SSDMV: Semi-supervised Deep Social Spammer Detection by Multi-View Data Fusion

Conference Paper · October 2018

DOI: 10.1109/ICDM.2018.00040

6 authors, including:

Chaozhuo Li
Beihang University (BUAA)
24 PUBLICATIONS **129** CITATIONS
SEE PROFILE

Senzhang Wang
Nanjing University of Aeronautics & Astronautics
107 PUBLICATIONS **770** CITATIONS
SEE PROFILE

Lifang He
Lehigh University
78 PUBLICATIONS **958** CITATIONS
SEE PROFILE

Philip S. Yu
University of Illinois at Chicago
1,529 PUBLICATIONS **69,382** CITATIONS
SEE PROFILE

SSDMV: Semi-supervised Deep Social Spammer Detection by Multi-View Data Fusion

Chaozhuo Li
Beihang University
Beijing, China
lichaozhuo@buaa.edu.cn

Senzhang Wang
Nanjing University of Aeronautics and Astronautics
Beijing, China
szwang@nuaa.edu.cn

Lifang He
Cornell University
New York, USA
lifanghescut@gmail.com

Philip S. Yu
Fudan University
University of Illinois at Chicago
Chicago, USA
psyu@uic.edu

Yanbo Liang
Hortonworks Incorporation
San Jose, USA
ybliang8@gmail.com

Zhoujun Li
Beihang University
Beijing, China
lizj@buaa.edu.cn

***Abstract*—The explosive use of social media makes it a popular platform for malicious users, known as social spammers, to overwhelm legitimate users with unwanted content. Most existing social spammer detection approaches are supervised and need a large number of manually labeled data for training, which is infeasible in practice. To address this issue, some semi-supervised models are proposed by incorporating side information such as user profiles and posted tweets. However, these shallow models are not effective to deeply learn the desirable user representations for spammer detection, and the multi-view data are usually loosely coupled without considering their correlations. In this paper, we propose a Semi-Supervised Deep social spammer detection model by Multi-View data fusion (SSDMV). The insight is that we aim to extensively learn the task-relevant discriminative representations for users to address the challenge of annotation scarcity. Under a unified semi-supervised learning framework, we first design a deep multi-view feature learning module which fuses information from different views, and then propose a label inference module to predict labels for users. The mutual refinement between the two modules ensures SSDMV to be able to both generate high quality features and make accurate predictions. Empirically, we evaluate SSDMV over two real social network datasets on three tasks, and the results demonstrate that SSDMV significantly outperforms the state-of-the-art methods.**

***Index Terms*—Social Spammer Detection, Deep Learning, Semi-supervised Learning**

I. Introduction

Microblogging websites, such as Twitter and Sina Weibo, have become prevalent social platforms for information dissemination and sharing [1], where users can interact with friends, post real-time short messages and share their opinions on trending topics. But on the other side of the coin, the popularity of social networks also draws growing attention of social spammers to unfairly overwhelm legitimate users with unwanted or fake contents by spreading advertisements, hijacking trending topics and abusing follow or mention functions. The malicious behavior of social spammers has not only significantly hindered the development of social networks, but also greatly threatened information security [2]. Therefore, social spammer detection has become an important research issue and attracted increasing research interest recently.

Senzhang Wang is the corresponding author of this paper.

Most existing social spammer detection methods are supervised, which need a large number of manually labeled samples to train a classification model [1]–[8]. However, for a social network with hundreds of millions users like Twitter, it is extremely time consuming and requires high labor cost to obtain enough annotations. To address this issue, unsupervised spammer detection methods are proposed by utilizing anomaly detection [9] and community clustering models [10]. Such methods can automatically filter spammers but often suffer from high false positive rate as no annotations are utilized [9], [11]. Compared to supervised and unsupervised methods, semi-supervised methods can utilize the unlabeled data to help capture the shape of the underlying data distribution, which are more promising to detect social spammers effectively and efficiently. Yu et al. [12] introduced a semi-supervised nonnegative matrix factorization model to filter spammers by utilizing the message matrix and social relation matrix. Li et al. [13], [14] proposed a self-training model to combine a supervised classification model with a ranking scheme.

To address the annotation scarcity issue, existing semi-supervised detection methods generally adopt a self-training strategy to identify some high confidence test samples to enrich the training set [13], [14]. However it may introduce and enlarge noise because early mistakes could reinforce themselves [15]. Different from existing works, we focus on learning annotation-guided high quality latent representations for social users. The motivation is that in the latent representation space, social users with similar properties (e.g., relations, tweets) as well as the same labels should be closely distributed and can be easily clustered into the same group. Then it would be much easier to learn a classifier to accurately distinguish spammers from legitimate users. Yet, there are three major challenges that we face while learning desired latent representations for social users.

First, it is challenging to learn a user representation that can comprehensively reflect his/her characteristics from different views. Social network users are generally associated with the multi-view data including social relations (view R for short), demographic information (view D), tweet text (view T) and used symbols (view S) such as hashtags, mentions and urls. Data of all the views as a whole characterize the social users, and ignoring any of them may potentially lead to information loss. Existing works [1], [6], [12] only utilized a part of the above multi-view data, leading to undesirable performance. The difficulty of multi-view representation learning for social users lies in the characteristics of high-dimension and high non-linearity [16] of the social media data, such as social relations and texts. Existing spammer detection methods are usually shallow models and are not capable to accurately approximate these highly nonlinear information. Hence, a deep model with stronger feature learning capacity is needed.

Second, the data in some views of partial users can be sparse and noisy, leading to the challenge of learning desired representations for them. For example, some Twitter users have few friends or post a very limited number of tweets, and the posted tweets are usually unstructured texts with linguistic noise and idiosyncratic style [17], [18]. Previous study [19] shows that the multi-view data in social networks are highly correlated and complementary to each other, and extracting features from them mutually rather than separately will contribute to learning more robust features. Intuitively, if two users are similar in the social relation view with many common neighbors, they are more likely to have similar interests and post similar tweets due to the social theory of homophily [20]. It motivates us to complement the sparse or noisy data of one view by incorporating the correlated information of other views, but how to do it exactly is not straight-forward. Existing methods [4], [13] extract features from different views separately and then loosely couple them together by directly concatenating the extracted features as the final representation. MVSD [3] utilizes matrix factorization to learn latent features in each view independently and linearly combine them into the final feature vector. Although multi-view data is incorporated, its feature learning modules in different views are separated and thus cannot capture the cross-view correlations. Hence a robust feature learning model with effective data fusion mechanism is needed.

Third, it is also challenging to learn the task-relevant features in a semi-supervised learning setting. Most existing spammer detection methods manually extract user features that might be useful in an unsupervised way [4], [5], [13], [14]. However, unsupervised feature learning cannot precisely capture the most discriminative details for social spammer detection. The empirically extracted features may be task irrelevant and burden the learning of the classifier, especially when annotations are scarce. Incorporating annotations into feature learning will contribute to generating more discriminative and task-relevant features. However, it is challenging for existing methods to perform an annotation embedded feature learning as they generally consider the feature learning and label prediction as two separate phases rather than an integrated whole [6], [8]. Hence a unified semi-supervised framework which can perform label inference and feature learning simultaneously is also needed.

To address the above challenges, we propose a novel Semi-Supervised Deep social spammer detection model SSDMV by fusing Multi-View data. SSDMV consists of two mutually complementary modules: the multi-view feature learning module and the label inference module. Specifically, we propose a Correlated-Ladder-Networks (CLN) model to perform multi-view feature learning. Ladder network is an auto-encoder deep model with a denoising mechanism to construct clean representations from the corrupted ones, which contributes to generate more robust features and prevent over-fitting [21]. Considering its superior feature learning and denoising ability, we select ladder network as the basic model to learn features for each view. Different from previous works that perform feature learning in multiple views separately, we propose a multi-view data fusion component called filter gate to integrate multiple ladder networks whose input are highly correlated. Information from other views will flow into the ladder network in one view through the filter gate to provide complementary information. The filter gate can also control how much information should be fused among these views. In the label inference module, the latent features learned from the CLN model are fed into a MLP (Multi-Layer Perceptron) model to fit the annotations. CLN and MLP models are jointly trained under a unified semi-supervised learning framework. The classification loss from MLP will be propagated back to CLN to guide it better capturing task-relevant features. We conduct extensive experiments over two datasets on three tasks. Especially, on an open Twitter dataset, SSDMV achieves 0.9 F1-score with only 200 labeled samples, which is comparable with a state-of-the-art supervised method with 16,000 labeled samples.

We summarize our main contributions as follows:

- To the best of our knowledge, we are the first to propose a semi-supervised deep learning model for social spammer detection. As a deep model, SSDMV is more powerful in multi-view feature learning, and can effectively capture the task-relevant details in the multi-view data.
- We propose a novel feature learning model CLN to effectively fuse information from different views to learn more robust features, which contributes to alleviate the effects of sparse and noisy data in social media.
- We integrate feature learning and label inferencing into a unified semi-supervised learning framework. The mutual refinement between the two modules ensures its effectiveness in both generating high quality features and achieving better prediction results.

The rest of this paper is organized as follows. Section 2 summarizes the related works. Then we formally define the studied problem in Section 3. Section 4 introduces the proposed SSDMV model. Section 5 presents the experimental results. Finally, we conclude this work in section 6.

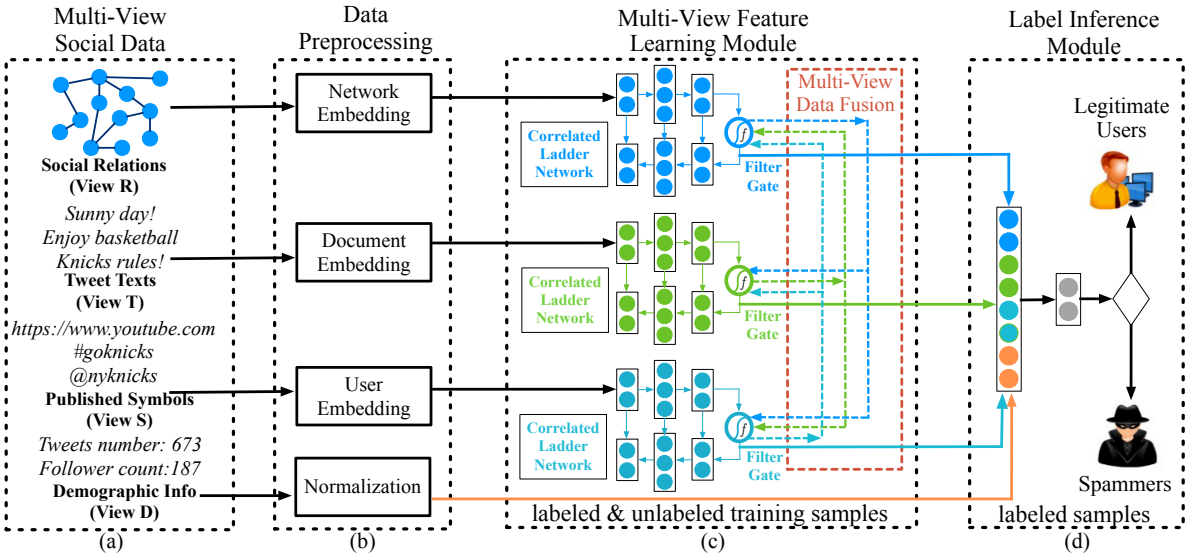


Fig. 1: An overview of our SSDMV approach.

II. Related Work

Most of existing social spammer detection methods are supervised, which need completely labeled samples to train a classifier [1]–[8]. Hu et al. proposed a set of Laplacian matrix factorization based methods to incorporate social relations and the generated contents [1], [6], [22]. Shobeir et al. [8] designed a hinge-loss Markov random field model to detect spammers in evolving multi-relational social networks. Supervised methods require high labor cost to annotate the data. To alleviate the dependency on annotations, a set of unsupervised social spammer detection methods are proposed [9], [10]. They generally utilized clustering or ranking method to automatically locate spammers. Without the guiding of annotations, these methods usually cannot achieve desirable performance.

Recently several semi-supervised models [12]–[14] are proposed by utilizing the underlying structures of unlabeled samples and the available annotations. However, there are three major limitations for existing semi-supervised methods. First, most of they consider partial social media data. For example, the published symbols are largely ignored in [13], [14]. Second, these models require manually extracted features or learn features in an unsupervised manner [12], which cannot ensure the generated features contribute to filter spammers. Third, these models are all shallow models, which cannot effectively handle the highly nonlinear social media data.

Note that, social spammer detection is different from spam detection [23], [24]. Generally spam detection aims to decide whether a review, an email or a tweet is malicious, while social spammer detection aims to filter malicious social users who are associated with multi-view data. thus spam detection methods usually cannot be applied for social spammer detection. For example, Wu et al. [24] proposed a deep model to filter spam tweets according to the linguistic proprieties, but it cannot be directly extended to the multi-view data case (e.g., network topology and contents) to detect social spammers.

III. Problem Definition

We represent a social network G as: $G = (X, R, D, T, S)$ where $X = \{x_1, x_2, \ldots, x_m\}$ denotes the social user set and m is the number of users. The multi-view data of a user can be modeled as the social relation matrix $R \in \mathbb{R}^{m\times r}$, the demographic matrix $D \in \mathbb{R}^{m\times d}$, the tweet text matrix $T \in \mathbb{R}^{m\times t}$ and the published symbol matrix $S \in \mathbb{R}^{m\times s}$, in which r, d, t, s are the dimensions of the data in the corresponding views. Each row in the matrices R, D, T, S denotes the representation vector of a specific user in each view. For example, $R_i \in \mathbb{R}^{1\times r}$ represents the social relations (followees or followers) of user x_i. We use $Y_l \in \mathbb{R}^{|X_l|\times c}$ to denote the available identity label matrix, where X_l is the set of labeled users and c is the number of identity labels. We focus on classifying users as spammers or legitimate users and thus in this paper $c = 2$. With the above notations, we formally define the studied problem as follows:

Definition 1: ***Multi-View Semi-Supervised Spammer Detection****: Given the training user set $X_{tr} \subset X$ in a social network with their social relation matrix R_{tr}, demographic matrix D_{tr}, tweet text matrix T_{tr}, published symbol matrix S_{tr}, a small number of labeled set $X_l \subset X_{tr}$ with its identity matrix Y_l and the unlabeled training set $X_u = X_{tr} - X_l$, we aim to learn a classifier to accurately assign labels for users in the test set X_{te} as spammers or legitimate users.*

IV. Methodology

Figure 1 provides an overview of our approach. Given an input multi-view data (a), we first perform a preprocessing step (b), which extracts feature vectors from each view using different embedding techniques. Here we need to normalize the data in all views such that one view would not dominant the others. Then we input these features into

the CLN model (correlated-ladder-networks) to facilitate the multi-view feature learning (c), with each correlated ladder network associated to one view data. The latent features of other views flow into the bottleneck layer of each correlated ladder network to provide complementary information. Note that here we add a filter gate into each correlated ladder network to control how much information will flow across other views. Finally, in the label inference module (d), the learned multi-view features are concatenated together with the demographical vector as the inputs of the MLP model to fit the annotations. The output 2-dimensional vector of MLP represents the predicted probabilities of the user belonging to a spammer or a legitimate user.

Note that in the training process, the multi-view feature learning module takes all the training samples as input, including both the labeled and unlabeled ones; while the label inference module only considers the labeled training samples, which forms a unified semi-supervised learning framework. In the feed forward step, the generated high quality multi-view features are fed into MLP model. In the error back propagation step, the classification loss from the label inference module along with the reconstruction loss from the feature learning module will both guide the parameter updating of the CLN to learn better task-relevant features.

A. Data Preprocessing

For the demographic features, we do not use the user profiles that are given by users themselves such as gender and age, because such information may not be true. Here we choose the number of followers and followees, and the count of posted tweets as the demographic features. We normalize these features and directly feed them into the label inference module. Data of other views are much more complex and unstructured. We utilize the popular embedding methods to transform the raw data into low-dimensional distributed vectors, which has been a widely employed data preprocessing strategy in natural language processing [25], graph mining [26], [27] and image processing [28]. Specifically, for the social relation view R, we construct an undirected user following network and utilize Node2Vec [29] to learn the node representations. For the text view T, after segmentation and stemming, we consider all the tweets posted by the same user as a document and use Doc2Vec [30] to learn the feature vector of view T. For the view S, we construct a user-symbol network. If a user has posted a symbol (e.g. url and hashtag), there will be an edge between them in the network. We assume that users publishing similar symbols tend to be similar. We feed this network into Node2Vec to learn the user representations in view S.

B. Multi-View Feature Learning Module

As social media data is often unstructured with noise [17], we select ladder network [31], [32] as the basic model to perform denoising and feature learning, which aims to reconstruct clean representations from the noisy ones. Since the original ladder network is designed for unsupervised learning in a single view, we modify the model structure to make it feasible

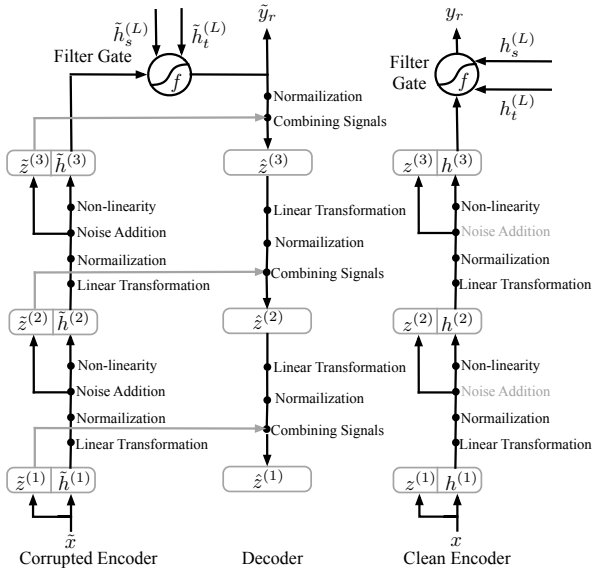


Fig. 2: Architecture of a correlated ladder network.

to the multi-view data. The proposed CLN model includes three correlated ladder networks with each one corresponding to one view of data. We only introduce the model details in the social relation view R for brevity, and the models for other views can be obtained in a similar way. We use x to represent the prepossessed embedding vector of a user in the view R.

As shown in Figure 2, the correlated ladder network includes four components: corrupted encoder, clean encoder, decoder and filter gate. A good feature learning model should be robust to the partial destruction of the input [33]. Hence, the corrupted encoder introduces noise into each layer to generate noisy representations, while with shared parameters, clean encoder generates clean representations without noise. Decoder generates reconstructed representations from the noisy ones. The filter gate introduces complementary information from other views into the corrupted and clean encoders. The correlated ladder network is formally defined as follows:

$$\tilde{z}^{(1)}, \cdots, \tilde{z}^{(l)}, \cdots, \tilde{z}^{(L)}, \tilde{y} = \text{CorruptedEncoder}(\tilde{x}),$$
$$z^{(1)}, \cdots, z^{(l)}, \cdots, z^{(L)}, y = \text{CleanEncoder}(x),$$
$$\hat{z}^{(1)}, \cdots, \hat{z}^{(l)}, \cdots, \hat{z}^{(L)} = \text{Decoder}(\tilde{z}^{(1)}, \cdots, \tilde{z}^{(L)}, \tilde{y}).$$

The variables x, $\tilde{x}$, y, and $\tilde{y}$ are the clean input, the corrupted input, the clean final feature vector, and the noisy final feature vector respectively. The variables $z^{(l)}$, $\tilde{z}^{(l)}$, and $\hat{z}^{(l)}$ are the clean hidden representation, its noisy version, and its reconstructed version at layer l. With the objective of minimizing the difference between the reconstructed representation $\hat{z}^{(l)}$ and the clean version $z^{(l)}$, the ladder network is able to denoise and generate robust representations. Here the filter gate is integrated into the encoders for clarity.

1) Corrupted Encoder: As shown in the left part of Figure 2, we add noise and perform the non-linear transformation in each layer. We integrate Gaussian noise with mean 0 and variance σ^2 into the clean input x to generate the corrupted

input $\tilde{x}$: $\tilde{x} = x + \mathcal{N}(0, \sigma^2)$ [21], [32]. Each layer l in the corrupted encoder includes two vectors: the corrupted hidden representation vector $\tilde{z}^{(l)}$ which will be used in decoder, and the post-activation vector $\tilde{h}^{(l)}$ which introduces non-linear transformation. In the first layer we have $\tilde{z}^{(1)} = \tilde{h}^{(1)} = \tilde{x}$. From layer $l-1$ to l, we perform the following steps:

Linear Transformation. It carries the dimensionality expansion/reduction and feature weights learning to convert the input vector from dimension d_{l-1} to dimension d_l:

$$\tilde{z}^{(l)}_{pre} = W^{(l)} \cdot \tilde{h}^{(l-1)},$$

where $W^{(l)}$ is the weight matrix from layer $l-1$ to layer l.

