Fall 2009

Enhancing spammer detection in online social networks with trust-based metrics.

Alexander J. Murmann

San Jose State University

ENHANCING SPAMMER DETECTION IN ONLINE SOCIAL NETWORKS WITH TRUST-BASED METRICS

A Thesis
Presented to
The Faculty of the Department of Computer Science
San José State University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science

by
Alexander J. Murmann
December 2009

SAN JOSÉ STATE UNIVERSITY

The Undersigned Thesis Committee Approves the Thesis Titled

ENHANCING SPAMMER DETECTION IN ONLINE SOCIAL NETWORKS
WITH TRUST-BASED METRICS

by

Alexander J. Murmann

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

11/03/2009

Dr. Teng-Sheng Moh, Department of Computer Science Date

2009-11-03

Dr. Cay Horstmann, Department of Computer Science Date

Jon Pearce 11-3-09

Dr. Jon Pearce, Department of Computer Science Date

APPROVED FOR THE UNIVERSITY

12/4/09

Associate Dean Office of Graduate Studies and Research Date

ABSTRACT

ENHANCING SPAMMER DETECTION IN ONLINE SOCIAL NETWORKS WITH TRUST-BASED METRICS

by Alexander J. Murmann

As online social networks acquire larger user bases, they also become more interesting targets for spammers. Spam can take very different forms on social Web sites and cannot always be detected by analyzing textual content. However, the platform's social nature also offers new ways of approaching the spam problem. In this work the possibilities of analyzing a user's direct neighbors in the social graph to improve spammer detection are explored. Special features of social Web sites and their implicit trust relations are utilized to create an enhanced attribute set that categorizes users on the Twitter microblogging platform as spammers or legitimate users.

ACKNOWLEDGEMENTS

To my fiancée Ana Drucker and my parents Irmgard and Peter Murmann who support and love me in everything I do.

I also want to thank my advisor Professor Teng-Sheng Moh for his time and passion as well as Professor Cay Horstmann and Professor Jon Pearce for serving on the thesis commitee.

TABLE OF CONTENTS

CHAPTER

LIST OF TABLES

Table

LIST OF FIGURES

Figure

CHAPTER 1

INTRODUCTION

Unsolicited e-mail is a problem with which every Internet user is familiar. The amount of spam e-mails is increasing year by year, and the costs for society of coping with this problem are increasing with it. In 2005 losses of productivity and costs of related IT-investments were estimated to be well over $10 billion.

The increasing number of users of social networking Web pages makes them more attractive for spammers. Common spam fighting approaches usually try to filter unwanted e-mail on the basis of its content. This approach might also be useful on some social networks. However, the spam found on social networks might take forms that do not allow detection by content analysis. This might be the case on video platforms where a video's content would have to be analyzed. On other platforms, spam is propagated by behavior rather than content. Examples for spam propagation by behavior can be found on social-tagging platforms where spammers promote their content or microblogging platforms where someone can get users' attention by subscribing to their feed.

Online social networks not only present us with new forms of spam but also give us new ways to fight it. Social mechanisms might be utilized to penalize malicious behavior. We can also utilize the underlying social graph to detect spammers.

The purpose of this thesis is not to create the best possible spammer detection system for the Twitter platform, but rather to explore the possibilities of relying on a user's direct neighbors in the social graph to categorize the user as a legitimate user or spammer. I expand existing work on detection of spammers via known classification algorithms and propose an enhanced attribute set for user classification that is based on users' neighbors and underlying trust relations between users.

The problem will be defined in Chapter 2. In Chapter 3 existing solutions to fight spam and spam bots will be discussed. In Chapter 4 a solution to differentiate legitimate users and spammers will be presented. Discussion of the conducted experiments is in Chapter 5. The solution will be evaluated and the findings discussed in Chapter 6. In Chapter 7 I will draw a conclusion and point out potential future research topics.

CHAPTER 2

PROBLEM

In October 2008, 76.4% of all e-mails sent were spam, according to a monthly study published by Symantec Inc. (Doug Bowers 2008). This is an increase of about 7% compared to 2007. In 2006 the costs were already estimated to be well over $10 billion (James Carpinter 2006). These costs arise because users have to manually filter wanted e-mail from the spam e-mail. This results in a loss of productivity. Another factor adding to these costs is IT-infrastructure purchased to fight spam.

These enormous amounts of spam exist because the business of spam is highly lucrative and depends on scale. The general response rate to spam is very low, but spammers send huge amounts of spam and make much money off a single response. Carpinter (James Carpinter 2006) gives the following numbers to describe this phenomenon:

> Commissions to spammers of 25-50% on products sold are not unusual. On a collection of 200 million email addresses, a response rate of 0.001% would yield a spammer a return of $25,000, given a $50 product.

Because spam is such a profitable business, spammers are increasingly targeting social networks and utilizing them in different ways. The first case of mass spam on myspace.com occurred in 2005[1] . A survey conducted by Harris Interactive for

[1] Mike Masnick, tech dirt, http://www.techdirt.com/articles/20050218/1558248_F.shtml

Cloudmark Inc.[2] revealed that 83% of social networking site users received spam. Spam on social networks is taken seriously enough by users that 66% of users said that they were at least somewhat likely to change platforms if the number of received spam became significant. However, 37% already had noticed an increase in spam during the last six months.

Most approaches to detect spam in e-mail are content based. The e-mail's text is analyzed and checked against lists of words or phrases common to spam. However, spam on social networks can consist of a very short text or even be a non-verbal action such as following someone's news feed or adding someone as a friend. Some of these behaviors actively alter the social graph underlying the social networking services. This way, approaches based on the social structure are not only unaffected but are actually supported by some of the techniques used by spammers.

2.1 Definitions

To clarify further discussion some terms are defined in this section.

2.1.1 Online Social Networks

In this work I use the terms *online social network* and *social network site* synonymously. I use the definition provided by Boyd et al. (Boyd and Ellison 2007) which defines social network sites as those that allow users to perform three main actions:

(1) Construct a public or semi-public profile within a bounded system

(2) Articulate a list of other users with whom they share a connection

(accessed 11-16-2008)

[2] Cloudmark Inc., http://www.harrisinteractive.com/NEWS/newsletters/clientnews/2008_Cloudmark.pdf (accessed October 2009)

(3) View and traverse their list of connections and those made by others within the system

The most common way to describe relations among members of social networks in general is the social graph. According to Freeman (Freeman 2000) Jacob L. Moreno established this way of representing social patterns in his work in 1932. He used undirected graphs with actors as nodes and edges to indicate relations between nodes. In 1934 he expanded this system to use directed edges to display directed relations between actors. This type of graph is commonly referred to as a social graph and it plays the central role in social network analysis and therefore its terminology will be used frequently in this work.

