Twitter Spammer Profile Detection

Grace Gee, Hakson Teh

gracehg@stanford.edu, hakson@cs.stanford.edu

December 9, 2010

1. Introduction

Twitter is a free microblogging service that allows users to post messages, called tweets, up to 140 characters in length. Its value is in its ease of sharing and accessing user-generated content, including opinions, news, and trending topics. Thus, Twitter provides an opportunity to generate large traffic and revenue, especially since it has hundreds of millions of users.

However, these opportunities make Twitter a prime target of spammers. It is easy for humans to distinguish spammers from actual users, but the existence of spammers wastes user time and attention, puts users at risk in accessing malicious and dangerous content, and devalues Twitter's services and the overall online social network. In a 2009 study of Twitter demographics by Pear Analytics, spam profiles made up about 4% of all Twitter profiles [1]. Up to date, Twitter does not have an effective way of addressing its spammer problem, which has long been an issue. It currently allows two options for users to deal with spammers themselves:

1. Users can report spammers to the @spam profile, but many of the profiles reported are normal users, and it usually takes a few days before the actual spammer profiles are identified and deleted.
2. Users can submit an online help ticket, which again, can take up to a few days before it is addressed.

In this paper, we will discuss how we approached the problem of detecting spammer profiles on Twitter, how we collected training data, and how we implemented a supervised learning algorithm aimed at classifying spammer profiles and normal user profiles.

2. Problem Statement

The goal of this project was to implement a supervised learning algorithm that would be able to label a Twitter profile as a spammer or normal user, after being trained on an initial training set of spammer profiles and normal user profiles. The first step was identifying an initial set of

features that could be used by the learning algorithm to distinguish between spammer profiles and normal user profiles. Then we needed to collect a large enough set of profiles to accurately train the algorithm; this step was time-consuming since our research had concluded that there are no publicly-available sets of labeled (as spammer or normal user) Twitter profile data. We planned on collecting and manually labeling our own set of data using the Twitter API, as discussed in section 4. Training Data Collection. We implemented a Naïve Bayes algorithm first, as it is simple and a good initial test of how well a learning algorithm could do in identifying spammer profiles. Finally, after some error and data analysis, we used a more sophisticated algorithm, the support vector machine (SVM), to further improve our model.

3. Feature Selection

When choosing features to distinguish between spammer profiles and normal user profiles, we first took into account the different types of spammer profiles:

- *Duplicate Tweeters*: Tweet the same post multiple times
- *Promoters*: Link to other businesses, surveys, etc.
- *Phishers*: Pose as a normal user to acquire private user data
- *Fake Users*: Pose as a normal user (with non-spam tweets), but tweeting spam content periodically

From these types of spammers, we identified the following key features to be good distinguishers between spammer and normal user profiles:

- *Followers-to-Following Ratio*: Spammers are expected to follow a large number of people, but have few followers themselves. This number is expected to be low (< 1) for spammers.
- *Number-of-Tweets-to-Account-Lifetime Ratio*: Spammers are expected to have short account lifetimes, but a large amount of tweets. This number is expected to be high for spammers.
- *Average Time Between Posts*: Spammers are expected to have more posts than normal users on average, over a period of the same time. This number is expected to be low for spammers.
- *Posting Time Variation*: Spammers are expected to post at predictable, scheduled times. This number is expected to be low for spammers.

- *Max Idle Hours:* Spammers are not expected to be idle for long periods of time. This number is expected to be low for spammers.
- *Link Fraction:* Spammers are expected to post links in a majority of their tweets. This number is expected to be high for spammers.

4. Training Data Collection

We used two different techniques to collect spammer profiles and normal user profiles for our training set. To collect (likely) normal user profiles, we polled Twitter's public timeline through the Twitter API and gathered user profiles one by one. The user data was represented in JavaScript Object Notation (JSON) and a second API call was made to additionally include up to 200 tweets from each user. This data was subsequently written out of disk in compressed GZIP format and a human-readable HTML file of the user profile was pulled directly from Twitter's servers.

For the (likely) spammer profiles, we polled for mentions of "@spam" and extracted any username of the form "@username" within mentions of "@spam." We then pulled the profile of all such usernames exactly as we did before with the (likely) normal user profiles. Essentially, we captured profiles that were reported to "@spam" for spamming, assuming that most of the reported profiles had a high probability of being a spammer.

We then took this raw dataset and manually labeled each profile as being a "spammer" or "non-spammer" based on the HTML user profile. Foreign language profiles were discarded as there was no way for us to ascertain whether or not they were spam.

Post-labeling, the JSON format of the data files were then read and compiled into two separate matrices in Comma Separated Value (CSV) format corresponding to spammers and normal users. Features were extracted from the data and inserted in the CSV files, with the usernames representing the rows of each matrix and each feature representing a column of each matrix.

These two CSV files composed the training set of feature input data for our learning algorithms.

We used 70% of our collected data to train our algorithms and saved 30% of the data to test them. All code for the profile extractor and feature matrix compiler was written in Python.

5. Learning Algorithm Implementation

We first tried to implement a Naïve Bayes learning algorithm in MATLAB to classify the collected Twitter profiles. Using a total of 225 normal user profiles and 225 spammer profiles yielded an error rate of about 27%. This error was higher than our goal of achieving an error rate of less than 20%.

We then implemented a linear classifier SVM using National Taiwan University's Machine Learning Group's LIBLINEAR [2]. Using the default dual L2-regularized L2-loss support vector classification solver, cost value (C) of 1, and no bias, we obtained an error rate of 40% using the SVM.

The unusually high error of the initial run of the SVM compared to Naïve Bayes suggested that there was an issue with the implementation of the SVM. After some investigation, we identified the problems to be a lack of data scaling and inaccurate calibration of SVM parameters (C and gamma); we also decided to try some non-linear kernels for better classification using Chih-Chung Chang and Chih-Jen Lin's LIBSVM library [3].

In our final SVM implementation, we normalized the entire feature dataset, used LIBSVM's default Radial Basis Function (RBF) kernel, and performed 5-fold cross validation on the training set to determine the optimal SVM parameters C and gamma (C = 8192, gamma = 0.0078125, CV rate = 84.9359).

6. Results

Our final results from the improved SVM classification are as follows:

- 89.3% precision: percentage of profiles classified as spammers that were true spammers
- 86.2% recall: percentage of true spammers classified as spammers
- 89.6% accuracy (10.4% error): percentage of all correctly classified profiles

7. Conclusion

The high precision and recall suggests that using this particular SVM for detecting spammer profiles could be usable in practice. Most of the work, however, lies in feature engineering. We were not able to pursue highly technical features (e.g. cohesiveness of social graph) and those that would require a lot of time to develop (e.g. semantic analysis of tweets). It is very likely that precision and recall can be improved to at least 90% if the full power of Twitter's database is leveraged to create features that can strongly distinguish between spammers and normal users, and to use as a more cohesive training set.

This learning algorithm would be very useful if deployed internally to flag profiles as likely spam before being forwarded to a human for further review. However, we would caution against aggressive deployment of this system, where for example, the result from the algorithm is used to automatically disable profiles for review. Such an aggressive deployment should only be considered if precision can be raised above 99%, due to the risk of accidentally disabling a normal user's profile.

References

[1] "Twitter Study," Available: http://www.pearanalytics.com/blog/wp-content/uploads/2010/05/Twitter-Study-August-2009.pdf [Accessed: October 31, 2010]

[2] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A Library for Large Linear Classification, Journal of Machine Learning Research 9(2008), 1871-1874. Software available at http://www.csie.ntu.edu.tw/~cjlin/liblinear

[3] Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm