Wu X, Fan W, Gao J *et al.* Detecting marionette microblog users for improved information credibility. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 30(5): 1082–1096 Sept. 2015. DOI 10.1007/s11390-015-1584-4

Detecting Marionette Microblog Users for Improved Information Credibility

Xian Wu [1] (吴 贤), Wei Fan [2] (范 伟), *Member, ACM*, Jing Gao [3] (高 晶), *Member, ACM, IEEE*
Zi-Ming Feng [1] (冯子明), and Yong Yu [1] (俞 勇)

[1] *Department of Computer Science, Shanghai Jiao Tong University, Shanghai 200240, China*
[2] *Baidu Research Big Data Laboratory, Sunnyvale, CA 94089, U.S.A.*
[3] *Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY 14214, U.S.A.*

E-mail: wuxian@apex.sjtu.edu.cn; wei.fan@gmail.com; jing@buffalo.edu; {zimingfeng, yyu}@apex.sjtu.edu.cn

Received November 15, 2014; revised June 15, 2015.

Abstract In this paper, we propose to detect a special group of microblog users: the "marionette" users, who are created or employed by backstage "puppeteers", either through programs or manually. Unlike normal users that access microblog for information sharing or social communication, the marionette users perform specific tasks to earn financial profits. For example, they follow certain users to increase their "statistical popularity", or retweet some tweets to amplify their "statistical impact". The fabricated follower or retweet counts not only mislead normal users to wrong information, but also seriously impair microblog-based applications, such as hot tweets selection and expert finding. In this paper, we study the important problem of detecting marionette users on microblog platforms. This problem is challenging because puppeteers are employing complicated strategies to generate marionette users that present similar behaviors as normal users. To tackle this challenge, we propose to take into account two types of discriminative information: 1) individual user tweeting behavior and 2) the social interactions among users. By integrating both information into a semi-supervised probabilistic model, we can effectively distinguish marionette users from normal ones. By applying the proposed model to one of the most popular microblog platforms (Sina Weibo) in China, we find that the model can detect marionette users with F-measure close to 0.9. In addition, we apply the proposed model to calculate the marionette ratio of the top 200 most followed microbloggers and the top 50 most retweeted posts in Sina Weibo. To accelerate the detecting speed and reduce feature generation cost, we further propose a light-weight model which utilizes fewer features to identify marionettes from retweeters.

Keywords marionette microblog user, information credibility, fake follower, fake retweet

1 Introduction

Microblog acts as both a social community and an information media. On one hand, it allows users to connect with each other by following, replying or retweeting; on the other hand, it attracts enormous users to produce, consume and propagate information. The dual functionality of microblog attracts users in hundreds of millions. According to recent statistics, the number of Twitter users has exceeded 645 million in July 2014①. In China, Sina Weibo and Tecent Weibo have attracted more than 275 million users in June 2014②. Such a large number of participants have made microblog a new social phenomenon that attracts attention from a variety of domains, such as business intelligence, social science, and life science.

In microblog services, the messages (tweets) usually deliver time-sensitive information, e.g., "What is the user currently doing". By following people he/she is interested in, a user will be notified of all their posted tweets and thus keeps track of what these people are doing or thinking about. Therefore, the number of fol-

Regular Paper
Special Section on Social Media Processing
A preliminary version of the paper was published in the Proceedings of ECMLPKDD 2013.
①http://www.statisticbrain.com/twitter-statistics/, Nov. 2014.
②http://www.cnnic.net.cn/, Nov. 2014.
©2015 Springer Science + Business Media, LLC & Science Press, China

lowers measures someone's popularity, and can indicate how much influence someone has. For celebrities, a large number of followers show their social impact and can increase their power in advertisement contract negotiations. As for normal users, a relatively large number of followers represent their rich social connections and promote their positions in social networks. Therefore, both celebrities and normal users are eager to get more followers.

Due to the retweet mechanism, information propagates quite efficiently in microblog services. Once a user posts a message, his/her followers will be notified immediately. If these followers further retweet this message, their followers can view it immediately as well. In this way, the size of the audience of this message can grow at exponential speed. Compared with common messages, popular messages could attract more users to retweet them. Therefore, the retweet count of a message can represent its popularity. In many microblog platforms (e.g., Sina Weibo), the retweet count is adopted as the key metric to select top stories③. As a result, some microblog users are willing to purchase some retweets to promote their messages for commercial purposes.

The desire for more followers and retweets triggers the emergence of a new microblog business: follower and retweet purchase. The backstage puppeteers maintain a large pool of marionette users. To purchase followers or retweets, the buyer first provides his/her user ID or tweet ID. Then the puppeteer activates certain number of marionette users to follow this buyer or retweet his/her messages. The number of followers or retweets depends on the price paid. The fee is typically modest, 25 USD for 5 000 followers in Twitter, and 15 Yuan (i.e., about 2.5 USD) for 10 000 followers in Sina Weibo. Moreover, the massive following process is quite efficient. For example, it only takes one night to add 10 000 followers in Sina Weibo, which can make someone become "famous" overnight.

From the perspective of people who purchase followers or retweets, the marionette users can satisfy their needs to become famous or to promote commercial advertisements. But the overall fabrications conducted by marionette users can lead to serious damages.

• The purchased follower and retweet counts are not an objective reflection of the social influence of the users or public attention paid to the messages. As a result, the fake number can mislead real users and data mining applications based on microblog data, such as [1].

• Beside promoting advertisements, the marionette users are sometimes employed to spread rumors[2]. It will not only mislead normal users but also provide wrong evidence for the establishment of business[3] and the government's policies and strategies. Thus, this becomes a serious financial and political problem.

• To disguise as normal users, the marionette users are operated by puppeteers to perform some random actions, including following, retweeting, and replying. Such actions can annoy normal users and result in unpleasant experiences.

Therefore, identifying marionette users is a key problem for ensuring the normal functioning of microblog services, but detecting marionette users is nontrivial. Back to Nov. 2011, we purchased 2 000 followers from Taobao (China EBay) and all these fake followers were recognized by the microblog platform and deleted within two days. Such a quick detection can be attributed to several discriminative features. For example, the marionette accounts are usually created from the same IP address within a short period of time, and many marionette users post no original tweets but only perform massive following or retweeting. Therefore, the microblog platform can employ simple rules to detect marionette users and delete their accounts. However, the marionette users are evolving and becoming more and more intelligent. Nowadays, the puppeteers hire people to create marionette accounts manually. To make these accounts behave like normal users, the puppeteers develop highly sophisticated strategies by operating the marionette users to follow celebrities, reply to hot tweets, and conduct other complicated operations. These disguises can easily overcome the filtering strategy of microblog platforms and make marionette users much more difficult to be detected. In February 2012, we purchased another 4 000 followers. At this time, 1 790 marionette users survived after five weeks, and around 1 000 marionette users were still active by Feb. 2013.

After analyzing the behavior of marionette users and comparing them with normal users, we find that the following two types of information are useful in detecting marionette users.

• *Local features*: the set of features that describe individual user behavior, which could be either textual or numerical. Normal and marionette users present different behaviors, which can be captured by these local features. For example, the counts of followings and followers are important features that distinguish a large

③http://hot.weibo.com, Oct. 2014.

portion of normal users from marionette users. The time intervals between tweets and the posting devices also serve as effective clues to detect marionette users.

• *Social relations*: the following, retweeting or other relationships among users. Such relations provide important information for marionette user detection. For example, the marionette users will follow both normal users and other marionette users. They follow normal users for disguise or profits, and follow other marionette users to help them to disguise. On the other hand, normal users are less likely to follow marionette users. In determining whether a user is a marionette or not, we can examine his/her followers. If the majority are normal users, this user is more likely to be a normal user. Otherwise, this user could be a marionette. Therefore, besides local features, the social relation is useful in detecting marionettes.

These two types of features provide complementary predictive powers for the task of marionette user detection. Therefore, we propose a probabilistic model that seamlessly takes both the rich local features and the social relations among users into consideration to detect marionette users more effectively. On the dataset collected from Sina Weibo, the proposed model is able to detect marionette users at the F-measure close to 0.9.

By applying the proposed model to label retweeters, we are able to measure the true popularity of the corresponding tweet. In case of hot tweets that possess millions of retweeters, we need to access a large volume of data to extract the local and social features. Such high IO workload could slow down the whole detection process or even exhaust all available IO bandwidth. To deal with this problem, we propose a light-weight model which works on fewer features. Instead of pulling full features for all users, we sample a subset of users and predict whether they are marionettes or not. Then we analyze the shared features within retweeting content and propagate labels from the sampled users to the rest. In this manner, for the majority of users, we only need to extract features from their retweets on a post which reduces the heavy IO workload. The experiments demonstrate that this light-weight model can detect marionettes from retweeters at a high accuracy.

2 Proposed Model

In this section, we describe the proposed probabilistic model that combines local user features and social relations in marionette user detection.

2.1 Notation Description

Let u_i denote a microblog user and let the vector $\boldsymbol{x}_i$ denote the features of u_i. Each dimension of $\boldsymbol{x}_i$ represents a local feature, which could be the follower count of u_i or the used tweeting device. Let binary variable y_i denote the label of u_i, 1 stands for the marionette user and 0 stands for the normal user. Let $V^{(i)} = \{v_1^{(i)}, v_2^{(i)}, \ldots, v_{M(i)}^{(i)}\}$ denote the $M(i)$ users who are related to u_i. Let $\boldsymbol{x}_j^{(i)}$ denote the local features of the j-th user in set $V^{(i)}$. In microblog, the social relations between users can be either explicit or implicit. To be concrete, "followed" and "following" are explicit relations while retweeting one's tweet or "mention" someone in a tweet could establish implicit social relations. In this paper, we target to predict the label y_i of u_i given his/her local features $\boldsymbol{x}_i$ and social relations $V^{(i)}$.

2.2 Problem Formulation

We will first introduce how to use local features that describe users' behaviors, such as follower/following counts, the posting devices, to build a discriminative model. Then we will describe how to incorporate social relations into this model to further improve the performance. If we only consider the local features, marionette user detection is a typical classification problem. A variety of classification models can be used, among which we choose logistic regression because it can be adapted to incorporate social relations. Let us first describe how to model local user features using logistic regression model.

We introduce the sigmoid function in (1) to represent the probability of belonging to marionette or normal class given feature values, i.e., $P(y_i|\boldsymbol{x}_i)$, for each user.

$$P_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_i) = h_{\boldsymbol{\theta}}(\boldsymbol{x}_i)^{y_i}(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}_i))^{(1-y_i)}, \tag{1}$$

where $h_{\boldsymbol{\theta}}(\boldsymbol{x}_i) = \frac{1}{1+e^{-\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}_i}}$ and $h_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$ is equal to the probability that u_i is a marionette user. $\boldsymbol{\theta}$ is the vector of parameters that govern the sigmoid function. With (1), we can formulate the joint probability over N labeled users in (2), in which we try to find the parameter vector $\boldsymbol{\theta}$ to maximize this data likelihood.

$$\max_{\boldsymbol{\theta}} \quad \prod_{i=1}^{N} P_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_i). \tag{2}$$

(2) is the objective function for training and the parameter $\boldsymbol{\theta}$ can be obtained by maximizing this objective function. We can apply various numerical

optimization methods like BFGS (Broyden-Fletcher-Goldfarb-Shanno algorithm) to solve this optimization problem.

In the above formulation, each user is treated separately and the prediction of a marionette user only depends on his/her own local features. However, besides the local features, the relations between users are also discriminative in the task of predicting marionette users. To incorporate the social relations, we modify the objective function from (2) to (3).

$$\max_{\boldsymbol{\theta},\boldsymbol{\alpha}} \quad \prod_{i=1}^{N}\{P_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_i)\prod_{j=1}^{M(i)}P_{\boldsymbol{\alpha}}(y_i|y_j^{(i)})^d\}. \tag{3}$$

In (3), we assume that, for each user u_i, the labels of his/her $M(i)$ neighbors $y_0^{(i)}$, $y_1^{(i)}$, ..., $y_{M(i)}^{(i)}$ are known in advance. Then we can combine the effect of local features and user connections together to predict marionette users. d is the co-efficient that balances between the social relations and local features. The larger d is, the more biased the model is towards the social relations in making predictions. Note that to simplify the presentation, we consider the case where only one type of social relations exists in (3). However, the proposed model is general enough and can be easily adapted to cover multi-type social relations. Take the microblog system for example, the common user relations include follower, following, mention, retweet and reply. We can introduce different parameters to correspond to each kind of relation and model all relations in one unified framework.

In (3), $P_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_i)$ is formulated using the same sigmoid function shown in (1). $P_{\boldsymbol{\alpha}}(y_i|y_j^{(i)})$ will be modeled using Bernoulli distribution and governed by parameter $\boldsymbol{\alpha}$ as shown in (4).

$$P_{\boldsymbol{\alpha}}(y_i|y_j^{(i)}=k)=\alpha_k^{y_i}(1-\alpha_k)^{(1-y_i)}, \quad k=0,1. \tag{4}$$

As k is either 1 or 0, we can write down all the possible $P_{\boldsymbol{\alpha}}(y_i|y_j^{(i)}=k)$ in (5).

$$\begin{pmatrix} P(y_i=0|y_j^{(i)}=0)=\alpha_0 & P(y_i=1|y_j^{(i)}=0)=1-\alpha_0 \\ P(y_i=0|y_j^{(i)}=1)=\alpha_1 & P(y_i=1|y_j^{(i)}=1)=1-\alpha_1 \end{pmatrix}. \tag{5}$$

For each user, the parameter $\boldsymbol{\alpha}$ measures the influence received from his/her neighbors. α_0 indicates the chance of a user being a normal user if his/her neighbor is a normal user. If the neighbor is normal, the larger α_0 is, the more likely this user is to be a normal user. Similarly, α_1 indicates the chance of a user being a normal user if his/her neighbor is a marionette user. If the neighbor is marionette, the larger α_1 is, the more likely this user is to be a normal user. The logarithm of the joint probability in (3) can be represented in (6):

$$\ell(\boldsymbol{\theta},\boldsymbol{\alpha})=l_1+l_2, \tag{6}$$

where

$$l_1=\sum_{i=1}^{N}y_i\log h_{\boldsymbol{\theta}}(\boldsymbol{x}_i)+(1-y_i)\sum_{i=1}^{N}\log(1-h_{\boldsymbol{\theta}}(\boldsymbol{x}_i)),$$

$$l_2=d\sum_{i=1}^{N}\sum_{j=1}^{M(i)}\sum_{k=y_j^{(i)}}(y_i\log\alpha_k+(1-y_i)\log(1-\alpha_k)).$$

The parameters $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$ will be inferred by maximizing the log-likelihood in (6). To solve this optimization problem, it is natural to apply gradient descent approaches. Notice that $\boldsymbol{\theta}$ is only included in l_1 and $\boldsymbol{\alpha}$ is only included in l_2, we can maximize l_1 and l_2 separately to infer the value of $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$. $\boldsymbol{\theta}$ can be acquired via numerical optimization methods using the same procedure in the aforementioned logistic regression formulation. As for $\boldsymbol{\alpha}$, we can derive the following analytical solution by maximizing the objective function in l_2.

$$\alpha_k=\frac{\sum_{i=1}^{N}\sum_{j=1}^{M(i)}\sum_{k=y_j^{(i)}}y_i}{\sum_{i=1}^{N}\sum_{j=1}^{M(i)}\sum_{k=y_j^{(i)}}1}.$$

Clearly, α_k can be regarded as the chance of observing class label k in the neighbors of the i-th user.

The above model takes social relations into consideration, but it has several disadvantages that may prevent its usage in real practice. First, the model can only work in a supervised scenario where the class labels of all the neighbors of each user are observed. This is a strong assumption and can only be achieved by spending huge amounts of time and labeling costs to get sufficient training data. Second, even if we acquire sufficient labeled data, the discriminative information hidden in the labeled data is not fully utilized in the model. As shown in (3), the labels on a user's neighbors are only used in modeling $P(y_i|y_j^{(i)})$ without considering the relationship between the labels of these neighbors and their local features. Intuitively, if two neighbors have the same class label but different local features, their effect on the target user's label should be different.

Therefore, we propose to adapt (3) to (7) by considering both class labels and local features of a user's neighbors:

$$\max_{\boldsymbol{\theta},\boldsymbol{\alpha}} \quad \prod_{i=1}^{N}\{P_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_i)\prod_{j=1}^{M(i)}P_{\boldsymbol{\alpha},\boldsymbol{\theta}}(y_i|\boldsymbol{x}_j^{(i)})^d\}. \tag{7}$$

The only difference between (3) and (7) is that we replace $P_{\boldsymbol{\alpha}}(y_i|y_j^{(i)})$ with $P_{\boldsymbol{\alpha},\boldsymbol{\theta}}(y_i|\boldsymbol{x}_j^{(i)})$. In this way, the proposed model incorporates the local features of the neighbors and the model does not have the strong assumption that the neighbors' labels are fully observed.

In (7), we represent $P_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_i)$ using the same sigmoid function shown in (1). As for $P_{\boldsymbol{\alpha},\boldsymbol{\theta}}(y_i|\boldsymbol{x}_j^{(i)})$, its formulation can be inferred based on (8).

$$\begin{aligned} & P_{\boldsymbol{\alpha},\boldsymbol{\theta}}(y_i|\boldsymbol{x}_j^{(i)}) \\ = & \sum_{k=0}^{1} P_{\boldsymbol{\alpha},\boldsymbol{\theta}}(y_i, y_j^{(i)}=k|\boldsymbol{x}_j^{(i)}) \\ = & \sum_{k=0}^{1} P_{\boldsymbol{\alpha},\boldsymbol{\theta}}(y_i|y_j^{(i)}=k, \boldsymbol{x}_j^{(i)})P_{\boldsymbol{\theta}}(y_j^{(i)}=k|\boldsymbol{x}_j^{(i)}). \end{aligned} \tag{8}$$

We assume that the label of a user y_i is conditionally independent of the local features $\boldsymbol{x}_j^{(i)}$ of his/her neighbor given the label of this neighbor $y_j^{(i)}$, and thus we have $P_{\boldsymbol{\alpha},\boldsymbol{\theta}}(y_i|y_j^{(i)}=k, \boldsymbol{x}_j^{(i)}) = P_{\boldsymbol{\alpha}}(y_i|y_j^{(i)}=k)$. Hence, we modify (8) accordingly into (9).

$$P_{\boldsymbol{\alpha},\boldsymbol{\theta}}(y_i|\boldsymbol{x}_j^{(i)}) = \sum_{k=0}^{1} P_{\boldsymbol{\alpha}}(y_i|y_j^{(i)}=k)P_{\boldsymbol{\theta}}(y_j^{(i)}=k|\boldsymbol{x}_j^{(i)}). \tag{9}$$

By plugging the above definition of $P_{\boldsymbol{\alpha},\boldsymbol{\theta}}(y_i|\boldsymbol{x}_j^{(i)})$ into the proposed objective function in (7), we combine the local features and social relations into one model to distinguish marionette from normal users. Accordingly, the log-likelihood in (6) is modified to (10):

$$\begin{aligned} & \ell(\boldsymbol{\theta},\boldsymbol{\alpha}) \\ = & \sum_{i=1}^{N} y_i \log h_{\boldsymbol{\theta}}(\boldsymbol{x}_i) + (1-y_i)\sum_{i=1}^{N}\log(1-h_{\boldsymbol{\theta}}(\boldsymbol{x}_i)) + \\ & d\sum_{i=1}^{N}\sum_{j=1}^{M(i)}\log\sum_{k=0}^{1}P_{\boldsymbol{\alpha}}(y_i|y_j^{(i)}=k)P_{\boldsymbol{\theta}}(y_j^{(i)}=k|\boldsymbol{x}_j^{(i)}). \end{aligned} \tag{10}$$

2.3 Parameter Estimation

In the proposed model, two sets of parameters need to be estimated: $\boldsymbol{\theta}$ in both $P_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_j)$ and $P_{\boldsymbol{\alpha},\boldsymbol{\theta}}(y_i|\boldsymbol{x}_j^{(i)})$, and $\boldsymbol{\alpha}$ in $P_{\boldsymbol{\alpha},\boldsymbol{\theta}}(y_i|\boldsymbol{x}_j^{(i)})$. These parameters should be obtained by maximizing the logarithm of (10). As the class labels of one's neighbors are unknown, we treat them as latent hidden variables during the inference procedure. The following hidden variable $z_{jk}^{(i)}$ is introduced in (11).

$$\begin{aligned} z_{jk}^{(i)} & \propto P_{\boldsymbol{\alpha},\boldsymbol{\theta}}(y_i, y_j^{(i)}=k|\boldsymbol{x}_j^{(i)}) \\ & \propto P_{\boldsymbol{\alpha}}(y_i|y_j^{(i)}=k)P_{\boldsymbol{\theta}}(y_j^{(i)}=k|\boldsymbol{x}_j^{(i)}). \end{aligned} \tag{11}$$

Based on this hidden variable, the objective function can be represented in (12):

$$\begin{aligned} & \ell'(z_{jk}^{(i)},\boldsymbol{\theta},\boldsymbol{\alpha}) \\ = & \sum_{i=1}^{N}\log P_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_i) + \\ & \sum_{i=1}^{N}\sum_{j=1}^{M(i)}\sum_{k=0}^{1} z_{jk}^{(i)}\log P_{\boldsymbol{\alpha}}(y_i|y_j^{(i)}=k) + \\ & \sum_{i=1}^{N}\sum_{j=1}^{M(i)}\sum_{k=0}^{1} z_{jk}^{(i)}\log P_{\boldsymbol{\theta}}(y_j^{(i)}=k|\boldsymbol{x}_j^{(i)}). \end{aligned} \tag{12}$$

We propose to use EM method to iteratively update model parameters and hidden variables. At the E-Step, the hidden variable $z_{jk}^{(i)}$ can be calculated via (13):

$$z_{jk}^{(i)} = \frac{P_{\boldsymbol{\alpha}}(y_i|y_j^{(i)}=k)P_{\boldsymbol{\theta}}(y_j^{(i)}=k|\boldsymbol{x}_j^{(i)})}{\sum_{k=0}^{1}P_{\boldsymbol{\alpha}}(y_i|y_j^{(i)}=k)P_{\boldsymbol{\theta}}(y_j^{(i)}=k|\boldsymbol{x}_j^{(i)})}. \tag{13}$$

At the M-Step, we maximize the parameter $\ell'(z_{jk}^{(i)},\boldsymbol{\theta},\boldsymbol{\alpha})$ with respect to $\boldsymbol{\alpha}$ and get the following solution of $\boldsymbol{\alpha}$ in (14).

$$\alpha_k = \frac{\sum_{i=1}^{N}\sum_{j=1}^{M(i)} z_{jk}^{(i)} y_i}{\sum_{i=1}^{N}\sum_{j=1}^{M(i)} z_{jk}^{(i)}}. \tag{14}$$

The estimation of $\boldsymbol{\theta}$ can be transformed into the parameter estimation process of logistic regression by constructing a training set. Initially, the training dataset only includes N labeled users $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$. Then for each neighbor of the users, two instances $(\boldsymbol{x}_j^{(i)}, y_j^{(i)}=0)$ and $(\boldsymbol{x}_j^{(i)}, y_j^{(i)}=1)$ are generated and added into the training dataset. In total, there are $2\sum_{i=1}^{N}M(i)$ new instances added. The weights of the newly added instances are different from those of the initial ones. For the initial training instance $(\boldsymbol{x}_i, y_i)$, its weight is 1, while the weight of the newly added instance $(\boldsymbol{x}_j^{(i)}, y_j^{(i)}=k)$ is $d\times z_{jk}^{(i)}$. The detailed parameter estimation process is summarized in Algorithm 1.

After obtaining the values of $\boldsymbol{\alpha}$ and $\boldsymbol{\theta}$ using Algorithm 1 from data, we can now use the proposed model

to predict the class label of a new user u_i. This user's label y_i can be predicted according to (15).

$$\arg\max_{y_i} \quad P_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_i) \prod_{j=1}^{M(i)} P_{\boldsymbol{\alpha},\boldsymbol{\theta}}(y_i|\boldsymbol{x}_j^{(i)})^d, \tag{15}$$

where $P_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_i)$ can be calculated using (1) and $P_{\boldsymbol{\alpha},\boldsymbol{\theta}}(y_i|\boldsymbol{x}_j^{(i)})$ can be calculated using (9).