Batch Normalization. We employ the normalization for each training batch [34] to accelerate the speed and alleviate the effect of different data distributions in different batches:

$$\begin{aligned}\tilde{\mu}^{(l)} &= mean(\tilde{z}^{(l)}_{pre}),\\ \tilde{\sigma}^{(l)} &= stdv(\tilde{z}^{(l)}_{pre}),\\ \tilde{z}^{(l)}_{bn} &= \frac{\tilde{z}^{(l)}_{pre} - \tilde{\mu}^{(l)}}{\tilde{\sigma}^{(l)}},\end{aligned}$$

where $mean$ and $stdv$ functions calculate the batch mean value and standard deviation for the input $\tilde{z}^{(l)}_{pre}$.

Noise Addition. As same as the first layer, we add noise to each layer to simulate the partial destruction of the input:

$$\tilde{z}^{(l)} = \tilde{z}^{(l)}_{bn} + \mathcal{N}(0, \sigma^2),$$

where σ is the hyper-parameter to control the strength of noise.

Non-linearity. It introduces non-linear transformation to the input to capture the non-linear information in social data:

$$\tilde{h}^{(l)} = \phi\big(\gamma^{(l)}(\tilde{z}^{(l)} + \beta^{(l)})\big),$$

where ϕ is the activation function. For the hidden layers we use rectified linear unit (ReLU) function, while for the output layer we use softmax function to scale the features to the range (0,1). Parameters $\gamma^{(l)}$ and $\beta^{(l)}$ are responsible for scaling and shifting to reduce the *Internal Covariate Shift* [34].

2) Clean Encoder: Compared to the corrupted encoder, the clean encoder has the similar steps but without the noise addition part as shown in the right part of Figure 2. Clean encoder shares the same parameters with the corrupted encoder, which generates reconstruction targets for the decoder and provides features for the model predicting. In the first layer we have $z^{(1)} = h^{(1)} = x$, and from layer $l-1$ to l, we perform

$$\begin{aligned}z^{(l)}_{pre} &= W^{(l)} \cdot h^{(l-1)},\\ \mu^{(l)} &= mean(z^{(l)}_{pre}),\\ \sigma^{(l)} &= stdv(z^{(l)}_{pre}),\\ z^{(l)}_{bn} &= \frac{z^{(l)}_{pre} - \mu^{(l)}}{\sigma^{(l)}},\\ h^{(l)} &= \phi\big(\gamma^{(l)}(z^{(l)} + \beta^{(l)})\big).\end{aligned}$$

Note that the clean encoder shares the same parameters $W^{(l)}$, $\gamma^{(l)}$ and $\beta^{(l)}$ with the corrupted encoder.

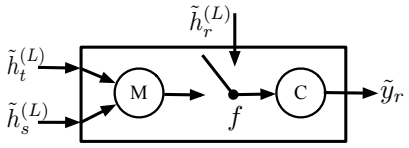


Fig. 3: The illustration of the filter gate in view R.

3) Filter Gate: A straight forward data fusing method is to incorporate all the information from other views, which may loss the unique characteristics of each view. Hence we need to design a component like a gate or switch to control how much information should flow across these views. As shown in Figure 3, we add a filter gate to fuse correlated information from other views. This makes the CLN model different from the original ladder network which directly feeds the after-activation vector $\tilde{h}^{(L)}$ into the decoder.

After the step of corrupted encoder in each view, we can get three after-activation vectors $\tilde{h}_r^{(L)}$, $\tilde{h}_t^{(L)}$ and $\tilde{h}_s^{(L)}$ from the top layers L of corrupted encoders in views R, T and S, respectively. For each view, we design a filter gate f to fuse the information from other views. Here we still take view R as an example. The filter gate includes two activation functions M and C, and a switch f. The function M controls the way of merging the two vectors from other views into a vector:

$$\tilde{h}_{ts} = \tanh(W_T \cdot \tilde{h}_t^{(L)} + W_S \cdot \tilde{h}_s^{(L)}),$$

where W_T and W_S are trainable weights. $\tilde{h}_{ts}$ is the weighted combination of the information from the other two views T and S. The switch f controls how much of the data from the other views will be fused with the current view vector $\tilde{h}_r^{(L)}$:

$$\tilde{f}_r = sigmoid(U_T \cdot \tilde{h}_{ts} + U_S \cdot \tilde{h}_r^{(L)}),$$

where U_T and U_S are trainable weights. The output value of the sigmoid function is in the range (0,1), representing how much information of each component should be let through. A value of zero means "let nothing through", while a value of one means "let everything through". The function C utilizes switch f to combine features from own view and other views:

$$\tilde{y}_r = (1 - \tilde{f}_r) \odot \tilde{h}_{ts} + \tilde{f}_r \odot \tilde{h}_r^{(L)},$$

where $\odot$ is the element-wise multiplication operator, and $\tilde{y}_r$ is the final representation learned from corrupted encoder in view R. Similarly, the outputs of clean encoders also go through a filter gate. The only difference is that they use clean inputs:

$$\begin{aligned}h_{ts} &= \tanh(W_T \cdot h_t^{(L)} + W_S \cdot h_s^{(L)}),\\ f_r &= sigmoid(U_T \cdot h_{ts} + U_S \cdot h_r^{(L)}),\\ y_r &= (1 - f_r) \odot h_{ts} + f_r \odot h_r^{(L)},\end{aligned}$$

where parameters W_T, W_S, U_T and U_S are the same to the filter gate used in the corrupted encoder.

4) Decoder: The decoder takes the noisy representations from corrupted encoder as its input, and aims to denoise and reconstruct clean representations from the corrupted inputs. The decoder in a normal auto-encoder model [35] only considers the output vector from the top layer in encoder. In the ladder network, however, each layer of the corrupted encoder is connected to its corresponding layer in the decoder through skip connections, which enables the higher layer features focusing on more abstract and task-specific features [32]. At each layer of the decoder, the signal $\hat{z}^{(l+1)}$ from the upper layer and the noisy signal $\tilde{z}^{(l)}$ are combined into the reconstructed representation $\hat{z}^{(l)}$ by the following equations:

$$\begin{aligned}\hat{u}_{pre}^{(l)} &= V^{(l)} \cdot \hat{z}^{(l+1)},\\ \hat{\mu}^{(l)} &= mean(\hat{u}_{pre}^{(l)}),\\ \hat{\sigma}^{(l)} &= stdv(\hat{u}_{pre}^{(l)}),\\ \hat{u}^{(l)} &= \frac{\hat{u}_{pre}^{(l)} - \hat{\mu}^{(l)}}{\hat{\sigma}^{(l)}},\\ \hat{z}^{(l)} &= g_l(\tilde{z}^{(l)}, \hat{u}^{(l)}),\end{aligned}$$

where $V^{(l)}$ is the weight matrix from layer $l+1$ to l. Function $g(\cdot,\cdot)$ is the combinator which combines the vertical $\hat{u}^{(l)}$ and horizontal $\tilde{z}^{(l)}$. Here we use the vanilla combinator [32]:

$$\begin{aligned}mu^{(l)} &= a_1^{(l)} \odot sigmoid(a_2^{(l)} \odot \hat{u}^{(l)} + a_3^{(l)}) + a_4^{(l)} \odot \hat{u}^{(l)} + a_5^{(l)},\\ v^{(l)} &= a_6^{(l)} \odot sigmoid(a_7^{(l)} \odot \hat{u}^{(l)} + a_8^{(l)}) + a_9^{(l)} \odot \hat{u}^{(l)} + a_{10}^{(l)},\\ \hat{z}^{(l)} &= (\tilde{z}^{(l)} - mu^{(l)}) \odot v^{(l)} + mu^{(l)},\end{aligned}$$

where $a_i^{(l)}$ is the weight vector. Following [32], $a_2^{(l)}$ and $a_7^{(l)}$ are initialized to 1 and the others are initialized to 0.

Reconstruction Loss. For a single view like view R, the loss function of its correlated ladder network is the weighted sum of the reconstruction errors in each layer:

$$\mathcal{L}^r = \sum_{1 \leqslant l \leqslant L} w_l \cdot ReconsError(z^{(l)}, \hat{z}^{(l)}), \tag{1}$$

where w_l is the weight of the reconstruction error, and

$$ReconsError(z^{(l)}, \hat{z}^{(l)}) = \| \frac{\hat{z}^{(l)} - \mu^{(l)}}{\sigma^{(l)}} - z^{(l)} \|^2,$$

where the normalization is to avoid the effect of the noise introduced by batch normalization. The loss function of CLN is the sum of the reconstruction errors from three views:

$$\mathcal{L}_{CLN} = \sum_{x_i \in X_{tr}} (\mathcal{L}_i^r + \mathcal{L}_i^s + \mathcal{L}_i^t).$$

C. Label Inference Module

Existing semi-supervised spammer detection methods usually focus on design sophisticated projection to map social users to annotation space. Here with the desired features learned from CLN model, an simple inference model like MLP will also achieve desirable performance.

In the training step, only the labeled training samples will be fed into the label inference module. Given a labeled sample x with its label y, we concatenate the final representations from the corrupted encoders in three views along with the demographical features D_x as the input of a single-layer MLP:

$$y' = softmax(W_{mlp} \cdot concat(\tilde{y}_r, \tilde{y}_t, \tilde{y}_s, D_x) + b_{mlp}), \tag{2}$$

where W_{mlp} is the weight matrix, b_{mlp} is the bias parameter and y' contains the predicted probabilities of the user belonging to a spammer or a legitimate user. We select cross-entropy as the classification loss:

$$\begin{aligned}\mathcal{L}_{MLP} &= \sum_{x \in X_l} crossen(y, y'),\\ crossen(y, y') &= -\sum_i y_i log(y_i').\end{aligned} \tag{3}$$

Since the inputs of MLP are generated by CLN model in the feature learning module, in the optimization process the classification loss will be propagated back to CLN to update the model parameters.

In the predicting step, given x in the test set, and the learned parameters W_{mlp} and b_{mlp}, the MLP model takes the clean representations from clean encoders as the input:

$$y' = softmax(W_{mlp} \cdot concat(y_r, y_t, y_s, D_x) + b_{mlp}). \tag{4}$$

D. Optimization Objective

The final optimization objective of the SSDMV model is to minimize the sum of the reconstruction loss of the CLN model and the classification loss of the MLP model:

$$minimize \quad \mathcal{L} = \mathcal{L}_{MLP} + \mathcal{L}_{CLN}. \tag{5}$$

The reconstruction loss ensures the SSDMV model can incorporate high-nonlinear information and correlations across views to better embed multi-view user data into high quality latent features. The classification loss can not only ensure SSDMV make accurate predictions, but also is propagated back to the CLN model to make the learned features focusing more on the annotation relevant details. We utilize Adma optimization algorithm [36] to minimize $\mathcal{L}$. In the predicting process, we utilize Formula (4) to predict the labels of the test samples. Note that only clean representations are utilized in the predicting process.

TABLE I: Statistics of the datasets

	Twitter	Sina Weibo
Number of Spammers	22,223	1,041
Number of Legitimate Users	19,276	3,139
Number of Tweets	5,583,166	221,543
Number of Relations	1,416,800	835,246

V. Experiments

A. Experiment Setup

We use the following two datasets for evaluation, and the detailed statistics are summarized in Table 1.

Twitter Dataset: It is a publicly available spammer detection dataset[1] collected from Twitter. This dataset contains user

[1]http://web.cs.wpi.edu/~kmlee/data.html

TABLE II: Comparison with semi-supervised spammer detection results on Twitter dataset.

Evaluation	F1-Score						Accuracy					
T_c/T_r	100/0.3%	200/0.6%	500/1.5%	1000/3.0%	2000/6.0%	5000/15%	100/0.3%	200/0.6%	500/1.5%	1000/3.0%	2000/6.0%	5000/15%
LP	0.745	0.759	0.764	0.783	0.802	0.814	0.739	0.753	0.762	0.780	0.791	0.803
SSSD	0.841	0.846	0.856	0.864	0.869	0.874	0.839	0.843	0.848	0.859	0.865	0.876
S3VM	0.836	0.839	0.841	0.848	0.853	0.865	0.834	0.841	0.845	0.856	0.860	0.872
SSDMV$_{ae}$	0.863	0.864	0.880	0.895	0.907	0.913	0.860	0.879	0.884	0.894	0.917	0.921
SSDMV$_{ng}$	0.887	0.892	0.909	0.912	0.919	0.928	0.883	0.895	0.903	0.916	0.921	0.924
SSDMV	**0.906**	**0.915**	**0.917**	**0.929**	**0.934**	**0.944**	**0.909**	**0.912**	**0.916**	**0.931**	**0.932**	**0.945**

TABLE III: Comparison with semi-supervised spammer detection results on Sina Weibo dataset.

Evaluation	F1-Score						Accuracy					
T_c/T_r	100/3%	200/6%	300/9%	500/15%	700/21%	1000/30%	100/3%	200/6%	300/9%	500/15%	700/21%	1000/30%
LP	0.704	0.717	0.728	0.733	0.744	0.750	0.719	0.723	0.745	0.747	0.759	0.768
SSSD	0.814	0.826	0.832	0.837	0.846	0.851	0.803	0.817	0.819	0.826	0.837	0.842
S3VM	0.808	0.812	0.823	0.827	0.832	0.836	0.793	0.806	0.813	0.824	0.828	0.830
SSDMV$_{ae}$	0.846	0.853	0.859	0.863	0.871	0.881	0.835	0.840	0.845	0.851	0.859	0.864
SSDMV$_{ng}$	0.861	0.869	0.877	0.880	0.886	0.894	0.855	0.859	0.864	0.869	0.878	0.882
SSDMV	**0.872**	**0.877**	**0.882**	**0.893**	**0.897**	**0.905**	**0.866**	**0.874**	**0.881**	**0.889**	**0.893**	**0.899**

profiles, social relations and the posted tweets. It is a balanced dataset with the similar amount of social spammers and legitimate users. Compared to Twitter dataset used in previous works (10,080 for [3], 13,809 for [14] and 12,453 for [1]), this open dataset is comparatively large.

Sina Weibo Dataset: Sina Weibo is a leading social network in China. From the open Weibo dataset[2], we manually labeled 1,041 spammers and 3,139 legitimate users. It is an imbalanced dataset with the more legitimate users.

