfA, which is unfavorable to the computing of goodness and fairness of nodes in FxG model.
- Although LIBFM is not specially designed for network-structured data, it still achieves fine performance compared with other network embedding methods. However, during experiments we find that LIBFM is unstable and prone to parameters tuning. This can also be validated by the fluctuating curves of LIBFM in Fig. 5c and Fig. 5d.

To compare the performance of SHINE and baselines in cold start scenario, we construct a test set of newly arrived users for Weibo-STC, in which the associated ordinary user of each sentiment link dose not appear in the training set. We report Accuracy and Micro-F1 for all users and new users in Table 2. From the results in Table 2 it is evident that SHINE can still maintain a decent performance in the cold start scenario, as it fully exploits the information from social network and profile network to compensate for the lack of sentiment links. By comparison, the performance of other baselines degrades significantly in cold start scenario. Specifically, the Accuracy decreases by 2.46% for SHINE and by 11.58%, 11.28%, 15.14%, 17.90%, 14.57% respectively for LINE, node2vec, SDNE, FxG and LIBFM, which proves that SHINE are more capable of effectively transferring knowledge among heterogeneous information networks, especially in cold start scenario.

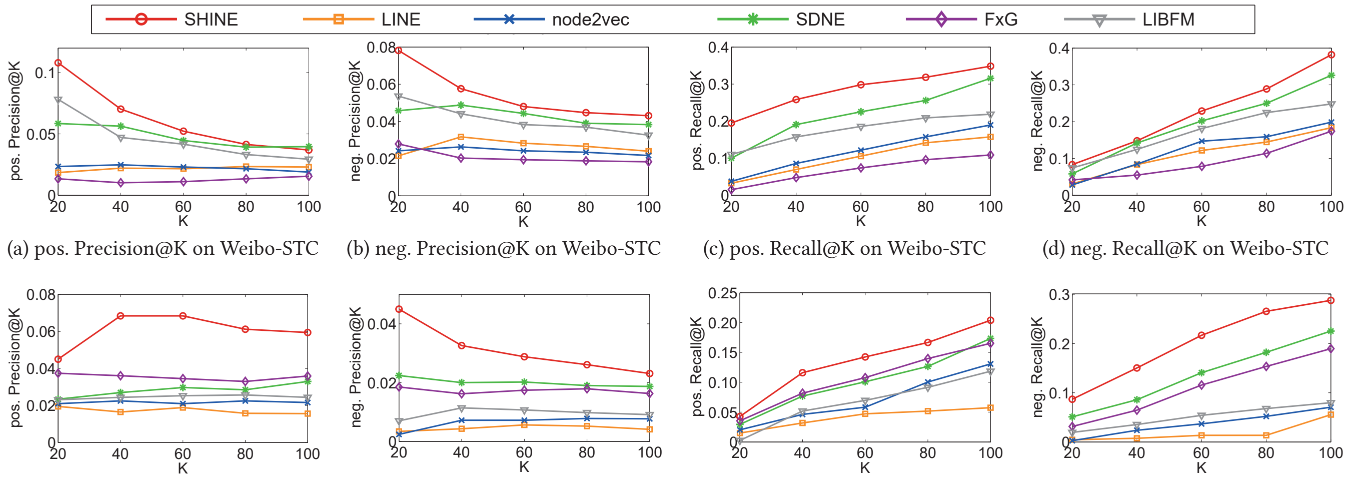
Table 2: Comparison of models in terms of Accuracy and Micro-F1 on Weibo-STC in cold start scenario.