2.1.2 Twitter

Since I use the Twitter microblogging service as my test platform, I need to define Twitter-specific terms. Twitter allows users to write short notes and messages not exceeding 140 characters. The sum of a user's messages is referred to as a user's feed. Other users can subscribe to the feed. This process is called *following* and creates a directed relation between two users. If user A follows user B, user B is called a friend of A. User A will be a follower of B. The process of unsubscribing from a user's feed is commonly referred to as *unfollow*. Throughout this work I will also refer to the sum of friends and followers a user has, as his peers. Peers are a user's direct neighbors in the social graph.

2.1.3 Spam

Definitions of spam are usually focused on e-mail spam. Common criteria to define spam are:

- No current relationship between sender and recipient exists (James Carpinter 2006; Cormack and Lynam 2005)
- Applicability to many other potential recipients (James Carpinter 2006)
- Sending messages in bulk [3]

These definitions are not applicable to all online social networks. With online social networks that focus on media sharing, such as video platforms or social tagging systems, most communication is not one-to-one communication, but rather broadcast. Messages do not have a distinct recipient, instead they are communicated to every user of the platform who is interested. For this reason definitions can't rely on the relation between spammer and user or on the message's applicability to a single recipient. Definitions of spam for these platforms need to resort to content specific definitions. Benevenuto et al. (Benevenuto, Rodrigues, Almeida, Almeida, and Ross 2008) for example, use the following definition for video-response spam on the YouTube platform: "We define a video response spam as a video posted as a response to an opening video, but whose content is completely unrelated to the opening video."

Twitter can be seen as a media sharing platform with the shared medium being short messages. The way spammers get attention is by following someone's Twitter feed and thus, by default, triggering a message to the user resulting in a form of one-to-one communication. This allows us to define spam on the Twitter platform in a way that's closer to the definition of e-mail spam, resulting in the following two criteria, that both have to be fulfilled:

- No current relationship between sender and recipient exists

[3] Spamhaus Project, The, http://www.spamhaus.org/definition.html (accessed 10-11-2009)

- Following other users in bulk.

The applicability to many recipients is still not applicable on Twitter, since the spammer cannot influence the content of the notification e-mail to the followed user.

CHAPTER 3

LITERATURE REVIEW

In this chapter I cover existing work on the topic of anti-spam methods. Bots are often used to spread spam on the Internet and some methods to detect them also rely on different behavior from legitimate, human users. Therefore spam bot detection is closely related to the purpose, as well as the approach of this thesis, and will briefly be covered as well. Since my approach utilizes trust mechanisms in the social network, I will also give a review of theories about online trust and existing systems based on these theories.

3.1 Spam in Social Networks

Different approaches exist to cope with spam on social networking Web sites. According to Heymann et al. (Heymann, Koutrika, and Garcia-Molina 2007) these can be categorized into three different groups.

- Detection-based
- Prevention-based
- Demotion-based

Detection-based approaches identify spam or spammers and then either delete them or display them as likely spammers. A common example from this category are spam filters for e-mail.

Prevention-based systems try to prevent spammers from getting into the system. They do this by authenticating users before they are allowed on the platform or by putting up obstacles that prevent malicious behavior; for example, using CAPTCHAs to keep bots out, or creating costs for the contribution of messages.

Demotion-based approaches rank spam lower than non-spam. This is a strategy commonly applied in Web search, where potential spam is ranked low and therefore appears at the bottom of the list of search results.

My approach is a detection-based approach, therefore the focus of this review will be on existing work in that category.

3.2 Bot Detection

Bots have become a problem for various online applications. The spectrum reaches from e-commerce with fears about stolen content (Poggi, Berral, Moreno, Gavald, and Torres 2007) over Web registration forms of Web mail providers (Schluessler 2007) to online games being played by bots instead of humans (Golle and Ducheneaut 2005).

Of interest for this thesis are methods that try to detect bots based on their different behavior. Other approaches exist to keep bots out of the system, such as the commonly used CAPTCHA. CAPTCHA is an abbreviation for “Completely Automated Public Turing test to tell Computers and Humans Apart.” These are tasks that are thought to be easy for humans, but hard for computers. These works are far from the focus of this work and are therefore not covered here.

The AUGURES system proposed by Poggi et al. (Poggi, Moreno, Berral, Gavaldà, and Torres 2007) uses the different behavior of buying customers and non-buying visitors to prioritize traffic. The AUGURES system consists of two parts. It has an offline component analyzing log files to classify users into different categories. It reconstructs a user's click pattern on the page and trains two Markov models. One Markov model is trained on the click stream of buying customers and one model is trained on non-buying visitors. During runtime, a user's click stream will be compared with the two Markov models and traffic will be prioritized accordingly. In an advanced version (Poggi, Berral, Moreno, Gavald, and Torres 2007) the system is used to detect content stealing bots as well.

Their tests found that 74% of all bots were identified as such and 81% of all users classified as bots were bots. However, that means that 19% of all detected bots were false positives. This number is too high for practical use since every human prevented from accessing the page is potentially a customer lost. Therefore this solution should only be considered if keeping bots out is the higher priority or if it is viable to check correct classification manually.

Differentiating buyers from non-buyers is of particular interest since human users were categorized based on their behavior. Interestingly this categorization was more precise than the one between bots and buyers. Of all buyers 91% were categorized correctly and 94% of all users categorized as buyers turned out to be actual buyers.

Sion et al. (Sion, Atallah, and Prabhakar 2002) used a similar approach to detect intruders on Web portals. The focus here is not on detecting bots but on detection of illegal access to data by comparing data access patterns. The so-called Hyper-data Shadow is constructed by analyzing users' data access patterns. This structure describes a user's common transitions from page to page using hyperlinks. The Hyper-data Shadow can be represented as a weighted graph with Web pages or data being nodes and with edges representing hyperlinks or transitions. The more often a link is being used or data is being accessed, the heavier the edge will be.