Baselines. We compare SSDMV with the following baselines, including state-of-the-art supervised, semi-supervised spammer detection methods and variants of SSDMV.

- **Label Propagation (LP)** [37] is a semi-supervised classification method for graph data. It extends labeling to all the nodes through topology connections. Here we apply it on the social relation network for spammer detection.
- **SSSD** [13] is a semi-supervised social spammer detection method based on the self-learning strategy, which combines the supervised classification model with a ranking scheme.
- **S3VM** [38] is an extension of support vector machines to semi-supervised learning with both labeled and unlabeled samples. We concatenate the learned embedding vectors from different views as the input features of S3VM.
- **MLSI** [7] is a supervised social spammer detection model which considers both topology and text information. Multi-label informed semantic indexing is used to model the content, and undirected graph Laplacian is used to incorporate the topology.
- **BSSD** [22] is also a supervised method which uses directed Laplacian constrained matrix factorization.
- **OSSD** [22] is the online version of BSSD.
- **MVSD** [3] is a supervised method which incorporates the multi-view information by matrix factorization.
- **SSDMV**$_{ae}$ is a variant of SSDMV. It utilizes normal Auto-Encoders (AE) [35] to extract multi-view features instead of ladder networks.
- **SSDMV**$_{ng}$ is also a variant of SSDMV by removing the filter gate, which is designed to verify the effectiveness of the proposed data fusion component.

Parameter Setup. The parameters of SSSD, MLSI, BSSD and OSSD are set by following the original papers. The S3VM algorithm is implemented using CPLE framework[3] with the RBF kernel. For the proposed SSDMV model, the dimension of the input vector in each view generated by the preprocessing step is 512. Each correlated ladder network has 5 layers and the number of neural cells in each layer from bottom to up is set as [512, 1000, 500, 250, 10]. The weight w_l of each layer l from bottom to up is set as [1000.0, 10.0, 0.10, 0.10, 0.10]. Such a weight setting can guarantee that the top layers focus more on the task-relevant feature learning while the bottom layers focus more on the reconstruction task. The strength of noise introduced into each layer is set to $\sigma = 0.3$.

Evaluation Metric. Following the previous literature [22], we use *F1-Score* and *Accuracy* as the evaluation metrics.

B. Quantitative Evaluation with Complete Multi-view Data

We first conduct the evaluation with the complete multi-view data. We randomly select 80% users as the training set and the rest 20% are the testing set. We use 5-fold cross-validation on the training set to select the optimal parameter for each of the algorithms. In the training set, T_c users which account for T_r% of the training data are randomly selected as the labeled samples, and the remaining training samples are unlabeled. We report the average result over 5 runs.

[2]https://aminer.org/data-sna\#Weibo-Net-Tweet

[3]https://github.com/tmadl/semisup-learn

TABLE IV: Comparison with supervised methods on Twitter.

Evaluation	F1-Score			Accuracy		
T_r	10%	50%	100%	10%	50%	100%
MLSI	0.838	0.859	0.879	0.842	0.853	0.877
BSSD	0.878	0.909	0.921	0.882	0.906	0.935
OSSD	0.870	0.907	0.918	0.875	0.911	0.920
MVSD	0.889	0.911	0.934	0.892	0.921	0.933
SSDMV	**0.935**	**0.951**	**0.958**	**0.938**	**0.942**	**0.957**

Comparisons with semi-supervised methods. Tables II and III show the performance of various semi-supervised detection methods on the two datasets. From the results one can see SSDMV consistently outperforms all the baseline methods with the increasing number of the labeled training data. LP performs the worst because it only considers the social relations and ignores the useful information in other views. By incorporating the multi-view data, SSSD and S3VM perform better than LP, but they are still inferior to the deep structure models because the shallow models cannot effectively capture the highly non-linear social information. With the denoising function and skip connections, SSDMV outperforms SSDMV$_{ae}$ by 4%, which proves the ladder network can generate higher quality features than normal auto-encoders. The performance of SSDMV$_{ng}$ without filter gate drops by 2.0% on average compared to SSDMV, which shows that the correlations across different views contribute to generating more discriminative features.

Comparisons with supervised methods. The performance comparison between SSDMV and the supervised methods on the Twitter dataset is shown in Table IV. The ration of labeled training samples T_r increases from 10% to 100%. SSDMV consistently outperforms state-of-the-art supervised shallow methods with the same number of labeled training samples. By incorporating the multi-view data, MVSD achieves the best performance among all the baselines. However, it cannot capture the cross-view correlations, which leads to the poorer performance compared to SSDMV. Meanwhile, from Tables II and IV, one can see SSDMV with only 200 (0.6%) labeled samples achieves a comparable performance (F1-Score: 0.915) to the MVSD (F1-Score: 0.911) with 16000 (50%) labeled samples, which proves the SSDMV can significantly reduce the labor cost for labeling.

Parameter Sensitive Analysis. We study the sensitivity of SSDMV on the parameters of layer number L and noisy strength σ. We set $T_c = 200$ and test the performance of SSDMV under different parameter settings on the Twitter dataset. We let L vary from 2 to 7 and σ vary from 0.1 to 0.5. Figure 4 shows the results. The left figure shows that the performance of SSDMV first increases and then keeps stable with the increase of the model depth. It demonstrates a deeper model helps to better capture the multi-view social media information. Considering a deeper model will cost more computational resources and the time complexity is high, we select a 5-layer model to trade off between the effectiveness and efficiency. When we increase the noisy strength σ, the detection performance first increases and then decreases. It shows that properly introducing noise can help generate more robust features to improve the performance, while excessive noise will hurt the model performance.

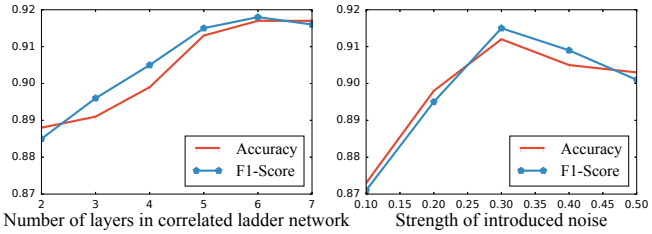


Fig. 4: The effects of parameters (number of layers and strength of noise) on the model performance.

TABLE V: Early spammer detection results.

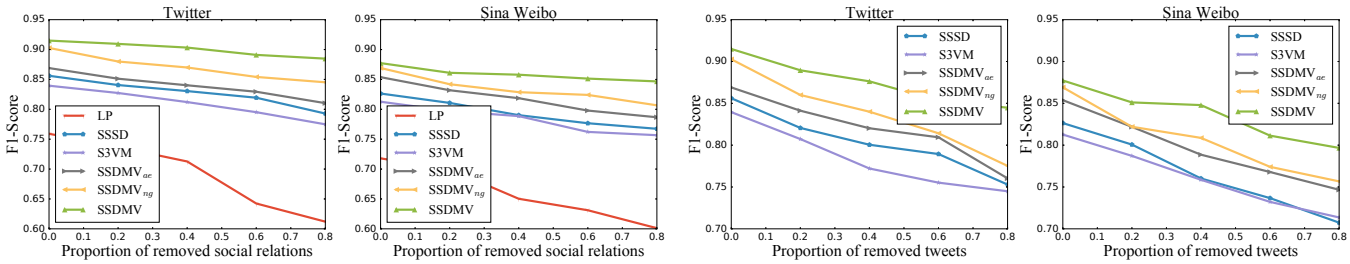
Dataset	Twitter		Sina Weibo	
Evaluation	F1-Score	Accuracy	F1-Score	Accuracy
LP	0.695	0.683	0.644	0.656
SSSD	0.808	0.804	0.782	0.780
S3VM	0.804	0.798	0.777	0.783
SSDMV$_{ae}$	0.829	0.827	0.800	0.790
SSDMV$_{ng}$	0.843	0.847	0.818	0.819
SSDMV	**0.869**	**0.855**	**0.827**	**0.822**

C. Quantitative Evaluation with Sparse and Noisy Data

The data of social users in some views can be sparse or noisy. In addition, it would be better to identify the spammers as early as possible. In this subsection, we will evaluate the robustness of SSDMV in the scenarios of sparse data, early detection and noisy data.

Evaluation with sparse social relations. We randomly remove R_r percentage of social relations from the original social relation network and then perform spammer detection. T_c is set to 200 for both Twitter and Sina Weibo datasets. We increase R_r from 20% to 80% to evaluate the performance of various methods under different proportions of the removed relations. Note that, the counts of followers and followees in the demographical view also change with the remove of social relations. Figure 5(a) shows the results. One can see that with the increase of the removed social relations, the performance of all methods drops. SSDMV consistently outperforms other methods because it presents the least performance decline in F1-Score on both datasets. By capturing the correlations across different views, SSDMV utilizes information from other views to compensate the social relation view, which ensures its more stable performance. LP presents the largest performance drop because it only utilizes the social relations, and thus its performance is very sensitive to the data loss in this view. Without the filter gate to fuse the information from different views, the performance of SSDMV$_{ae}$ and SSDMV$_{ng}$ are also remarkably affected by the reduced social relations.

Evaluation with sparse tweets. Similarly as above, we randomly remove R_t (from 20% to 80%) percentage of tweets from the original corpus, and report the F1-Scores of different



(a) Sensitivity of various methods on the sparsity of social relations.

(b) Sensitivity of various methods on the sparsity of tweets.

Fig. 5: Spammer detection performance w.r.t the proportion of removed social relations and tweets.

TABLE VI: Spammer detection results on noisy tweets.

Dataset	Twitter		Sina Weibo	
Evaluation	F1-Score	Accuracy	F1-Score	Accuracy
S3VM	0.783	0.792	0.760	0.751
SSDMV$_{ae}$	0.825	0.819	0.803	0.805
SSDMV$_{ng}$	0.865	0.864	0.833	0.835
SSDMV	**0.894**	**0.892**	**0.859**	**0.851**

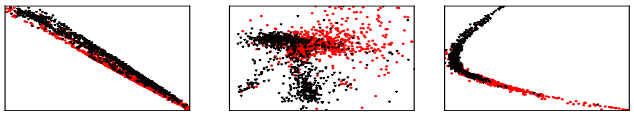
methods on the remained multi-view data. Note that the published symbol view *S* is changed because the urls, hashtags and mentions in the removed tweets are also removed. The count of published tweets in view *D* is also changed. The performance of different methods is shown in Figure 5(b). We ignore the LP method because it does not utilize tweets. One can see that the removed tweets have a much larger impact on the detection performance than the social relations, which demonstrates the text information contributes greater to detect spammers than the social following relationships. Although SSDMV presents a performance drop trend, it still consistently outperforms other baselines. With the increase of R_t, the F1-Score gap between SSDMV and SSDMV$_{ng}$ is enlarged from 1.5% to 5%, which verifies the fusion of multi-view data contributes to alleviating the data sparsity issue.

Early spammer detection. Here we aim to identify whether a new user is a spammer at the very early stage when there are not many malicious behaviors happening. We randomly select 10% of the samples as new users, and then remove 50% of their social relations and tweets to simulate the information scarcity of the new users. We train and tune different methods on the rest 90% users with complete multi-view data, and then apply the learned models to identify spammers in the new user set. Table V shows the early spammer detection performance of different methods. Due to the information scarcity in all views, the performance of all the methods drops significantly, while SSDMV still outperforms all the baselines. By deeply fusing information from different views, SSDMV can comprehensively utilize a few available annotations to capture task-relevant details for spammer detection.

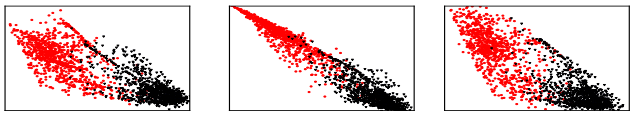
Evaluation with noisy tweets. The posted tweets are usually unstructured texts with linguistic noise, which may seriously affect the detection performance. To evaluate the robustness of SSDMV to the noisy tweets, we randomly select 20% users as test samples and add noise into their tweets. Specifically, we randomly draw noise from a normal Gaussian distribution $\mathcal{N}(0, \sigma_n^2)$ and add the Gaussian noise to the embedding vectors in the tweet text view of the test samples. We train and tune different detection models on the rest 80% users with clean data. Then the learned models are applied to predict the noisy test samples. Table VI shows the results on noisy test set. Compared with the results on the clean test set shown in Tables II and III with T_c=200, one can see the performance of all methods drops because the introduced Gaussian noise changes the data distribution of the test set. SSDMV outperforms other methods due to the two reasons: 1) the ladder network integrates denoising function which contributes to learning more robust features; and 2) the filter gate fuses complementary information from other views with clean data.

D. Visualization

To intuitively present the quality of the learned latent representations, we further visualize the latent user representations learned by SSDMV. We randomly select 2000 users from the Twitter dataset, and then train SSDMV model on the rest samples with $T_c = 1000$. The number of neural cells in the bottle-neck layer of the CLN model is set to 2. With the learned model, we can get the 2-dimensional clean representations y_t, y_r and y_s of three views for the selected 2000 users. We utilize scatter diagram to represent each user as a node. Spammers are the red circles and legitimate users are the black ones. For comparison, we also utilize Node2Vec to transform the selected 2000 users to 2-dimensional vectors in the views R and S, and use Doc2Vec to learn the 2-dimensional feature vectors in view T. Figure 6 shows the visualization of the users in the two different latent spaces. The upper three figures show the distributions of users in the unsupervised feature space. One can see that spammers cannot be easily separated from legitimate users in this space. The representations learned by unsupervised methods are not discriminative enough, leading to the less promising detection performance. The lower three figures show the distributions of the users in the latent spaces learned by SSDMV. One can see that the spammers and legitimate users are much more separable than they are in the upper figures, which verifies the superiority of SSDMV in discriminative feature learning.



(a) Representations of view R with Node2Vec (b) Representations of view T with Doc2Vec (c) Representations of view S with Node2Vec



(d) Representations of view R with SSDMV (e) Representations of view T with SSDMV (f) Representations of view S with SSDMV

Fig. 6: Visualization of the user distributions in the learned 2-dimensional latent representation spaces.

VI. Conclusion

In this paper, we proposed a semi-supervised deep social spammer detection model SSDMV by multi-view data fusion. Specifically, in the multi-view feature learning module, we proposed a deep correlated-ladder-networks model to embed high-nonlinear information and correlations across different views into high quality features. The label inference module predicted labels for users and guided the feature learning module to focus on the task-relevant details. By optimizing the reconstruction loss and the classification loss simultaneously under a unified semi-supervised learning framework, SSDMV significantly outperformed state-of-the-art methods on both effectiveness and robustness through extensive evaluations over two real datasets.

Acknowledgments

This work was supported by the Natural Science Foundation of China (Grand Nos. U1636211,61672081,61370126,61602237,61672313,61503253), Beijing Advanced Innovation Center for Imaging Technology (No.BAICIT-2016001), National Key R&D Program of China (No.2016QY04W0802),NSF of Guangdong Province (No. 2017A030313339) and NSF of Jiangsu Province (No. BK20171420) . This work was also supported in part by NSF through grants IIS-1526499, IIS-1763325, and CNS-1626432.

References

[1] X. Hu, J. Tang, Y. Zhang, and H. Liu, "Social spammer detection in microblogging." in *IJCAI*, vol. 13, 2013, pp. 2633–2639.

[2] F. Wu, J. Shu, Y. Huang, and Z. Yuan, "Co-detecting social spammers and spam messages in microblogging via exploiting social contexts," *Neurocomputing*, vol. 201, pp. 51–65, 2016.

[3] H. Shen, F. Ma, X. Zhang, L. Zong, and X. Liu, "Discovering social spammers from multiple views," *Neurocomputing*, 2017.

[4] K. Lee, B. D. Eoff, and J. Caverlee, "Seven months with the devils: A long-term study of content polluters on twitter," in *ICWSM*, 2011.

[5] A. H. Wang, "Don't follow me: Spam detection in twitter," in *Security and Cryptography (SECRYPT)*. IEEE, 2010, pp. 1–10.

[6] X. Hu, J. Tang, H. Gao, and H. Liu, "Social spammer detection with sentiment information," in *ICDM*. IEEE, 2014, pp. 180–189.

[7] Y. Zhu, X. Wang, E. Zhong, N. N. Liu, H. Li, and Q. Yang, "Discovering spammers in social networks," in *AAAI*, 2012.

[8] S. Fakhraei, J. Foulds, and L. Getoor, "Collective spammer detection in evolving multi-relational social networks," in *KDD*. ACM, 2015, pp. 1769–1778.

[9] E. Tan, L. Guo, S. Chen, X. Zhang, and Y. Zhao, "Unik: Unsupervised social network spam detection," in *CIKM*. ACM, 2013, pp. 479–488.

[10] Z. Miller, B. Dickinson, W. Deitrick, W. Hu, and A. H. Wang, "Twitter spammer detection using data stream clustering," *Information Sciences*, vol. 260, pp. 64–73, 2014.

[11] W. Shao, L. He, C.-T. Lu, X. Wei, and P. S. Yu, "Online unsupervised multi-view feature selection," *ICDM*, 2016.

[12] D. Yu, N. Chen, F. Jiang, B. Fu, and A. Qin, "Constrained nmf-based semi-supervised learning for social media spammer detection," *Knowledge-Based Systems*, vol. 125, pp. 64–73, 2017.

[13] Z. Li, X. Zhang, W. Liang, and Z. He, "A semi-supervised framework for social spammer detection," in *PAKDD*, 2015, pp. 177–188.

[14] X. Zhang, H. Bai, and W. Liang, "A social spam detection framework via semi-supervised learning," in *PAKDD*. Springer, 2016, pp. 214–226.

[15] X. Zhu, "Semi-supervised learning tutorial," in *ICML*, 2007, pp. 1–135.

[16] D. Luo, F. Nie, H. Huang, and C. H. Ding, "Cauchy graph embedding," in *ICML*, 2011, pp. 553–560.

[17] L. Derczynski, A. Ritter, and S. Clark, "Twitter part-of-speech tagging for all: Overcoming sparse and noisy data," in *RANLP*, 2013, pp. 198–206.

[18] S. Wang, X. Hu, P. S. Yu, and Z. Li, "Mmrate: inferring multi-aspect diffusion networks with multi-pattern cascades," in *KDD*. ACM, 2014, pp. 1246–1255.

[19] D. Yang, S. Wang, C. Li, and X. Zhang, "From properties to links: Deep network embedding on incomplete graphs," in *CIKM*, 2017.

[20] G. Andrew, R. Arora, J. Bilmes, and K. Livescu, "Deep canonical correlation analysis," in *ICML*, 2013, pp. 1247–1255.

[21] M. Pezeshki, L. Fan, P. Brakel, A. Courville, and Y. Bengio, "Deconstructing the ladder network architecture," in *ICML*, 2016.

[22] X. Hu, J. Tang, and H. Liu, "Online social spammer detection," in *AAAI*, 2014.

[23] A. T. Kabakus and R. Kara, "A survey of spam detection methods on twitter," *IJACSA*, vol. 8, no. 3, pp. 29–38, 2017.

[24] T. Wu, S. Liu, J. Zhang, and Y. Xiang, "Twitter spam detection based on deep learning," in *ACSWM*, 2017, p. 3.

[25] C. Li, Y. Wu, W. Wu, C. Xing, and M. Zhou, "Detecting context dependent messages in a conversational environment," *COLING*, 2016.

[26] C. Li, Z. Li, S. Wang, Y. Yang, X. Zhang, and J. Zhou, "Semi-supervised network embedding," in *DASFAA*. Springer, 2017, pp. 131–147.

[27] C. Li, S. Wang, D. Yang, Z. Li, Y. Yang, X. Zhang, and J. Zhou, "Ppne: property preserving network embedding," in *DASFAA*. Springer, 2017, pp. 163–179.

[28] F. Huang, X. Zhang, Z. Li, and T. Mei, "Learning social image embedding with deep multimodal attention networks," *arXiv*, 2017.

[29] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *KDD*. ACM, 2016, pp. 855–864.

[30] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *ICML*, 2014, pp. 1188–1196.

[31] H. Valpola, "From neural pca to deep unsupervised learning," *Advances in Independent Component Analysis and Learning Machines*, pp. 143–171, 2015.

[32] A. Rasmus, M. Berglund, M. Honkala, H. Valpola, and T. Raiko, "Semi-supervised learning with ladder networks," in *NIPS*, 2015, pp. 3546–3554.

[33] P. Vincent, H. Larochelle, and Y. Bengio, "Extracting and composing robust features with denoising autoencoders," in *ICML*, 2008, pp. 1096–1103.

[34] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015, pp. 448–456.

[35] G. Alain and Y. Bengio, "What regularized auto-encoders learn from the data-generating distribution," *JMLR*, vol. 15, no. 1, pp. 3563–3593, 2014.

[36] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[37] X. Zhu and Z. Ghahramani, "Learning from labeled and unlabeled data with label propagation," 2002.

[38] F. Gieseke, A. Airola, T. Pahikkala, and O. Kramer, "Fast and simple gradient-based optimization for semi-supervised support vector machines," *Neurocomputing*, vol. 123, pp. 23–32, 2014.