Algorithm 1. Parameter Estimation

Data: training dataset $D = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$ and their unlabeled neighbors
Result: values of $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$
1 **while** EM not converged **do**
2 **E-step:**
3 Update $z_{jk}^{(i)}$ according to (13)
4 **M-step:**
5 Update $\boldsymbol{\alpha}$ according to (14)
6 For each neighbor of D, add two instances $\{(\boldsymbol{x}_j^{(i)}, y_j^{(i)} = k), k = 0, 1\}$ and assign weight $d \times z_{jk}^{(i)}$
7 Apply parameter estimation process of logistic regression to calculate $\boldsymbol{\theta}$

2.4 Time Complexity

Another perspective we want to discuss is the time complexity and the number of iterations needed to converge. As shown in Algorithm 1, the parameter estimation process basically consists of EM iterations. During each iteration, the values of the hidden variables $z_{jk}^{(i)}$, $\boldsymbol{\theta}$, and $\boldsymbol{\alpha}$ are updated. According to (13) and (14), the time complexity for calculating $z_{jk}^{(i)}$ and $\boldsymbol{\alpha}$ is $O(NM)$ where N is the number of instances and M is the average of the number of neighbors. As for $\boldsymbol{\theta}$, the calculation is the same as the parameter estimation of weighted logistic regression, whose time complexity depends on the optimization method adopted. In total, the time complexity for training is $O(TNM + T \times (LR))$ where T denotes the number of iterations and LR represents the time complexity of logistic regression optimization.

We illustrate the convergence speed of Algorithm 1 in Fig.1. We conduct this experiment on a microblog dataset which will be introduced later in Subsection 4.1. We calculate the log-likelihood after each round of iteration and plot the values of log-likelihood with respect to each iteration. It can be observed that Algorithm 1 converges quickly. After eight rounds, the log-likelihood becomes stable. Therefore, a small number of iterations can achieve good performance. On the training dataset consisting of 12 000 users with 30 iterations, the proposed approach only takes less than 10 seconds to converge on a commodity PC.

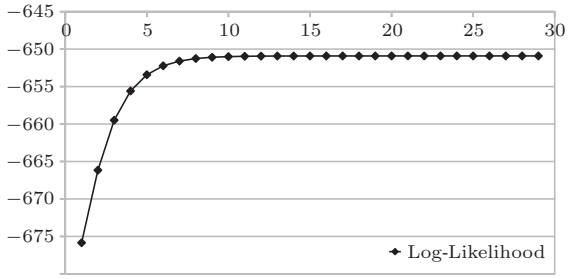


Fig.1. Log-likelihood value with EM iterations.

3 Detecting Marionettes in Retweets

Utilizing the model proposed in Section 2, we can detect marionette users and separate them from normal ones. To estimate the true retweet count of a hot post, a straight-forward manner is to collect features of each retweeter and classify whether the retweeters are marionettes or not. By excluding retweets from those detected marionettes, we can estimate the true retweeting count for this hot post. In this manner, the applications based on microblog data[1,3-4] can get rid of marionettes and work on more reliable signals.

The main challenge of this detect-and-exclude approach is the high IO workload. In practice, a hot tweet could be retweeted by millions of users④. To apply the proposed model, we need to collect the profile, posted tweets, and the neighborhood data from millions of users. Accessing such a large volume of data is very time-consuming and could exhaust all IO bandwidth.

To relieve from high IO workload, we present a light-weight detection method which relies on less input. Instead of pulling the full data of all users, we sample a small fraction of users and classify whether they are marionettes or not. Based on these predictions, we train another classifier with fewer features. For example, we only collect the features which can be acquired directly from the retweets, like the word bag and device information. In this manner, we no longer need to collect the full data of all users and thus can reduce the total IO workload. The example in Fig.2 illustrates how this light-weight approach works.

Fig.2 lists four retweets posted by four different users over the same tweet. The original tweet promotes for a commercial activity. If a user participates in this

④http://hot.weibo.com, Oct. 2014.

activity, he/she will have the chance to visit Hong Kong for free. Among these retweets, we find that although posted by different users, the first one and the third one share almost the same content. Since such coincidence is unlikely to happen, we may guess these two retweets are posted by the same backstage user with different accounts. Therefore if the first user is classified as a marionette, the third one could be a marionette as well. Based on this intuition, we extract the full features of the first user and predict his/her label with the model presented in Section 2. Then we propagate this prediction to the third user as well as others sharing similar features in retweets. In this manner, we only pull full features from one user and can predict the labels of a group of users.

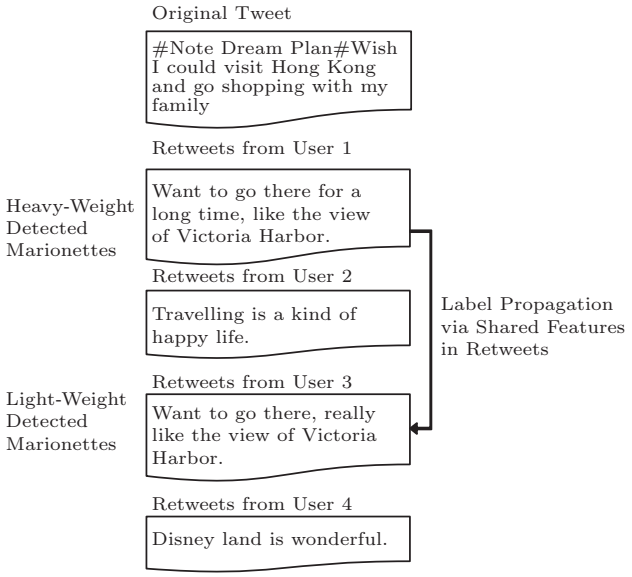


Fig.2. Example of the light-weight marionette detection.

We present the detailed process of light-weight marionette detection in Algorithm 2. Here we denote all N retweets over a post T with the set $R = \{(u_1, t_1, d_1), \ldots, (u_N, t_N, d_N)\}$ where each element R_i represents the user u_i retweets T and adds his/her comment t_i with the device d_i. Algorithm 2 consists of two classification processes: 1) Algorithm 2 employs the model in Section 2 to predict the labels of sampled users in R_s; 2) based on the prediction results on R_s, Algorithm 2 trains another classifier with the light-weight features and applies this classifier to predict the rest users in R. Since Algorithm 2 works on fewer features than the full model, the heavy burden of IO workload is relieved. The experiments in Subsection 4.5 prove that even with less features, the light-weight classifier performs well in detecting marionettes. Please note that this light-weight model replies on the context features of retweets to make predictions. In case of other scenarios without additional information to leverage, we still need to apply the full model presented in Section 2 to detect marionettes.

Algorithm 2. Light-Weight Detection

Data: the retweet set of a hot post $R = \{(u_1, t_1, d_1), \ldots, (u_N, t_N, d_N)\}$
Result: label of each item in R
1 Derive a sample set R_s from R
2 Collect full data for sampled users in R_s
3 Predict the labels of sampled users
4 Collect light-weight features for all users in R
5 Train a predictor on sampled users with light-weight features
6 Predict the labels of the rest users in R

4 Experiments

This section is organized as follows: Subsection 4.1 describes the datasets used in experiments; Subsection 4.2 analyzes the features that are discriminative in marionette detection; Subsection 4.3 calculates the classification accuracy of the proposed model and shows that incorporating social relations can indeed improve the performance; Subsection 4.4 lists several applications and calculates the ratio of marionettes in the top 200 most followed microbloggers and the top 50 most retweeted posts; Subsection 4.5 evaluates the performance of the light weight model with different settings of the sample set.

4.1 Datasets

4.1.1 Classification Corpus

We acquire a dataset that consists of labeled marionette and normal users to evaluate the proposed model.

• *Marionette Users.* To collect the corpus of marionette users, we first created three phishing Sina Weibo accounts and bought followers from three Taobao shops for three times. Each time we purchased 2 000 followers and altogether there are 6 000 in total. The first purchase was made in November 2011 and the other two were made in February 2012. In Feb. 2013, we re-examined these bought marionette users and found that around 1 000 are still active while the rest have already been deleted or blocked by Sina Weibo. Over

1/6 marionette users are not discovered by Sina Weibo for over a year. To target a more challenging problem and compensate the existing detecting methods of Sina Weibo, we select these well hidden marionette users into our corpus.

• *Normal Users*. As for the normal users, we first select ten seed users manually and crawl the users they are following. After that, the crawled users are taken as new seeds to continue the crawling. Through this iterative procedure, we collect 70 288 unique users. Among them, 29 334 have been verified by Sina Weibo (according to the fields "verified" and "verifiedType" in Sina Weibo API). That is 41.7% of all users. As Sina requires the user to fax an ID copy for verification, we can confirm these users are normal users. From these verified users, we randomly select 1 000 into our corpus that is the same amount as the marionette users. In real life, the distribution of normal and marionette users is usually imbalanced. However, to make the classifier more accurate, we decide to undersample the normal users and use a balanced training set to train the classifier which is commonly used in imbalanced classification[5].