Model	Accuracy		Micro-F1	
	all users	new users	all users	new users
SHINE	**0.855**	**0.834**	**0.881**	**0.858**
LINE	0.751	0.664	0.763	0.739
node2vec	0.736	0.653	0.749	0.667
SDNE	0.786	0.667	0.803	0.751
FxG	0.732	0.601	0.765	0.652
LIBFM	0.748	0.639	0.802	0.746

6.5 Node Recommendation

In addition to link prediction, we also conduct experiments on node recommendation, in which for each user we aim to recommend a set of users who have not been explicitly expressed attitude to but may be liked by the user. The performance of node recommendation can reveal the quality of learned representations as well. Specifically, for each user, we calculate his sentiment score toward all other users, and select K users with largest sentiment score for recommendation. For completeness, we recommend not only the nodes that a user may like but also the nodes that he may dislike. Therefore, we use positive and negative Precision@K and Recall@K respectively for evaluation in corresponding experimental scenarios. The results are shown in Fig. 6, which provides us the following observations:

- The curve of SHINE is almost consistently above the curves of baselines, which proves that SHINE can better learn the representations of heterogeneous networks and perform recommendation than baselines.



(a) pos. Precision@K on Weibo-STC (b) neg. Precision@K on Weibo-STC (c) pos. Recall@K on Weibo-STC (d) neg. Recall@K on Weibo-STC

(e) pos. Precision@K on Wiki-RfA (f) neg. Precision@K on Wiki-RfA (g) pos. Recall@K on Wiki-RfA (h) neg. Recall@K on Wiki-RfA

Fig. 6: Positive and negative Precision@K and Recall@K on Weibo-STC and Wiki-RfA for node recommendation.

Table 3: Accuracy on Weibo-STC w.r.t. the combinations of similarity measurement function and aggregate function.

f	g		
	Summation	Max pooling	Concatenation
Inner product	0.802	0.761	**0.855**
Euclidean distance	0.788	0.779	0.837
Logistic regression	0.816	0.782	0.842

- Negative precision is low than positive precision while negative recall is higher than positive recall for most methods. This is because negative links are far fewer than positive links in both datasets, which makes it easier to cover more negative links in the recommendation set.
- In general, the results of precision and recall on Weibo-STC is better than Wiki-RfA, which is in accordance with the results in link prediction. The reason lies in that Weibo-STC provides more side information which can greatly improve the quality of learned user representations.

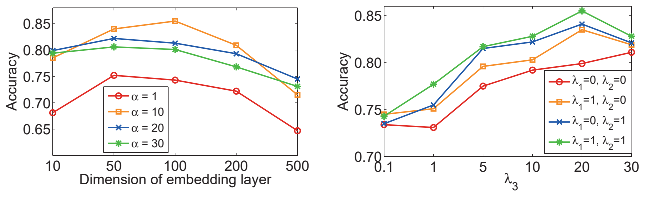
6.6 Parameters Sensitivity

SHINE involves a number of hyper-parameters. In this subsection we examine how the different choices of parameters affect the Accuracy of SHINE on Weibo-STC dataset. Except for the parameter being tested, all other parameters are set as default.

Similarity measurement function f and aggregation function g. We first investigate how the similarity measurement function f and aggregation function g affect the performance by testing on all combinations of f and g, and present the results in Table 3. It is clear that the combination of inner product and concatenation achieves the best Accuracy, while max pooling performs worst, which is probably due to the reason that concatenation preserves more information out of the three types of embeddings than summation and max pooling during embedding aggregation. It should also be noted that there is no absolute advantage of all the three f functions according to the results in Table 3.

Dimension of embedding layer and reconstruction weight of non-zero elements α. We also show how the dimension of embedding layer in the three autoencoders of SHINE and the hyper-parameter α affect the performance in Fig. 7a. We have the following two observations: 1) The performance is initially improved with the increase of dimension, because more bits in embedding layer can encode more useful information. However, the performance drops when the dimension further increases, as too large number of dimensions may introduce noises which mislead the subsequent prediction. 2) α controls the reconstruction weight of non-zero elements in autoencoders. When α is too small (e.g., $\alpha = 1$), SHINE will reconstruct the zero and non-zero elements without much discrimination, which deteriorates the performance because non-zero elements are more informative than zero ones. However, the performance will decrease if α gets too large (e.g., $\alpha = 30$), because large α will lead SHINE to totally ignore the dissimilarity (i.e., zero elements) among users.

Balancing parameters λ_1, λ_2, and λ_3. λ_1, λ_2, and λ_3 balance the loss terms of the objective function in Eq. (9). We treat λ_1 and λ_2 as binary parameters and vary the value of λ_3 to study the performance of SHINE. Note that whether λ_1 or λ_2 equals 1 indicates that whether we use the additional social information or profile information in link prediction. Therefore, the study of λ_1 and λ_2 can also be seen as to validate the effectiveness of social network embedding module and profile network embedding module. The result is presented in Fig. 7b, from which we can conclude that: 1) The curve of $\lambda_1 = 1, \lambda_2 = 0$ and $\lambda_1 = 0, \lambda_2 = 1$ are both above the curve of $\lambda_1 = 0, \lambda_2 = 0$, which demonstrates the significant gain by incorporating the social information and profile information (especially the latter) into the sentiment network. Moreover, combining both additional information can further improve the performance. 2) Increasing the value of λ_3 can greatly boost the accuracy, as SHINE will concentrate more on the prediction error rather than the reconstruction error. However, similar to other hyper-parameters, too large λ_3 is not satisfactory since it breaks the trade-off among loss terms in objective function.



(a) dim. of embedding layer and α (b) λ_1, λ_2, and λ_3

Fig. 7: Parameter sensitivity w.r.t. the dimension of embedding layers, α, λ_1, λ_2, and λ_3.

7 CONCLUSIONS

In this paper we study the problem of predicting sentiment links in absence of sentiment related content in online social networks. We first establish a labeled, heterogeneous, and entity-level sentiment dataset from Weibo due to the lack of explicit sentiment links. To efficiently learn from these heterogeneous networks, we propose Signed Heterogeneous Information Network Embedding (SHINE), a deep-learning-based network embedding framework to extract users' highly nonlinear representations while preserving the structure of original networks. We conduct extensive experiments to evaluate the performance of SHINE. Experimental results prove the competitiveness of SHINE against several strong baselines and demonstrate the effectiveness of usage of social relation and profile information, especially in cold start scenario.

ACKNOWLEDGMENTS

We thank our anonymous reviewers for their feedback and suggestions. This work was partially sponsored by the National Basic Research 973 Program of China under Grant 2015CB352403.

REFERENCES

[1] Mikhail Belkin and Partha Niyogi. 2001. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*, Vol. 14. 585–591.

[2] Felipe Bravo-Marquez, Eibe Frank, and Bernhard Pfahringer. 2015. Positive, negative, or neutral: Learning an expanded opinion lexicon from emoticon-annotated tweets. In *IJCAI 2015*, Vol. 2015. AAAI Press, 1229–1235.

[3] Shiyu Chang, Wei Han, Jiliang Tang, Guo-Jun Qi, Charu C Aggarwal, and Thomas S Huang. 2015. Heterogeneous network embedding via deep architectures. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 119–128.

[4] Wanxiang Che, Zhenghua Li, and Ting Liu. 2010. Ltp: A chinese language technology platform. In *Proceedings of the 23rd International Conference on Computational Linguistics: Demonstrations*. Association for Computational Linguistics, 13–16.

[5] Xin Dong, Lei Yu, Zhonghuo Wu, Yuxia Sun, Lingfeng Yuan, and Fangxi Zhang. 2017. A Hybrid Collaborative Filtering Model with Deep Structure for Recommender Systems. In *Thirty-First AAAI Conference on Artificial Intelligence*.

[6] Cícero Nogueira Dos Santos and Maira Gatti. 2014. Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts.. In *COLING*. 69–78.

[7] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, Jul (2011), 2121–2159.

[8] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 855–864.

[9] Ramanthan Guha, Ravi Kumar, Prabhakar Raghavan, and Andrew Tomkins. 2004. Propagation of trust and distrust. In *Proceedings of the 13th international conference on World Wide Web*. ACM, 403–412.

[10] Xiao Huang, Jundong Li, and Xia Hu. 2017. Label informed attributed network embedding. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 731–739.

[11] Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif Mohammad. 2014. NRC-Canada-2014: Detecting aspects and sentiment in customer reviews. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*. 437–442.

[12] Srijan Kumar, Francesca Spezzano, VS Subrahmanian, and Christos Faloutsos. 2016. Edge weight prediction in weighted signed networks. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*. IEEE, 221–230.

[13] Jérôme Kunegis, Stephan Schmidt, Andreas Lommatzsch, Jürgen Lerner, Ernesto W De Luca, and Sahin Albayrak. 2010. Spectral analysis of signed graphs for clustering, prediction and visualization. In *Proceedings of the 2010 SIAM International Conference on Data Mining*. SIAM, 559–570.

[14] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. 2010. Predicting positive and negative links in online social networks. In *Proceedings of the 19th international conference on World wide web*. ACM, 641–650.

[15] Thien Hai Nguyen and Kiyoaki Shirai. 2015. PhraseRNN: Phrase Recursive Neural Network for Aspect-based Sentiment Analysis.. In *EMNLP*. 2509–2514.

[16] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric transitivity preserving graph embedding. In *Proc. of ACM SIGKDD*. 1105–1114.

[17] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 701–710.

[18] Steffen Rendle. 2012. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)* 3, 3 (2012), 57.

[19] Hassan Saif. 2015. *Semantic Sentiment Analysis of Microblogs*. Ph.D. Dissertation. The Open University.

[20] Ruslan Salakhutdinov and Geoffrey Hinton. 2009. Semantic hashing. *International Journal of Approximate Reasoning* 50, 7 (2009), 969–978.

[21] Jiliang Tang, Shiyu Chang, Charu Aggarwal, and Huan Liu. 2015. Negative link prediction in social media. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*. ACM, 87–96.

[22] Jiliang Tang, Yi Chang, Charu Aggarwal, and Huan Liu. 2016. A survey of signed network mining in social media. *ACM Computing Surveys (CSUR)* 49, 3 (2016), 42.

[23] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. ACM, 1067–1077.

[24] Joshua B Tenenbaum, Vin De Silva, and John C Langford. 2000. A global geometric framework for nonlinear dimensionality reduction. *science* 290, 5500 (2000), 2319–2323.

[25] Peter D Turney. 2002. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 417–424.

[26] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1225–1234.

[27] Hongwei Wang, Jia Wang, Miao Zhao, Jiannong Cao, and Minyi Guo. 2017. Joint-Topic-Semantic-aware Social Recommendation for Online Voting. In *Proceedings of the 26th ACM International Conference on Conference on Information and Knowledge Management*. ACM, 347–356.

[28] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2015. Learning hierarchical representation model for nextbasket recommendation. In *Proceedings of the 38th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 403–412.

[29] Suhang Wang, Jiliang Tang, Charu Aggarwal, Yi Chang, and Huan Liu. 2017. Signed network embedding in social media. In *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM, 327–335.

[30] Robert West, Hristo S Paskov, Jure Leskovec, and Christopher Potts. 2014. Exploiting social network structure for person-to-person sentiment analysis. *arXiv preprint arXiv:1409.2450* (2014).

[31] Jihang Ye, Hong Cheng, Zhe Zhu, and Minghua Chen. 2013. Predicting positive and negative links in signed social networks by transfer learning. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 1477–1488.

[32] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. 2014. Personalized entity recommendation: A heterogeneous information network approach. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 283–292.

[33] Shuhan Yuan, Xintao Wu, and Yang Xiang. 2017. SNE: Signed Network Embedding. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 183–195.

[34] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 353–362.

[35] Quan Zheng and David B Skillicorn. 2015. Spectral embedding of signed networks. In *Proceedings of the 2015 SIAM International Conference on Data Mining*. SIAM, 55–63.

[36] Chang Zhou, Yuqiong Liu, Xiaofei Liu, Zhongyi Liu, and Jun Gao. 2017. Scalable Graph Embedding for Asymmetric Proximity.. In *AAAI*. 2942–2948.