Changes in users' access patterns create a mismatch with the Hyper-data Shadow, which may indicate a possible security breach, since someone other than the legitimate user might be logged in.

Sion et al. also suggest this method for bot detection. Here the system could be trained on human users. Other users' access patterns then would have to be matched against the Hyper-data Shadow created with the human user data. I also see the possibility to train the system on access patterns used by known human spammers. Matching users' access patterns against the spammer's Hyper-data Shadow might then be used for spammer detection since similarity indicates that a user might be a spammer.

3.3 Spammer Detection

Using information contained in social graphs to detect spam has already been proposed outside of online social networking sites. Boykin and Roychowdhury (Boykin and Roychowdhury 2005) propose an algorithm that uses the social graph created by e-mails sent and received by users. Addresses that occurred together in e-mail headers are connected with an edge in the graph. In this graph Boykin and Roychowdhury find strongly connected components which are then analyzed. Because spammers send e-mails to huge numbers of recipients at the same time, but never send e-mails to each other, spammers will end up in different components than legitimate users. Based on a component's size, maximum degree and clustering coefficient, these components are categorized as components consisting of spammers or non-spammers.

The service provided by online social networking sites relies on the benefits provided by its underlying social structure. Therefore constructing the social graph for these platforms is a much more trivial task than it is in the case of e-mail described by Boykin and Roychowdhury.

Social bookmarking systems such as del.icio.us or digg.com contain a social network component and are likely targets of spammers and content promoters. Spammers can simply add a page they want to promote. Different schemes exist to increase the rank of a promoted Web site on a bookmarking platform. Examples include adding unfitting, but popular tags to an element or using multiple accounts to vote for an entry. Krause et al. (Krause, Schmitz, Hotho, and Stumme 2008) describe a system to detect spammers on these platforms. The outline of the process used can be seen in figure 3.1. The data set is split into training and test data. A categorization algorithm builds a model to distinguish between legitimate users and spammers based on the entries from the training data. This model is then tested on the test set. Attributes resembling a users data and behavior on the platform are grouped into four different feature sets and used as input for the categorization algorithm:

Profile features Commonly entered information by the user upon registration, such as user name, real name and e-mail address and attributes of these. Notably, it was shown that the number of digits used in those values is useful to distinguish between spammers and legitimate users.

Location based features Numbers of users in domain and top level domain, as well as number of spammers with the same IP address.

Activity based features Attributes describing a users tagging behavior and the time difference between registration and first post.

Semantic features Several values relating to usage of tags by legitimate users and spammers and co-occurrences of tags and users. Tags are also compared to a list of words that are known to frequently occur together with spam content.

These attributes were used as input for different categorization algorithms. The best result was achieved with a Support Vector Machine which yielded a F1-Value of 0.986.

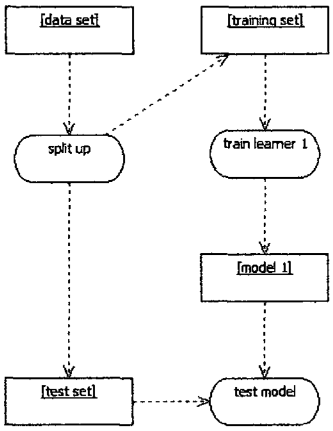


Figure 3.1: Basic approach used by Krause et al. (Krause, Schmitz, Hotho, and Stumme 2008) as well as Benevenuto et al. (Benevenuto, Rodrigues, Almeida, Almeida, and Ross 2008; Benevenuto, Rodrigues, Almeida, Almeida, and Gonçalves 2009)

Benevenuto et al. (Benevenuto, Rodrigues, Almeida, Almeida, and Ross 2008; Benevenuto, Rodrigues, Almeida, Almeida, and Gonçalves 2009) use a similar approach to detect spammers and content promoters on the YouTube video sharing platform. The YouTube platform allows users to post a video as a response to another video. Responding videos will appear as a video response under the responded video. This makes it attractive for spammers to post videos as a response to a popular video. Another tactic, used by content promoters, is to post unrelated videos as a response to a video they want to promote, since many responses increase the ranking of a video on the platform.

Benevenuto et al. utilize the social graph created by video responses. Edges in the graph connect users who responded to each other's videos. They then apply a similar system as used by Krause et al. (Krause, Schmitz, Hotho, and Stumme 2008) to detect spam on social bookmarking sites. Again, different attributes are given to a Support Vector Machine to detect spammers. In their work attributes from the following three sets were used:

User-based features Attributes related to a user's profile, such as number of uploaded videos and number of favorited videos.

Video-based features Aggregated values of videos uploaded by the user. The aggregation is done twice. Once for all videos and once only for video responses. Example attributes include average rating and number of ratings.

Social network features Features based on the social graph created by the video responses.

3.4 Online Trust

In this work I utilize information about a user's peers to decide if a user is a spammer. Part of this is to capture the underlying, implicit trust relation between follower and friend. Although we are very familiar with the concept of trust from every day life, finding a clear, general definition of trust is difficult.
Shapiro (Shapiro 1987) describes the efforts to define trust as a "confusing potpourri of definitions applied to a host of units and levels of analysis." Nonetheless, we need some kind of definition. For this purpose I choose the definition of trust by Gambetta (Gambetta 1988) that was described as *Reliability Trust* by Jøsang et al. (Josang, Ismail, and Boyd 2007):

> Trust is the subjective probability by which an individual, A, expects that another individual, B, performs a given action on which its welfare depends.

3.4.1 Forming Online Trust

Many attempts to model trust creation have been made. Notably, X. Zhang and Q. Zhang (Zhang and Zhang 2005) tried to create an integrated model for online trust forming mechanisms. They incorporate personal aspects such as beliefs, attitudes, and technical aspects such as Web page appearance. However, I want to limit this thesis to a basic model of online trust forming which is based on two fundamental concepts also used by X. Zhang and Q. Zhang: Social Exchange Theory (SET) and Expectation-Confirmation Theory (ECT).

SET developed by Blau (Blau 1964) explains social interaction with other individuals. According to SET, social interaction depends on initial trust that future returns of a relation will be higher than the current costs.

Oliver's (Oliver 1980) ECT explains how consumer trust is built. According to ECT, consumers enter a transaction with a certain expectation about the product. Once they have purchased the product, the satisfaction depends on the relation between the original expectation and the actual experience. According to the degree to which the expectation is fulfilled, the customer will be satisfied and future expectations will be adjusted. According to X. Zhang and Q. Zhang (Zhang and Zhang 2005), the same mechanism applies to online transactions and online trust.

These two concepts are important for the attempt to gain information from a user's peers since according to SET, all parties will be holding expectations to gain some kind of value from each other. According to ECT, if these expectations are not fulfilled, users' future expectations, or in other words their trust in future fulfillment of their initial expectations, will decrease. If their expectations drop low enough, they may assume that current costs will not be covered by future value. Consequently they might unfollow their friend. Clearly spammers' expectations will vary hugely from legitimate users' expectations. Legitimate users likely expect to receive informative or entertaining posts from their friends. Spammers are not interested in their friends' content, but in getting their attention. Therefore they will most likely not unfollow a friend because they are unsatisfied by the content provided. This should result in differences among followers and friends of spammers and legitimate users.

3.4.2 Reputation

Many platforms, such as yelp.com or epinions.com, try to summarize many users' trust or lack of trust into businesses and products. The aforementioned platforms aggregate explicit ratings by users through which their expectations or trust of the product or business is expressed. Search engines try to formalize the implicit trust expressed by Web links or citations in order to rank their search results. Reputation, according to Jøsang et al. (Josang, Ismail, and Boyd 2007) can "...be considered as a collective measure of trustworthiness (in the sense of reliability) based on the referrals or ratings from members in a community." It is that reputation that search engines and community based rating systems try to formalize and utilize.

3.5 The PageRank Algorithm

The PageRank algorithm is the best known representative of trust propagation algorithms that try to calculate a reputation value without collecting explicit ratings. That approach is closely related to some of the approaches followed in this work and a basic understanding of the PageRank algorithm will be useful for later sections.

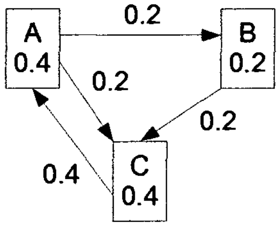


Figure 3.2: Example of a converged calculation of the simplified PageRank

PageRank is related to the basic technique of citation counting or backlink counting. As the term implies, citation counting calculates references pointing to an object (such as a Web page or a document) and then ranks all objects accordingly. This has its weaknesses since a single link from an important Web page might be more significant than many links from many unimportant Web pages. Page et al. (Page, Brin, Motwani, and Winograd 1999) actually describe PageRank as "...providing a more sophisticated method for doing citation counting."

Page et al. use a simplified version of PageRank called R to explain the basic concept:

> Let u be a web page. Then let F be the set of pages u points to and B be the set of pages that point to u. Let $N_u = ||F_u||$ be the number of links from u and let c be a factor used for normalization (so that the total rank of all web pages is constant).

They then go on to define the following formula:

$$R(u) = c \sum_{v \in B_u} \frac{R(V)}{N_v}$$

Thus the value assigned to a Web page u will be propagated in equal parts to all pages it links to, as can be seen in figure 3.2. This basic version has problems but gives a deep enough understanding of the PageRank algorithm for our purpose. See

Page et al. (Page, Brin, Motwani, and Winograd 1999) for a discussion of the full PageRank algorithm.

CHAPTER 4

SOLUTION

My solution takes the basic approach used by Krause et al. (Krause, Schmitz, Hotho, and Stumme 2008) and Benevenuto et al. (Benevenuto, Rodrigues, Almeida, Almeida, and Ross 2008; Benevenuto, Rodrigues, Almeida, Almeida, and Gonçalves 2009), then computes and adds additional features to the feature set. The added features can be separated into two different categories: Features describing a user's peers and features that try to describe the trust that is given to a user by followers. Both metrics are based on the assumption that a spammer will have different peers than a legitimate user. The first feature basically attempts to summarize friend and follower attributes by simply computing average values for both friends and followers of a user. The second feature set uses a trained classifier to predict if a user's followers are spammers or legitimate users and calculates different metrics based on this. Additionally, a trust metric is introduced that tries to grasp the trust that followers have in the user. This trust metric is a more elaborate version of basic backlink counting and is similar to the PageRank algorithm in this regard. However, calculation of my trust metric is specific for spam detection and non-recursive.

4.1 Feature Set

Since the enhanced feature set is crucial for this work, we will take a close look at all the features used in the basic attribute set and the two additional feature sets that constitute the enhancement.

4.1.1 Basic Feature Set

The values in the basic feature set originally resembled all the attributes available for a user via the Twitter API. Preliminary tests showed that the most valuable feature of a user was the account age. This might be due to the fact that I acquired most spammer account names from the platform twitspam.org (see Chapter 5). Once an account is listed with twitspam.org it is more likely to be deleted from the Twitter platform, therefore resulting in a lower average account age with other metrics being altered accordingly. Therefore I decided to calculate all values that commonly vary by account age on a per-day basis. This is done to minimize the influence on test results that twitspam.org being the source for spammer account names might have. For example, a user generally will pick up more followers over time. I do not use the attribute *number of followers*, but instead divide the number of followers by the account age in days, resulting in *followers per day.*

I also added the number of digits in the account name as an additional feature. It has been shown on other platforms (Krause, Schmitz, Hotho, and Stumme 2008) that spammers are more likely to use many digits in their account names than legitimate users. Together this resulted in the following nine basic user features:

- friend-follower ratio
- number of posts marked as favorites

- friends added per day
- followers added per day
- account is protected?
- updates per day
- has URL?
- number of digits in account name
- reciprocity

4.1.2 Average Peer Values

In this work I try to explore how much user information can be gained by looking at a user's peers. The simplest way is by calculating average values for a user's peers. These values are calculated separately for friends and followers, thus allowing a more differentiated evaluation and doubling the amount of features. I calculate the average values for the following features:

- friend-follower ratio
- updates per day
- friends added per day
- followers added per day
- reciprocity
- account is protected?

In the case of the boolean attribute *account is protected?* the percentage of all protected friend/follower accounts was calculated. This resulted in 12 new features, since features were aggregated for friends and followers separately.

4.1.3 Trust-Based Metrics

Backlink counting is a common technique in ranking search results on the Web. The most prominent example making use of this technique is Google's PageRank algorithm (Page, Brin, Motwani, and Winograd 1999). A similar approach is applied to ranking scientific publications by counting citations. Examples for this are Citeseer (http://citeseer.ist.psu.edu/), as described by Giles et al. (Giles, Bollacker, and Lawrence 1998) and Google Scholar (http://scholar.google.com/), as described by Noruzi et al. (Noruzi 2005). Benevenuto et al. (Benevenuto, Rodrigues, Almeida, Almeida, and Ross 2008) used the PageRank algorithm on the social graph created by video responses on youtube.com. Since the algorithm is applied to users and not to Web pages it is referred to as UserRank in this context. They then used UserRank as part of the feature set describing a user. However, the UserRank algorithm needs to be run for several iterations to converge. For example, on a data set containing 161 million pages, it needs forty-five iterations to converge. Based on backlink counting I try to create a similar metric that does not need as many iterations but still utilizes the trust that followers have in a friend to categorize users.

The UserRank algorithm needs the UserRank values for all followers of a user in order to calculate a user's UserRank. This is the basis for the recursive nature of the UserRank/PageRank algorithm and the reason why it needs many iterations. In the basic trust metric I assign the same value, 1, to all followers of a node which is then spread equally over all of the user's friends.

Let u be a user, then F_u are all friends of node u and B_u be all followers node u has. The number of friends of node u be $N_u = |F_u|$. Then the formula for the basic version of my trust metric is:

$$\text{TrustMetric}(u) = \sum_{v \in B_u} \frac{1}{N_v}$$

This basically comes down to the PageRank algorithm without the recursion, which saves computation time but is almost a fall back to simple backlink counting. A follower who is a spammer will contribute the same value as a highly popular legitimate user. The UserRank algorithm solves this problem by calculating the importance of users and assigning a weight to them accordingly which results in the recursion. Since I am not interested in finding the most important or popular Twitter user but just in finding spammers, I only need to distinguish between trustworthy and non-trustworthy users in order to obtain a good metric to weight the values propagated from followers. Therefore I estimate if a user's followers are likely to be spammers. Only if they are predicted not to be spammers, their values are forwarded to friends. A spammer will follow everyone and therefore they are expected to provide no value when evaluating a user and are left out. In addition to this modification we also use two other modifiers to explore the behavior and value of our newly introduced trust metric, resulting in the following three modifiers:

legit accumulate only the values coming from users who are predicted to be legitimate users

capped accumulate only values coming from up to 200 users

squared use $\sum_{v \in B_u} \frac{1}{N_v * N_v}$ instead of $\sum_{v \in B_u} \frac{1}{N_v}$

All these modifications are tested in all possible combinations. As an additional attribute, the ratio between predicted spammers and predicted legitimate users following a user is calculated.

4.2 Model Creation

To calculate some of the values in the second feature set, a model is needed to predict if a user is a spammer or a legitimate user. Therefore an additional step is added in which a model is generated to categorize a user's followers as spammers or legitimate users. As seen in Figure 4.1, the training set is first used to generate the prediction model *model1* using an arbitrary categorization algorithm. This model is then used to categorize a user's followers into spammers and legitimate users. These predictions are then used to calculate the ratio of spammers and legitimate users among a user's followers, as well as to calculate a modified version of the trust metric which only includes predicted legitimate users. Based on this new, extended feature set, I go on to create prediction model *model2* which is used to categorize user's whose feature set was enhanced with *model1*.

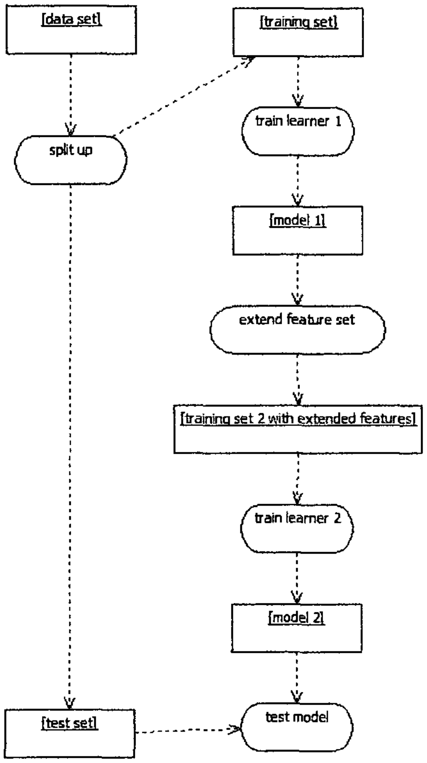


Figure 4.1: Process used for this work: In the first step a model is created to extend the basic feature set based on predictions about a user's followers. In the second step another model is created based on the extended feature set.

CHAPTER 5

EXPERIMENTS

5.1 Test Data

To collect a large enough data set, I started out with a number of known spammers and known legitimate users. First, account information was collected for these users using the Twitter API. Starting from this user set, I collected account information for peers of users already known to the system. For each user I collected information for up to 200 friends and followers.

Most of the account names of spammers were acquired using the Web page twitspam.org, where users can submit the names of suspected spammers. A few users were added as spammers which got my attention while I was collecting data. To obtain trusted users, I added Twitter users whom I was following. This resulted in a data set containing 77 spammers and 155 legitimate users. In addition, for each of these users, information for up to 200 of their followers was acquired. Because many users have friends or followers in common, information on more than 200 of their followers is available for some users in the data set.

5.2 Experiments

In preliminary tests I evaluated different categorization algorithms to create the prediction model in the first and second step. An extended data set was created

Table 5.1: Preliminary accuracies for different learner combinations

1st\2nd step	JRIP	J48	SOM	Naïve Bayse
JRIP	93.0%	93.0%	71.5%	79.9%
J48	89.2%	91.9%	73.9%	84.8%
SOM	89.8%	87.6%	76.8%	85.1%
Naïve Bayse	88.2%	86.0%	75.9%	78.3%

using the first algorithm to which the second algorithm was applied with ten-fold cross-validation.[1] It is important to use cross-validation to minimize the chance that an unfortunate split between training- and test-set spoils test results. This might happen if elements in the training set are overly representative or unrepresentative for the data set as a whole. It is also important to make sure that all training and test sets contain a representative mix of all classes (spammers and legitimate users in our case).