For each obtained user, we further randomly select five users from all their followers into the dataset. As a result, this dataset consists of 2 000 labeled users and 10 000 unlabeled users. The profiles and posted tweets of these 12 000 users are also crawled.

4.1.2 Suspicious Hot Tweet Corpus

In Sina Weibo, the account of social network analysis⑤ has listed several hot tweets that were suspiciously promoted by marionette users. This account visualizes the retweet prorogation of these suspicious tweets and finds the topological differences compared with the normal hot tweets. For each mentioned tweet, we retrieve 200 users who have retweeted this tweet.

4.1.3 Top Microblogger and Hot Tweets

To evaluate the marionette rates in microblog, we collect the top 200 most followed microbloggers⑥ and the top 50 most retweeted posts⑦ in Sina Weibo. We collect the data of popular microbloggers on June 16th, 2014. For each microblogger, we crawl 2 000 of his/her followers. We collect the hot tweets from the hourly rank board at 7 pm on Oct. 13th, 2014. For each tweet, we crawl 2 000 users who retweeted this tweet. In total, we crawl as many as 500 000 users.

4.2 Feature Selection

In microblog platforms, for each user, we are able to acquire many kinds of local features. In this subsection, we analyze some features to see whether they are discriminative in marionette user classification.

4.2.1 Number of Tweets/Followings/Followers

For each user, we extract the number of posted tweets, followings, and followers. Then we demonstrate the comparison results in three sub-figures of Fig.3 respectively. The x-axis represents different numbers of tweets, followings, and followers, while the y-axis represents the number of users with the same number of tweets, followings, and followers. Both axes use logarithmic scale.

As shown in Fig.3(a), we find the marionettes are relatively inactive in tweeting, and a large proportion of marionettes post less than 20 tweets. On the contrary, the normal users are more active. The most "energetic" normal user posts more than 30 000 tweets. Therefore, a large number of tweets can be an effective feature to recognize normal users. In Fig.3(b), we find the number of followings of most marionettes is between 100 and 1 000. One possible explanation to this range is that the puppeteer restricts the maximal following time to avoid being detected by microblog services. In Fig.3(c), we find most marionettes receive less than 200 followers, and the majority of followers are also marionettes who are helping them to disguise as normal users. The rest are probably inexperienced users who follow back when getting followed by marionettes.

4.2.2 Tweet Posting Device

As a convenience to users, microblog platforms provide multiple access manners. Besides the typical web interface, users can post tweets via mobile clients, the third party microblog applications, etc. Thus, we try to figure out whether there are any differences in posting devices between normal users and marionettes. All the tweets are posted from 1 912 different sources, in which 1 707 different sources are used by normal users and 869 different sources are used by marionette users. Table 1 lists the top 5 mostly used sources for normal users and marionettes respectively. We find that more than half tweets of normal users are posted via "Sina Web"; thus the web interface remains the primary choice for

⑤http://weibo.com/dmonsns, Nov. 2014.
⑥http://top.weibo.com/, June 2014.
⑦http://hot.weibo.com/, Oct. 2014.

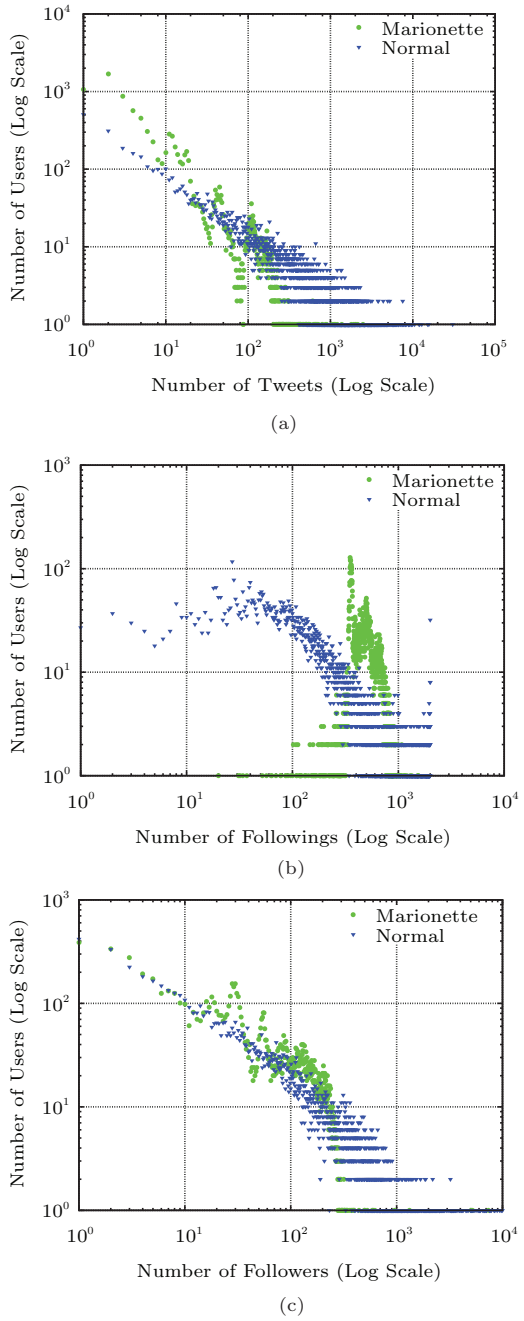


Fig.3. User number distribution on the number of (a) tweets, (b) followings, and (c) followers.

accessing microblog, and "iPhone" and "Andorid" are the two most popular mobile clients. While for marionette users, most tweets are posted via "Sina Mobile" which denotes that majority tweets are posted via cell phones web browser. In this case, if massive user accounts are created from some mobile IP address, the microblog service could not block this IP as it could be the real requests from normal users in the same district. Besides, the IP address can be changed when the puppeteer relocates.

Table 1. Top 5 Most Used Devices to Post Tweets

Normal		Marionette	
Device	Number of Tweets	Device	Number of Tweets
Sina Web	356 192	Sina Mobile	209 739
iPhone	59 996	Sina Web	29 365
Android	54 778	UC Browser	4 775
Sina Mobile	19 733	Android	2 577
S60	19 278	iPhone	2 112

Besides above local features, we also select: the maximal, minimal, middle and average length of tweets; the maximal, minimal, middle and average time interval between tweets; the percentage of retweets. We do not include the word bag features here. This is because we want to make the model more generic. Since the marionette users owned by the same backstage puppeteer will retweet the same tweet, if the bag-of-word features are utilized as features, the trained model will incline to these word features and become over-fit. To our knowledge, the bot detection of Bing[6] only uses behavior features and the words are used in blacklist for pre-filtering.

4.3 Classification Evaluation

To show the advantages of incorporating social relations, we compare the proposed model with the baseline method which only applies logistic regression on the local features without considering social relations. When evaluating the proposed model, we set different values of d and different numbers of neighbors to illustrate the impact of social relations on the marionette user detection task. We implement the proposed method based on Weka[7] and the recorded accuracy is the average computed based on 5-fold cross validation.

• *Baseline*. The baseline model is a logistic regression classification model which adopts the local features introduced in Subsection 4.2.

• *Light-Neighbor*. This model is the proposed model which adopts the same local features as the baseline model and incorporates the social relations with the setting of five neighbors and $d = 0.1$.

• *Heavy-Neighbor*. It is similar to the light-neighbor model, except that this model biases more towards social relations with a higher degree setting $d = 0.5$.

Table 2 lists the weighted classification precision, recall, and F-measure over the three models. We can find that incorporating social relation increases the performance of detecting marionette users.

Table 2. Classification Results on the Three Models

	Precision	Recall	F-Measure
Baseline	0.884	0.875	0.872
Light-neighbor	0.900	0.890	0.887
Heavy-neighbor	**0.907**	**0.895**	**0.892**

We also evaluate the proposed model with different settings of d and $\#neighbor$ (the number of neighbors) and show the results in Fig.4. From this figure, we can find that incorporating social relations in modeling can improve the classification accuracy. For example, when $\#neighbor$ equals 1, 2 or 3, the accuracy improves w.r.t. the increase of d; on the other hand, when d equals 0.1, the accuracy improves w.r.t. the increase of $\#neighbor$. However, setting the value of $\#neighbor$ or d too high could decrease the accuracy. Therefore, we need to balance between local features and social relations and choose proper values for d and $\#neighbor$. In practice, the parameter adjustment methods like grid search can be applied.