I used implementations of the algorithms as provided by the WEKA package. [2] Table 5.1 shows the accuracy achieved with each combination of algorithms. It can be seen that for this purpose JRIP, which is an implementation of the RIPPER rule learner that was created by William W. Cohen (Cohen 1995), turned out to perform the best, with J48, which is an implementation of Ross Quinlan's C4.5 algorithm (Quinlan 1993), as a close second.

To compare the performance between the basic attribute set and the two extended feature sets, I run a ten-fold cross-validation ten times which results in one hundred different result sets. This is done for the basic feature set, the basic

[1] For cross-validation the test data is split up into n subsets. In step one, subset 1 is being used as the test set and the subsets 2 to n are used as the training set. In step two, the second subset will be used as the test set and the remaining set function as the training set. This is repeated until each subset was used once as the test set. If n has the value ten we talk about ten-fold cross-validation which is the commonly used number of used folds.

[2] http://www.cs.waikato.ac.nz/ml/weka/

features extended with average peer values, the basic feature set extended with the values based on implicit trust and predictions on follower being spammers, and a final test involving the basic feature set, as well as both additional feature sets.

5.2.1 Evaluation Criteria

To evaluate if the extended attribute set is able to improve the performance of a classification algorithm, I use different established metrics on test results acquired with and without the extended attribute sets. I calculate *accuracy*, *precision*, *recall*, *F1*, and finally draw a Receiver Operating Characteristic Curve (ROC curve) to evaluate the test results. Since all these metrics are highly common in data mining, I will only give a brief overview. For exact definitions of these metrics see Tan, Steinbach, and Kumar (Tan, Steinbach, and Kumar 2005).

The most basic metric used is *accuracy*, which is the share of all instances that are classified as belonging to their actual class. The accuracy does not provide any information on instances where a class tends to be misclassified or in which way the misclassification most commonly took place. In spammer detection, it is worse if a legitimate user is being falsely classified as a spammer, than if a spammer is misclassified as a legitimate user. Falsely accusing legitimate users of being spammers is likely to drive them off the platform. Therefore, I also calculate the *precision* (p), which determines the fraction of actual positives in the group of instances classified as positives. *Precision* will be high if the number of correctly classified spammers is high and the number of false positives is low. *Recall* (r) measures how many elements of a class (usually the positive class) are correctly classified. In our case, I use it to measure how many of all actual spammers are detected. In addition, I use the *F1 measure*, which is the harmonic mean between *precision* and *recall*: $F = 2 * \frac{p*r}{p+r}$. Thus, *F1* takes both measurements into consideration but penalizes a big difference between both values.

The ROC curve plots the false positive rate against the true positive rate. Thus, one can see the trade-off between catching more spammers and falsely classifying more legitimate users as spammers. This way the curve shows which results can be achieved by using appropriate probability cut-offs (Witten and Eibe 2005).

CHAPTER 6

FINDINGS

Tables 6.1 and 6.2 show that the extended attribute sets were able to generally improve the results. It is important to note that the *precision* was improved in all combinations, as it shows that fewer legitimate users are being classified as spammers. This compensates for the loss in general accuracy that C4.5 takes when using the average values for peers in addition to the basic attribute set. C4.5 gets its best results with a combination of all attribute sets. Although the results for the trust-based features with the basic feature set are only slightly worse and actually have a lower false positive rate. For the RIPPER algorithm, however, trust-based features in combination with the basic feature set perform best and achieve a *F1* measure that's 0.03 higher than the *F1* of all features combined. This combination also yields better results than C4.5 with any feature set.

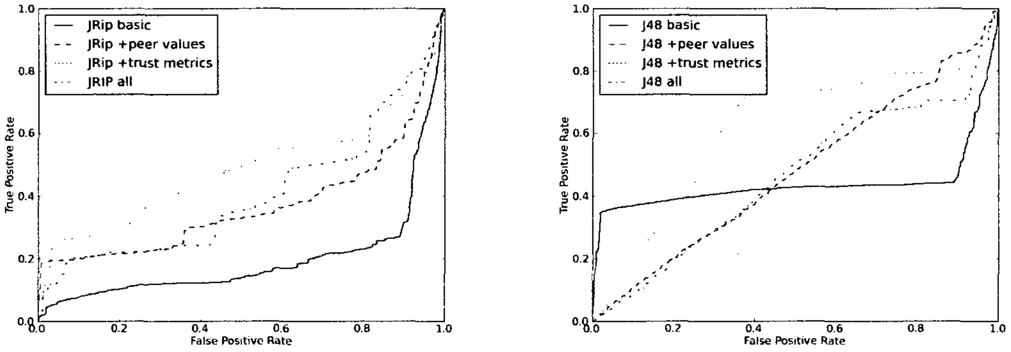
Table 6.1: Evaluation metrics for RIPPER algorithm with the different extended feature sets

Metric	basic	basic + peer values	peer values	basic + trust	all features
Precision	0.79	0.80	0.75	0.88	0.84
Recall	0.84	0.83	0.71	0.85	0.85
F1	0.81	0.81	0.73	0.87	0.84
Accuracy	0.87	0.87	0.82	0.91	0.90

Table 6.2: Evaluation metrics for C4.5 algorithm with the different extended feature sets were used

Metric	basic	basic + peer values	peer values	basic + trust	all features
Precision	0.80	0.81	0.72	0.85	0.86
Recall	0.85	0.79	0.67	0.85	0.86
F1	0.83	0.80	0.69	0.85	0.86
Accuracy	0.88	0.87	0.80	0.90	0.90

Figure 6.1 shows the ROC curves generated with the C4.5 learner and the RIPPER learner. The curves using the C4.5 algorithm show that the curve for the feature set including the peer based features and basic features is below the curve using only the basic feature set until a false positive rate of about 0.4 is reached. The feature set using trust metrics and *spammers to legit followers* together with the basic features after a false positive rate of 0.21 surpasses the basic attribute set, and only near the end is surpassed by the peer based values and the combination of all feature sets. Surprisingly the combination of all feature sets had the best results in Table 6.2. This difference between the ROC curve and the other measurements used might come from an uneven distribution of data points in the result set.



(a) ROC Curve using RIPPER

(b) ROC Curve using C4.5

Figure 6.1: ROC curves with extended feature sets: (a) and (b) show the ROC curves with basic feature set and the two feature sets and basic feature set in combinations.

The ROC curves calculated for the Ripper algorithm paint a much clearer picture. All extended feature sets perform consistently better than the basic feature set. The best results for most false positive values was achieved by the feature set including trust metrics and the predicted amount of spammers among one's followers. Only for very low and high false positive rates a combination of all features and the feature set with peer values perform better. It is interesting to see that the basic attribute set seems to perform much better in combination with C4.5 than with RIPPER. Overfitting might be an explanation for this.

To measure the contribution of each feature I calculated the information gain and the chi square values for the all features. I calculated the values on a data set, that was constructed by adding up the extended data sets created using the Ripper algorithm with ten-fold cross-validation for follower classification. The top ten ranked attributes and their values can be seen in the two tables 6.3 and table 6.4. The ratio between legitimate followers and followers who are spammers in both rankings turned out to be the highest rated value. Different versions of the trust metric also consistently ranked pretty high. Their usefulness is confirmed by the very good results that they achieved in the other tests. It is interesting to notice that in both measurements the average friend-follower ratio for a user's friends ranks 7, but the same value for one's followers had a information gain and chi square value of 0. The friend-follower ratio was rated very high by both metrics as well. This might change in the future because notification e-mails generated by Twitter reporting on new followers now include information on the number of friends and followers a user acquired. A high number of friends is a fairly good indicator if someone is a spammer or not. Since Twitter's notification e-mails now contain these numbers, users might stop behaving as intended by spammers. However, spammers can easily adjust to this changed behavior by unfollowing users and thus keeping their friend count low. I have already encountered some spammers with reasonably low friend numbers during these studies, which might be an indicator that this is already happening. I expect that the number of friends added daily will thus lose its importance.

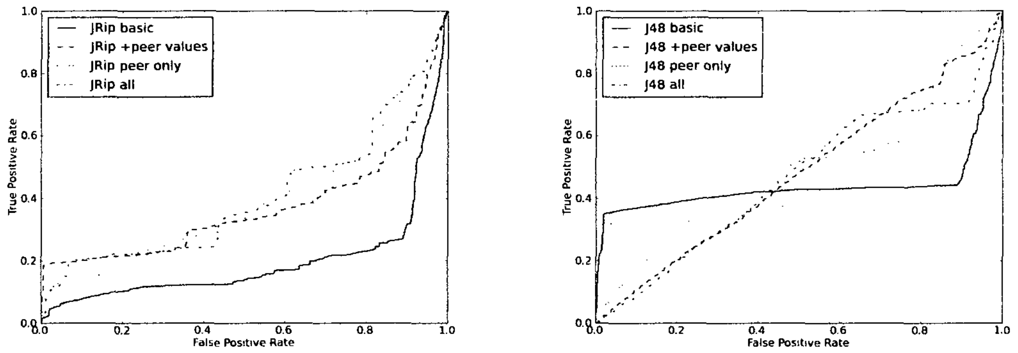
Table 6.3: Top ten chi square values for data set extended with JRip

Attribute	Chi square value
spammers to legit followers	128.68
friends per day	106.72
trust metric legit.	105.49
friend-follower ratio	101.23
trust metric legit. capped	94.8697
trust metric	88.78
friend-follower average for friends	81.54
average protected for followers	80.57
trust metric legit. square	79.93
trust metric legit. square capped	74.99

Table 6.4: Top ten information gain values for data set with extended JRip

Attribute	Information gain
spammers to legit followers	0.48
friend-follower ratio	0.35
friends per day	0.34
trust metric legit.	0.34
trust metric legit. capped	0.29
trust metric	0.29
friend-follower average for friends	0.27
average protected for followers	0.25
trust metric legit. square	0.24
average protected for friends	0.24

Due to the mixed impact that using the features based on peer averages had on the outcomes, I ran additional tests only involving those metrics. The curves in Figure 6.2 compares these features' performance with some of the other feature sets used.



(a) ROC Curve using RIPPER

(b) ROC Curve using C4.5

Figure 6.2: ROC curves focusing on peer values: (a) and (b) show the ROC curves comparing the feature set only based on the average peer values compared to other feature sets.

Notably the peer based features seem to do better with the RIPPER algorithm than the basic feature set does. It might seem surprising that using more attributes does not necessarily yield the best results. However, that can be explained with the common issue of overfitting. A look at a rule set that was used by the RIPPER algorithm and a decision tree used by C4.5 during the tests, supports this theory. The rules generated by the RIPPER algorithm displayed in code listing 1 shows that RIPPER uses a very slim set of rules. The decision tree generated by C4.5 as shown in code listing 2 uses a multitude of criteria and goes down even to depth eight to make a decision. This makes the tree much more prone to overfitting than RIPPER's rules are.

```
(trust metric legit <= 0.629386)
 and (friends per day >= 2.817241) => class=true
(friend-follower ratio >= 1.517537)
   and (friend-follower ratio for friends >= 1.022858) => class=true
(updates per day <= 0.106952) => class=true
=> class=false
```

Code Listing 1: Example rule set generated by WEKA's JRIP implementation of the RIPPER algorithm during test runs.

```
friend-follower ratio <= 1.727273
|   spam-follower ratio <= 0.177489
|   |   trust metric <= 2.929861: true
|   |   trust metric > 2.929861: false
|   spam-follower ratio > 0.177489
|   |   updates per day <= 0.108696
|   |   |   updates per day for followers <= 1.885: false
|   |   |   updates per day for followers > 1.885: true
|   |   updates per day > 0.108696
|   |   |   friend follower ratio for friends <= 1.022858: false
|   |   |   friend follower ratio for friends > 1.022858
|   |   |   |   trust metric <= 2.418246
|   |   |   |   |   has url? = true
|   |   |   |   |   |   friend follower ratio for friends <= 1.314457
|   |   |   |   |   |   |   updates per day for followers <= 10.08254: true
|   |   |   |   |   |   |   updates per day for followers > 10.08254: false
|   |   |   |   |   |   friend-follower ratio for friends > 1.314457: false
|   |   |   |   |   has url? = false: false
|   |   |   |   trust metric > 2.418246: false
friend-follower ratio > 1.727273: true
```

Code Listing 2: Example decision tree generated by WEKA's J48 implementation of the C4.5 algorithm during test runs.

CHAPTER 7

CONCLUSION AND OUTLOOK

The much improved classification results and the high values received by the additional attributes for both the chi-square statistic and information gain show that a user's peers indeed tell much about the nature of a user. The RIPPER algorithm was able to obtain consistently better results on the extended attribute set as compared to the basic attribute set. The C4.5 algorithm was able to detect a higher number of spammers with the extended attribute set than it could detect on the basic attribute set. I am curious to see if the same methods work on other online social networks. Applying the same methods to another platform should be fairly straightforward.

Although the additional features improved spammer detection, I see several ways of improving the system further. Some users might be more careful in evaluating whom they follow than others. This is supported by the fact that many users just follow everyone who follows them. This will result in a much higher number of spammers in their friend list. I now only use the number of users that are being followed to weight the value added to a friend's trust metric. I could use the ratio of predicted spammers among a user's friends to weight this value even further and make it more meaningful. However, this would require a much larger data set than I have now.

I would also like to explore community belongingness as an indicator. I expect that legitimate users are more often part of a closely connected subgroup, in contrast to spammers who just follow everyone and therefore will be connected to many otherwise divided groups.

Another promising possibility is to combine the current system with a system that analyzes a user's access patterns such as the AUGURES system used by Poggi et al. (Poggi, Berral, Moreno, Gavald, and Torres 2007) or the Hyper-data Shadow used by Sion et al. (Sion, Atallah, and Prabhakar 2002). These system's predictions could be easily integrated in the extended feature set. However this would require access to detailed log files that could only be acquired by the platform owner.

BIBLIOGRAPHY

Benevenuto, F., T. Rodrigues, V. Almeida, C. Almeida, Jussara Zhang, and K. Ross (2008). Detecting spammers and content promoters in online video social networks. In *Proceedings of the 4th international workshop on Adversarial information retrieval on the web*, pp. 45–52. ACM.

Benevenuto, F., T. Rodrigues, V. Almeida, J. Almeida, and M. Gonçalves (2009). Detecting spammers and content promoters in online video social networks. In *SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, New York, NY, USA, pp. 620–627. ACM.

Blau, P. (1964). *Exchange and Power in Social Life*. Wiley.

Boyd, D. and N. B. Ellison (2007, November). Social network sites: Definition, history, and scholarship. *Journal of Computer-Mediated Communication 13*(1-2).

Boykin, P. O. and V. P. Roychowdhury (2005). Leveraging social networks to fight spam. *Computer 38*(4), 61–68.

Cohen, W. W. (1995). Fast effective rule induction. In *Proceedings of the Twelfth International Conference on Machine Learning*, pp. 115–123. Morgan Kaufmann.

Cormack, G. and T. Lynam (2005). Trec 2005 spam track overview. *Text Retrieval Conference*.

Doug Bowers, Dermot Harnett, C. E. (2008, nov). The state of spam. web.

Freeman, L. (2000). Visualizing social networks. *Journal of Social Structure 1*.

Gambetta, D. (1988). *Can We Trust Trust?*, pp. 213–237. Basil Blackwell.

Giles, C. L., K. D. Bollacker, and S. Lawrence (1998). Citeseer: an automatic citation indexing system. In *International Conference On Digital Libraries*, pp. 89–98. ACM Press.

Golle, P. and N. Ducheneaut (2005). Keeping bots out of online games. In *Proc. of the ACM SIGCHI International Conference on Advances in Computer Entertainment Technology*.

Heymann, P., G. Koutrika, and H. Garcia-Molina (2007, Nov.-Dec.). Fighting spam on social web sites. *Internet Computing, IEEE 11*(6), 36–45.

James Carpinter, R. H. (2006, march). Tightening the net: A review of current and next generation spam filtering tools. In *Apricot 2006.*

Josang, A., R. Ismail, and C. Boyd (2007). A survey of trust and reputation systems for online service provision. *Decision Support Systems 43*(2), 618 – 644. Emerging Issues in Collaborative Commerce.

Krause, B., C. Schmitz, A. Hotho, and G. Stumme (2008). The anti-social tagger: detecting spam in social bookmarking systems. In *AIRWeb '08: Proceedings of the 4th international workshop on Adversarial information retrieval on the web*, New York, NY, USA, pp. 61–68. ACM.

Noruzi, A. (2005). Google scholar: The new generation of citation indexes. *Libri 55*, 170–180.

Oliver, R. L. (1980). A cognitive model of the antecedents and consequences of satisfaction decisions. *Journal of Marketing Research 17*, 460–469.

Page, L., S. Brin, R. Motwani, and T. Winograd (1999, November). The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab. Previous number = SIDL-WP-1999-0120.

Poggi, N., J. L. Berral, T. Moreno, R. Gavald, and J. Torres (2007). Automatic detection and banning of content stealing bots for e-commerce.

Poggi, N., T. Moreno, J. L. Berral, R. Gavaldà, and J. Torres (2007). Web customer modeling for automated session prioritization on high traffic sites. In *UM '07: Proceedings of the 11th international conference on User Modeling*, Berlin, Heidelberg, pp. 450–454. Springer-Verlag.

Quinlan, J. R. (1993). *C4.5: programs for machine learning.* San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Schluessler, T. (2007). Is a bot at the controls? detecting input data attacks. In *Proc. of 6th Annual Workshop on Network and Systems Support for Games: Netgames 2007.*

Shapiro, S. P. (1987). The social control of impersonal trust. *The American Journal of Sociology 93*(3), 623–658.

Sion, R., M. Atallah, and S. Prabhakar (2002). On-the-fly intrusion detection for web portals. Technical report, In Proceedings of IEEE ITCC.

Tan, P.-N., M. Steinbach, and V. Kumar (2005). *Introduction to Data Mining.* Addison-Wesley.

Witten, I. H. and F. Eibe (2005). *Data Mining - Practical Machine Learning Tools and Techniques.* Morgan Kaufmann.

Zhang, X. and Q. Zhang (2005). Online trust forming mechanism: approaches and an integrated model. In *ICEC '05: Proceedings of the 7th international conference on Electronic commerce*, New York, NY, USA, pp. 201–209. ACM.

Index