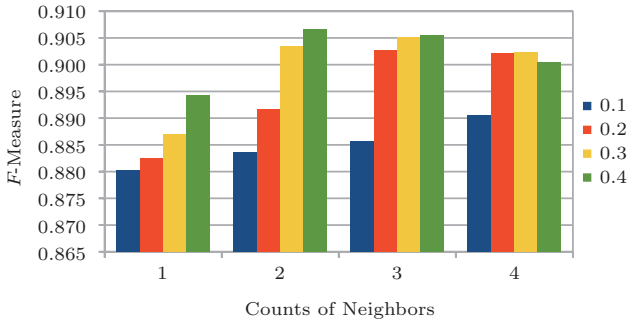


Fig.4. Classification F-measure with different neighbor and degree settings.

4.4 Applications of Marionette Detection

We apply the proposed probabilistic model to detect the credibility of hot retweets and apply the model learnt from the classification corpus to identify marionettes from suspicious hot tweets corpus, and finally we can obtain the percentage of users that are classified as marionette users among the users who retweet this message.

Table 3 lists the Weibo accounts that post suspicious hot tweets, their promotion purpose and percentage of marionette users. It can be seen that the percentage of the first four tweets is quite high, which suggests that most of their retweets are conducted by marionette users. Although the retweet of the last tweet shown in Table 3 involves more normal users, it might be attributed to the fact that marionette users attract the attention of many normal users and thus the goal of promotion is achieved through marionette user purchase.

To provide a quantitative analysis on the popular microblogger and hot tweets, we apply the proposed model to estimate the ratios of marionette in the followers of the top 200 most followed microblggers and in the retweeters of the top 50 most retweeted posts. Fig.5 displays the results. The bar represents the follower count or the retweet count of the corresponding microblgger or post; the line represents the ratio of marionettes among all followers or retweeters. In Fig.5, we place the microbloggers or tweets in descending order according to their follower counts and retweeting counts. Please note that we acquire the hot tweets from a hourly ranking list, and thus the number in Fig.5(b) only records the retweet count within one hour.

Table 4 summarizes the statistics of the ratios of detected marionette. Among all popular microbloggers, the one who ranks the 18th in followers (42 million) has the highest marionette rate, up to 55.6%. In other words, this microblogger is followed by 23 million marionettes. This microblogger is a founder of a regional clothing company. According to the popularity, this microblogger should not be listed in the top 200. It seems that he/she purchased marionettes to be ranked the top 200. In this manner, the company can receive more attention from the outside.

Table 3. Marionette User Percentage of Suspicious Hot Tweets

Microblogger	Promotion Purpose	Marionette Rate (%)
A web site of clothing industry	Web site promotion	100.00
A famous brand of women's dress	Weibo account promotion	98.61
Provincial Culture Communication CO., LTD	Ceremony advertisement	93.62
A anti-worm software for mobile device	A security issue reminder	92.44
A famous China smart phone manufacturer	Advertisement of sale promotion	43.04

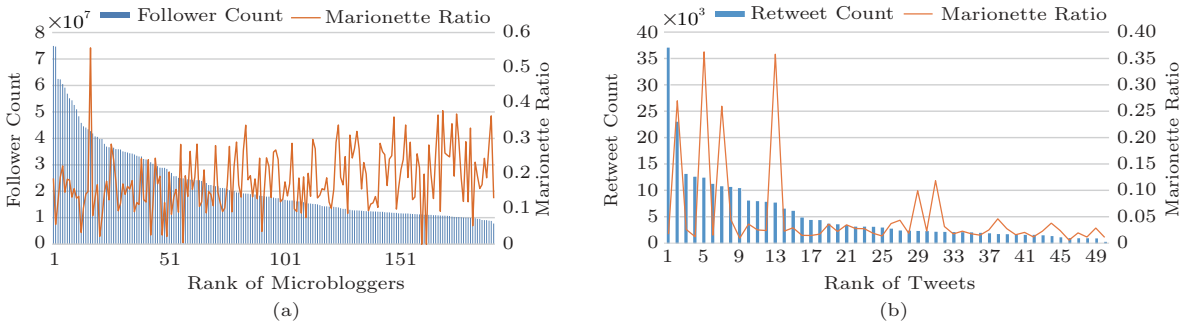


Fig. 5. (a) Marionette ratio of followers of top 200 most followed microblggers. (b) Marionette ratio of retweeters of top 50 most retweeted posts (hourly rank board).

Table 4. Basic Statistics of Marionette Ratio

	Min (%)	Median (%)	Mean (%)	Max (%)
Follower	0.5	16.7	18.1	55.6
Retweeter	0.6	1.7	5.0	36.2

4.5 Light-Weight Model

We select four hot tweets with the highest marionette ratios in Fig.5(b). For each hot tweet, we collect 2 000 retweeters (the maximum limit of Sina Weibo API) from which we sample a seed set of users. Then for each selected user, we collect the full features and apply the proposed model to predict their labels. As for the rest users, we only collect the following features which are available from the retweets. 1) Word bag: we apply FudanNLP[8] to segment the sentences into words. We also parse the hashtag, mentions and URLs from text and treat them as word bag features. 2) Posting devices: we extract the name of the device used to retweet. 3) Statistical features: the number of hashtags, mentions, and URLs. Based on these three kinds of features, we train a classifier over sampled users and apply it to predict the labels of the rest. Here we set the size of sampled users to be from 1% to 4% of the entire set and display the results in Table 5.

To acquire the golden labels on the entire dataset, we apply the proposed model trained over full features to predict all retweeters for these four tweets. From Table 5, we find that with only 1% users, we can achieve an F-measure of 0.85 in average. In this manner, we no longer need to prepare the full features for 99% users which significantly reduce the IO workload. Please note that this light weight leverages the local features in the retweets to achieve such high accuracy. In scenarios without additional information to utilize, we need the full model presented in Section 2 to detect marionettes.

5 Related Work

In this section, we describe related work from three perspectives: 1) the applications that are conducted on microblog data; 2) the credibility issues of web data and corresponding solutions; 3) the credibility issues of microblog data.

5.1 Microblog Data Applications

Microblog data is the characteristic of time-sensitivity and geological information associated. To leverage such particular features, many emergent approaches have been proposed: Twitter Monitor[9] was proposed to detect trends over tweet streams and identify the emergence of events; Sakaki *et al.*[1] utilized the geographical information of tweets to locate natural disasters, such as earthquakes, or track the path of typhoon. Interestingly, the notification of earthquake acquired by the method provided in [1] is even quicker than that given by Japan Earthquake Bureau. Yin *et al.*[10] modeled both the content and the geographical information of tweets in the same statistical

Table 5. Prediction Accuracy of Light-Weight Classifier with Different Sizes of Sample Set

Seed Ratio (%)	Tweet No.1			Tweet No.2			Tweet No.3			Tweet No.4		
	Precision	Recall	$F1$	Precision	Recall	$F1$	Precision	Recall	$F1$	Precision	Recall	$F1$
1	0.733	1.000	0.846	0.817	0.993	0.897	0.646	0.955	0.773	0.823	1.000	0.903
2	0.797	0.920	0.854	0.865	0.993	0.925	0.701	0.886	0.783	0.867	0.992	0.926
3	0.797	0.953	0.868	0.880	0.993	0.934	0.704	0.927	0.801	0.906	0.992	**0.947**
4	0.800	0.972	**0.878**	0.890	0.993	**0.939**	0.744	0.931	**0.827**	0.928	0.964	0.946

framework, and discovered the topic variety at different places; Duan *et al.*[11] focused on the problem of topic summarization in Twitter, which aims to provide a short and compact summary for a collection of tweets on the same or similar topics. Different from typical document summarization, Duan *et al.*[11] leveraged the authority of tweet publishers as well as the user connections to improve the summarization quality; Lehmann *et al.*[12] analyzed the temporal pattern of hot hashtag on Twitters. Lehmann *et al.*[12] introduced two dimensions: the days before the peak and the days after the peak to classify the hashtag into four different categories.

There are many approaches that utilize microblog data to improve traditional tasks: Dong *et al.*[13] found that some tweets include URL as a part of the content and such information can help with improving web page ranking especially for the newly created web pages; Yang *et al.*[4] leveraged the textual content surrounding the URL in tweets and proposed a dual wing model to improve the accuracy of web page content summarization.

Since a variety of services are built upon microblog data, the credibility of microblog data is becoming more and more important.

5.2 Credibility of Web Data

Prior to the emergence of microblog service, the web has existed for over two decades. Many web services have been dealing with all kinds malicious behaviors and cheating for a long time. For example, since the search engine utilizes user log to develop search result ranking, related search suggestion and query autosuggestion, many robots are operated to submit specific queries or conduct fake clicks to hack the log of search engines[14]. Many approaches[6,15] have been proposed to detect and exclude these robot users. The main difference between the robot users and the marionette microblog users is that we can obtain the social relations from marionette users, which can be utilized in building classification models. Besides robot users and automated traffic, another web data issue is the link spam which tries to increase the PageRank of certain pages by creating a large number of links pointing to them. [16-19] propose to optimize search engine ranking and minimize the effects of link spam. The link spam detection is different from marionette microblog user detection. The former is a ranking problem, while the latter is a classification problem. The former targets to lower the rank of link spam web pages and many proposed methods like [16] rely on the link structure of the web, but the latter task targets to separate the marionette users from normal users with local features and their social connections.

5.3 Credibility of Microblog Data

Due to the enormous consumers of microblog data from both applications and individual perspectives, the credibility of microblog data becomes extremely important. Castillo *et al.*[20] explored the information credibility of news propagated through Twitter and proposed to assess the credibility level of newsworthy topics.

The existence of destructive users, such as spammers and marionettes, reduces the credibility of microblog data. Many approaches were proposed to detect and exclude these destructive users. In Table 6, we categorize existing work according to two dimensions: the vertical dimension denotes the targeted type, spammers or marionettes; the horizontal dimension denotes whether the dependencies of connected users are included in modeling.

Table 6. Two-Dimensional Categorization of Related Work on Detecting Destructive Users

	Modeling User Dependency	
	No	Yes
Spammers	[21-24]	[25-28]
Marionettes	[29-31]	Proposed approach

5.3.1 Detecting Spam or Malicious Users

Gowri and Mohanraj[32] reviewed existing work on detecting Twitter spam users. They summarized previous work according to introduced features, evaluation metrics and employed models. Yardi *et al.*[33] employed three simple rules to detect Twitter spammers including: 1) searches for URLs; 2) username pattern matches; 3) keyword detection. Other approaches like [23-24, 34] analyze the discriminative features on tweet content and user social behaviors and trained supervised models to separate spammers from normal users.

Laboreiro *et al.*[22] focused on one kind of Twitter spammer, the automatic posting accounts. They employed multiple features to detect such accounts. Since the spammers are evolving to evade the detection of microblog platform, Yang *et al.*[21] applied empirical methods to analyze how spammers disguise as normal users and proposed 26 robust and discriminative features to build the detection model. Thomas *et al.*[35]

collected a set of trusted users as seeds and then recursively included more users whom trusted users communicate with. But in case of new users who are not connected to trusted users, Thomas *et al.*[35] failed to detect whether they were spammers or not.

Ghosh *et al.*[26] identified a particular type of harmful microblog users: the link farmer. This type of users try to acquire more followers and distribute spams. The main difference between the link farmers and the marionette users is that the former one is seeking for followers and the latter one is providing followers. Yang *et al.*[25] identified another type of harmful microblog users, the cyber criminals. Different from marionette users, the cyber criminals generate direct harm to normal users by spreading phishing scams.

The SMFSR method proposed by [27] employs a matrix factorization based method to find spammers in social networks. Different from our proposed approach, this method is transductive rather than inductive. In other words, it is difficult to be used to predict over unseen users who are not in the training set. Every time, when new users are added, the entire matrix factorization needs to run again. Due to the computational bottleneck of matrix factorization, SMFSR is hard to scale up to a large number of features. In contrast, the proposed approach in this paper is built upon the sigmoid function which can easily scale to high-dimensional features.

[28] is related to our proposed approach in the sense that it combines user activities, social regularization, and semi-supervised labeling in one framework. The main difference is that [28] aims to detect the spammers who did harm to normal users and thus were deleted by the microblog platform. However, in this paper, we target to detect a different group of users, the marionettes who are paid to promote for a microblogger or a tweet.

For other social network applications, Rahman *et al.*[36] provided a plugin to exclude the Facebook specific malware: socware. They introduced various local features to detect the socware and outperformed the baseline blacklist method.

5.3.2 Detecting Marionettes or Purchased Users

Stringhini *et al.*[37] studied the phenomenon of Twitter account markets and showed the negative impacts caused by purchased followers: 1) faking the number of followers of buyers' accounts; 2) distributing spams which sometimes could be malicious. Jiang *et al.*[38] compared the behaviors between purchased followers and normal users. They found that 1) purchased followers tend to share similar patterns; 2) the patterns of purchased followers are different from those of normal users.

Aggarwal and Kumaraguru[29] provided an in-depth analysis of purchased followers on Twitter and built a prediction model based on local features of each user. Shen *et al.*[30] explored the purchased followers in the most popular microblog platform of China, Sina Weibo. They studied the discriminative features between fake followers and normal users and applied these features in building the detecting model. Liu *et al.*[31] also focused on Sina Weibo and proposed to detect marionette users with existing classification models.

The main differences between our paper and [29-31] are as follows.

• [29-31] formulate marionette detection as a binary classification problem and directly apply existing models like SVM, naive Bayes and logistic regression for training. These approaches are similar to the baseline model (Table 2) in terms of adopted features and employed model, while in this paper, we propose a statistical framework that models both local features and social relations. According to the results in Table 2, the integration of social relations can improve the detection accuracy.

• The proposed model is semi-supervised. Compared with supervised classification models, the proposed model relies less on manual labelling and is more robust to imbalanced training data.

• In detecting marionettes from massive retweeters, we further propose a light-weight classifier which could work on fewer features and save the total IO cost.

6 Conclusions

In the paper, we first discussed the business model of marionette users or how they make profits in microblog services. The following facts motivate the emergence of marionettes: 1) to increase the number of followers and fake their popularity, some users purchase marionette users to follow them; 2) to increase the retweeting count, the advertisers pay marionette users to retweet their tweets. Marionette users deceive microblog services by faking retweeting and following counts, and the random actions of marionette users can annoy normal users. Therefore, to ensure information trustworthiness, it is extremely important to detect marionette users in a timely manner. In this paper, we proposed an effective probabilistic approach to fully utilize local user features and social relations in detect-

ing marionette users. We extracted user behavior features together with their neighbors' information from both normal and marionette users, and integrated such information into a probabilistic model. We proposed an iterative EM procedure to infer model parameters which can then be used to predict whether a user is a marionette or normal. Experiments on Sina Weibo data showed that the proposed method achieves a very high F-measure close to 0.9, and the further analysis on some retweet examples demonstrated the effectiveness of the proposed model in measuring the true credibility of information on microblog platforms. We applied the proposed model to estimate the ratios of marionettes in the top 200 most followed microbloggers and the top 50 most retweeted posts. We found that in the worst case, up to 55.6% followers and 36.2% retweeters are marionettes. To detect massive retweeters with limited IO bandwidth, we further proposed a light-weight model which works on much fewer features to achieve an F-meausre of 0.85 in average.

References

[1] Sakaki T, Okazaki M, Matsuo Y. Earthquake shakes Twitter users: Real-time event detection by social sensors. In *Proc. the 19th International Conference on World Wide Web*, April 2010, pp.851-860.

[2] Yu L L, Asur S, Huberman B A. Artificial inflation: The real story of trends and trend-setters in Sina Weibo. In *Proc. the International Conference on Privacy, Security, Risk and Trust and International Conference on Social Computing*, September 2012, pp.514-519.

[3] Bollen J, Mao H, Zeng X. Twitter mood predicts the stock market. *arXiv.1010.3003*, 2010. http://arxiv.org/abs/1010.3003, June 2015.

[4] Yang Z, Cai K, Tang J, Zhang L, Su Z, Li J. Social context summarization. In *Proc. the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, July 2011, pp.255-264.

[5] Chawla N V, Bowyer K W, Hall L O, Kegelmeyer W P. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 2002, 16(1): 321-357.

[6] Kang H, Wang K, Soukal D, Behr F, Zheng Z. Large-scale bot detection for search engines. In *Proc. the 19th International Conference on World Wide Web*, April 2010, pp.501-510.

[7] Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten I H. The WEKA data mining software: An update. *SIGKDD Explorations*, 2009, 11(1): 10-18.

[8] Qiu X, Zhang Q, Huang X. FudanNLP: A toolkit for Chinese natural language processing. In *Proc. the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, August 2013, pp.49-54.

[9] Mathioudakis M, Koudas N. TwitterMonitor: Trend detection over the Twitter stream. In *Proc. the 2010 ACM SIGMOD International Conference on Management of Data*, June 2010, pp.1155-1158.

[10] Yin Z, Cao L, Han J, Zhai C, Huang T. Geographical topic discovery and comparison. In *Proc. the 20th International Conference on World Wide Web*, March 28-April 1, 2011, pp.247-256.

[11] Duan Y, Chen Z, Wei F, Zhou M, Shum H. Twitter topic summarization by ranking tweets using social influence and content quality. In *Proc. the 24th International Conference on Computational Linguistics*, December 2012, pp.763-780.

[12] Lehmann J, Gonçalves B, Ramasco J J, Cattuto C. Dynamical classes of collective attention in Twitter. In *Proc. the 21st International Conference on World Wide Web*, April 2012, pp.251-260.

[13] Dong A, Zhang R, Kolari P, Bai J, Diaz F, Chang Y, Zheng Z, Zha H. Time is of the essence: Improving recency ranking using Twitter data. In *Proc. the 19th International Conference on World Wide Web*, April 2010, pp.331-340.

[14] Buehrer G, Stokes J W, Chellapilla K. A large-scale study of automated web search traffic. In *Proc. the 4th International Workshop on Adversarial Information Retrieval on the Web*, April 2008, pp.1-8.

[15] Yu F, Xie Y, Ke Q. SBotMiner: Large scale search bot detection. In *Proc. the 3rd ACM International Conference on Web Search and Data Mining*, February 2010, pp.421-430.

[16] Gyöngyi Z, Garcia-Molina H, Pedersen J. Combating web spam with TrustRank. In *Proc. the 30th International Conference on Very Large Data Bases*, August 31-September 3, 2004, pp.576-587.

[17] Wu B, Davison B D. Identifying link farm spam pages. In *Proc. Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, May 2005, pp.820-829.

[18] Krishnan V, Raj R. Web spam detection with anti-trust rank. In *Proc. the 2nd International Workshop on Adversarial Information Retrieval on the Web*, August 2006, pp.37-40.

[19] Benczúr A A, Csalogány K, Sarlós T, Uher M. SpamRank — Fully automatic link spam detection. In *Proc. the 1st International Workshop on Adversarial Information Retrieval on the Web*, May 2005, pp.25-38.

[20] Castillo C, Mendoza M, Poblete B. Information credibility on Twitter. In *Proc. the 20th International Conference on World Wide Web*, Mar. 2011, pp.675-684.

[21] Yang C, Harkreader R C, Gu G. Empirical evaluation and new design for fighting evolving Twitter spammers. *IEEE Transactions on Information Forensics and Security*, 2013, 8(8): 1280-1293.

[22] Laboreiro G, Sarmento L, Oliveira E C. Identifying automatic posting systems in microblogs. In *Proc. the 15th Portuguese Conference on Artificial Intelligence*, October 2011, pp.634-648.

[23] McCord M, Chuah M. Spam detection on Twitter using traditional classifiers. In *Proc. the 8th International Conference on Autonomic and Trusted Computing*, September 2011, pp.175-186.

[24] Benevenuto F, Magno G, Rodrigues T, Almeida V. Detecting spammers on Twitter. In *Proc. the 7th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference*, July 2010.

[25] Yang C, Harkreader R, Zhang J, Shin S, Gu G. Analyzing spammers' social networks for fun and profit: A case study of cyber criminal ecosystem on Twitter. In *Proc. the 21st International Conference on World Wide Web*, April 2012, pp.71-80.

[26] Ghosh S, Viswanath B, Kooti F, Sharma N K, Korlam G, Benevenuto F, Ganguly N, Gummadi K P. Understanding and combating link farming in the Twitter social network. In *Proc. the 21st International Conference on World Wide Web*, April 2012, pp.61-70.

[27] Zhu Y, Wang X, Zhong E, Liu N N, Li H, Yang Q. Discovering spammers in social networks. In *Proc. the 26th AAAI Conference on Artificial Intelligence*, July 2012, pp.171-177.

[28] Hu X, Tang J, Zhang Y, Liu H. Social spammer detection in microblogging. In *Proc. the 23rd International Joint Conference on Artificial Intelligence*, August 2013, pp.2633-2639.

[29] Aggarwal A, Kumaraguru P. Followers or phantoms? An anatomy of purchased Twitter followers. *arXiv:1408.1534*, 2014. http://arxiv.org/abs/1408.1534, June 2015.

[30] Shen Y, Yu J, Dong K, Nan K. Automatic fake followers detection in Chinese micro-blogging system. In *Proc. the 18th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, May 2014, pp.596-607.

[31] Liu H, Zhang Y, Lin H, Wu J, Wu Z, Zhang X. How many zombies around you? In *Proc. the 13th International Conference on Data Mining*, December 2013, pp.1133-1138.

[32] Gowri C D, Mohanraj V. A survey on spam detection in Twitter. *International Journal of Computer Science and Business Informatics*, 2014, 14(1): 92-102.

[33] Yardi S, Romero D M, Schoenebeck G, Boyd D. Detecting spam in a Twitter network. *First Monday*, 2010, 15(1).

[34] Hentschel M, Alonso O, Counts S, Kandylas V. Finding users we trust: Scaling up verified Twitter users using their communication patterns. In *Proc. the 8th International Conference on Weblogs and Social Media*, June 2014.

[35] Thomas K, Grier C, Song D, Paxson V. Suspended accounts in retrospect: An analysis of Twitter spam. In *Proc. the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, November 2011, pp.243-258.

[36] Rahman M S, Huang T K, Madhyastha H V, Faloutsos M. Efficient and scalable socware detection in online social networks. In *Proc. the 21st USENIX Conference on Security Symposium*, August 2012, Article No. 32.

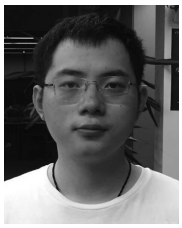
[37] Stringhini G, Egele M, Kruegel C, Vigna G. Poultry markets: On the underground economy of Twitter followers. In *Proc. the 2012 ACM Workshop on Online Social Networks*, August 2012, pp.1-6.

[38] Jiang M, Cui P, Beutel A, Faloutsos C, Yang S. Detecting suspicious following behavior in multimillion-node social networks. In *Proc. the Companion Publication of the 23rd International Conference on World Wide Web Companion*, April 2014, pp.305-306.

Xian Wu is now a Ph.D. candidate in the Department of Computer Science of Shanghai Jiao Tong University. His research interests include data mining, statistical learning and nature language processing. Xian received his Master's degree from Shanghai Jiao Tong University in 2007 and Bachelor's degree from Southeast University, Nanjing, in 2004, both in computer science.

Wei Fan is now the deputy head of Baidu Research Big Data Lab. Before joining Baidu, Wei worked in IBM T. J. Watson Research Lab and Huawei Noah's Ark Lab. Wei received his Ph.D. degree in computer science from Columbia University in 2001, M.E. and B.E. degrees in computer science from Tsinghua University, Beijing, in 1995 and 1993 respectively. He published more than 60 papers in top data mining, machine learning and database conferences, such as KDD, SDM, ICDM, ECML/PKDD, SIGMOD, VLDB, ICDE, AAAI, ICML and so on. His main research interests and experiences are in various areas of data mining and database systems.

Jing Gao is now an assistant professor in the Department of Computer Science and Engineering of the University at Buffalo. She got her Ph.D. degree in computer science from University of Illinois at Urbana Champaign in 2011 under the supervision of Prof. Jiawei Han. She received her M.E. and B.E. degrees from the Department of Computer Science and Technology at Harbin Institute of Technology in China.



Zi-Ming Feng received his M.E. and B.E. degrees in computer science from Shanghai Jiao Tong University in 2014 and 2011 respectively. His research interests include computer vision, machine learning and deep learning.

Yong Yu is now a professor in the Department of Computer Science of Shanghai Jiao Tong University (SJTU). Yong got his Master's degree in computer science from East China Normal University, Shanghai, in 1986. Yong is the head coach of SJTU ACM-ICPC team. His teams won the 2002, 2005 and 2010 ACM ICPC Championships. His research interests include semantic web, web mining, information retrieval and computer vision.