HSpam14: A Collection of 14 Million Tweets for Hashtag-Oriented Spam Research

Surendra Sedhai
School of Computer Engineering
Nanyang Technological University, Singapore
surendra001@e.ntu.edu.sg

Aixin Sun
School of Computer Engineering
Nanyang Technological University, Singapore
axsun@ntu.edu.sg

ABSTRACT

Hashtag facilitates information diffusion in Twitter by creating dynamic and virtual communities for information aggregation from all Twitter users. Because hashtags serve as additional channels for one's tweets to be potentially accessed by other users than her own followers, hashtags are targeted for spamming purposes (*e.g.,* hashtag hijacking), particularly the popular and trending hashtags. Although much effort has been devoted to fighting against email/web spam, limited studies are on *hashtag-oriented spam* in tweets. In this paper, we collected 14 million tweets that matched some trending hashtags in two months' time and then conducted systematic annotation of the tweets being spam and ham (*i.e.,* non-spam). We name the annotated dataset **HSpam14**. Our annotation process includes four major steps: (i) heuristic-based selection to search for tweets that are more likely to be spam, (ii) near-duplicate cluster based annotation to firstly group similar tweets into clusters and then label the clusters, (iii) reliable ham tweets detection to label tweets that are non-spam, and (iv) Expectation-Maximization (EM)-based label prediction to predict the labels of remaining unlabeled tweets. One major contribution of this work is the creation of HSpam14 dataset, which can be used for hashtag-oriented spam research in tweets. Another contribution is the observations made from the preliminary analysis of the HSpam14 dataset.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing

Keywords

Twitter; tweets; hashtag; spam

1. INTRODUCTION

Popularity of micro-blogging sites such as Twitter and Weibo has drawn attention of not only legitimate users but also spammers. Spammers create accounts to promote their products and/or services or even steal accounts of legitimate users for the same. The issues become more complicated when third-party applications post to Twitter on behalf of the legitimate users with the permission granted by the users. As the result, tweets posted on one's timeline may not be solely composed by the account owner. The tweets from third-party applications may contain spam information. Grier *et al.* reported that 0.13% of message advertised on Twitter are clicked which is two orders of magnitude higher than email spam [13]. Due to the growing interest of spammers in micro-blogging sites, spam becomes a pervasive problem on Twitter. In their recent study, Thomas *et al.* reported that 17% of spam users exploit hashtags to make their tweets visible in search and trending topics [35]. Hijacking trending topics and popular search terms for spamming has also been observed in web spam and blog spam [14, 40].

Most existing studies on Twitter spam focus on identification of spammers for account blocking. This approach is less effective for spammers who may act as legitimate users by posting non-spam content regularly. This approach may even hurt a legitimate user who happens to grant permission to a third-party application which posts spammy tweets under her username. This legitimate account may be blocked because of the spam tweets posted by the third-party application. While identifying and blocking spammer accounts remain a crucial and challenging task, *tweet-level spam detection* is essential to fight against spamming in a more fine-grained level. In this paper, we take the first step to create a large scale tweet collection for hashtag-oriented spam research. Among all different types of spamming, we mainly focus on the identification and annotation of tweets with hashtags, because hashtags serve as channels to increase the visibility of spam tweets. The cost of hijacking hashtags is very low with the availability of trending hashtags published on many web sites including Twitter.

We make two major contributions in this research. The first contribution is the creation of **HSpam14**, a collection of 14 million annotated tweets in English for hashtag-oriented spam research. We collected trending hashtags on daily basis from `hashtags.org` and used the collected hashtags as query keywords to collect tweets through Twitter streaming API, for two months' time. We then annotated the 14 million tweets with four major steps:

(i) *Heuristic-based tweets selection* to search for tweets that are likely to be spam. The selection was based on popular hashtags, and keywords related to adult content, money gain and others.

(ii) *Near-duplicate cluster based annotation* to group similar tweets into clusters and then label the clusters. Tweets are clustered by using MinHash based algorithm. The assumption is that spam tweets are likely to be posted many times with minor changes for more visibility. The annotation process includes both manual annotation and confident k-nearest neighbor based annotation with human intervention in the process. Tweets posted by the same users and tweets containing links from the same domains are also clustered and annotated in this step.

SIGIR'15, August 09 - 13, 2015, Santiago, Chile.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-3621-5/15/08 ...$15.00.
DOI: http://dx.doi.org/10.1145/2766462.2767701.

(iii) *Reliable ham tweets detection* to label tweets that are non-spam (also known as ham). Using the labeled tweets in the last step, we have a reasonably large number of examples which are utilized to build a classifier to detect reliable ham tweets. A reliable ham tweet does not contain any word that appears with higher probability in spam tweets than other tweets (*i.e.,* the labeled ham tweets and unlabeled tweets).

(iv) *EM-based label prediction* to predict the labels of the remaining unlabeled tweets using an Expectation-Maximization (EM) algorithm. At this step, we have 2.387 million spam tweets and 4.897 million ham tweets already labeled. We then utilize EM-based label prediction algorithm to label the remaining tweets.

Through manual inspections after each annotation step, the precision of the labels is above 0.94. We believe such quality of labeling is good enough for analyzing hashtag-oriented spam tweets for tweet-level spam detection.

The second contribution made in this paper is the preliminary analysis of the HSpam14 dataset. In our analysis, we made the following observations. First, spam and ham tweets do not differ much in the usage of mentions, but the usage of hashtags is significantly different. More specifically, 76% of ham tweets have no hashtags, and 97% of them have at most 2 hashtags. For spam tweets, only 37% of them have no hashtags, and 40% of them have 3 or more hashtags. Second, spam tweets are likely to contain hashtags in capital letters and with the word "follow" as part of the hashtag. The hashtags mainly present in ham tweets are usually in lower case letters or mixture of lower and upper cases, and the hashtags are more topic specific. We also propose a measure named *Spammy Index* to quantify the extent a hashtag is being used in spam tweets. Last, we observe that spammy hashtags are more likely to co-occur with spammy hashtags.

2. RELATED WORK

Spam is prevalent in all types of online communication medium. There are different types of spam: email spam [9], web spam [33], video spam [2], micro-blog spam [1], comment spam [28], and review spam [17], to name a few. Spammers have used different strategies in different platforms, and they also keep changing their strategies to hide themselves from spam detection systems [5]. Recently, crowdsourcing systems have also been exploited for posting malicious content [21, 37]. Spam detection systems use a wide range of information such as content, link, user behavior, as well as HTTP session information to tackle spam problem [33]. Next, we brief the spam on other platforms (*e.g.,* email, web, comment, and product review), and then review the spam on Twitter.

Spam on Email, Web, Comment, and Product Review. Email spam delivers unwanted advertisement, fraud scheme, promotion, or computer malwares designed to hijack the recipient's computer. To identify such emails, machine learning techniques as well as whitelisting and blacklisting of senders, domains or IP addresses have been used [9]. Various techniques such as language model, content duplication, link-based trust and distrust propagation, graph based techniques, and user behavior modeling have been applied to fight against web spamming which can be in the form of content spamming, link spamming, cloaking and click spamming, and comment spamming [5, 28, 33].

Online product reviews are valuable source of information for potential customers to find the opinions of other users about products before purchase. Due to the importance of product review, spammers are attracted to promote their products and defame the competitors' products. Review content and reviewers' behavior are utilized to detect spam reviews [17, 24, 29]. For example, a topic model based approach was proposed for spam detection in product reviews in [23]. Features derived from part-of-speech tag, n-gram, and sentiment of the reviews, have been used to detect such spams [22, 31]. As spammers are likely to post many pieces of similar content, near duplicate detection techniques have been exploited to identify such similar content and also the spammers [3, 6, 38]. Duplicate detection and classification technique have also been used for spam review detection [17, 18]. They considered all the duplicate reviews as spam reviews. However, it is not valid to consider all duplicate micro-blog posts as spam posts.

Spam on Twitter. Thomas *et al.* reported that there is an underground market in Twitter network which provides spam-as-service in a recent study [35]. In their study, they also reported that 77% of spam accounts are identified by Twitter within the first day of creation, and 92% of spam accounts within first three days of creation. The authors also made the observations that 89% of spam accounts have fewer than 10 followers and 17% of spam users exploit hashtags to make their tweets visible in search and trending topics [35]. This is similar phenomena as in web spam and blog spam, where trending topics and popular search terms are hijacked for spamming [14, 40]. However, there is no existing study specifically on hashtag-oriented spam detection.

The reflexive reciprocity indicates that many users simply follow back when they are followed by someone for the sake of courtesy [39]. It is not difficult for spammers to gain a relatively large number of followers in Twitter. Realizing this fact, social honeypot are deployed to harvest deceptive spam profiles by Lee *et al.* [19]. The profiles that are captured by social honeypot are classified using standard classification technique. Other features derived from user demographics, follower/following social graph, tweet content and temporal aspect of user behavior have also been analyzed and used to identify content polluter [20]. Castillo *et al.* analyzed the credibility of tweets on trending topics based on users' tweeting and retweeting behaviors, tweet content and link present in the tweets [4]. We believe that many of these features can be used for hashtag-oriented spam detection. However, before such research can be conducted, a benchmark dataset needs to be constructed.

User experience in Twitter is adversely affected by spam social bots. Social bots are programs that automatically produce content and interact with humans on social media. Social bots post tweets about popular and focused topics and follow back the users who follow the bots [27]. Using such simple strategies bots could get high influence in Twitter and may pollute timeline of users. These spam bots can be detected by social honeypot traps [19] and also based on features derived from temporal behavior, tweet content, and user profile [7, 10]. In our data collection, we also observe that many tweets share very similar format and are posted by bots. However, not all Twitter users are equally vulnerable to social bots, and the behaviors that make users more susceptible to social bots have been studied in [36].

Other studies on Twitter spam include the studies on the link structure of the users and the links to external resources. As in Web, spammers have also used link farming technique to promote content in Twitter [12, 34]. Tan *et al.* [34] reported that spammers and non-spammers belong to different communities in user graph. Spammers may also include unrelated links with trending words (*e.g.,* hashtags) in tweets [1]. To detect such spam tweets, matrix factorization model was proposed by Hu *et al.* [15] to learn lexical information from external spam resources. The authors further proposed online learning algorithms to cope with the fast evolution of social spammer in [16].

3. DATA COLLECTION AND STATISTICS

3.1 Data Crawling

To collect data for hashtag-oriented spam tweet research, one key issue is to identify candidate hashtags. The tweets containing these candidate hashtags are then collected and labeled. Without the luxury of accessing all tweets or all hashtags, we rely on the trending topics (*i.e.,* hashtags) reported on Hashtags.org.[1] Hashtags.org reports trending hashtags in three categories: *trending up*, *trending down*, and *most popular hashtags*. As expected, we observe that the hashtags in *most popular hashtags* category do not change much for many days. To cover more varieties of hashtags in our data collection, we used the hashtags in *trending up* and *trending down* categories as query keywords to search for tweets on daily basis. On each day, we collect the trending hashtags in these two categories and we then use each collected hashtag as a query keyword to collect tweets using Twitter's streaming API for that day.[2] A tweet is collected through the API if it contains the query keyword as a *hashtag*, *word*, or *link* in its content. On average 135 hashtags were used as query keywords on each day. The data collection process last for two months, May and June 2013.

In total, we collected 24.36 million tweets, which were published by 11.97 million users. The collected tweets contain 20.21 million hashtags and 6.97 million hyperlinks. After resolving the short URLs, there are 3.43 million distinct URLs. Ignoring the deadlinks and the links that require authentication, 2.05 million web pages were successfully downloaded. However, we have not fully utilized these web pages for this labeling process.

Among the collected tweets and web pages, 14.07 million tweets and 1.37 web pages are in English.[3] In our following discussion, we mainly focus on the 14.07 million (or simply 14 million in approximation) tweets in English. The same set of tweets will be annotated with spam and ham labels in the next section.

3.2 Data Statistics

User Distribution. The 14 million English tweets were posted by 7.25 million users and the distribution of the number of tweets per user is plotted in Figure 1(a). Shown in the figure, the data demonstrates a typical power-law distribution that is widely observable in many social related studies. While more than 4.83 million users each contributed only one tweet, the most active user contributed 754 tweets to this collection. Recall that this collection covers two-month period and the tweets were collected by using trending hashtags as queries (*i.e.,* biased sampling). The few extremely active users may post a larger number of tweets regularly or post many tweets targeting on some trending hashtags. Because we do not have the full profiles of the users in this data collection, the underlying reasons cannot be verified. In our following analysis, we mainly focus on the content of the tweets, particularly the usages of hashtags and mentions.

Usages of Hashtags and Mentions. Figure 1(b) plots the frequency distributions of hashtags and mentions. This figure shows that the usages of both hashtags and mentions follow power-law distributions with slightly different slopes in the log-log plot. On the one hand, this figure suggests that hashtags are more commonly used than mentions in tweets, particularly the extremely popular ones. On the other hand, a small set of usernames are frequently being mentioned in tweets. A mention in a tweet is also a form of hyperlink, and clicking a mention leads to the user profile page where much more information about this user is displayed (*e.g.,* user description and recent tweets). For this reason, mention can be intentionally used for spamming purposes, similar to hashtags[4].

Table 1: The top-20 most popular hashtags. All these 20 hashtags appeared as trending up/down hashtags which were used for sampling the tweets.

Hashtag (#)	Freq.	Hashtag (#)	Freq.
TEAMFOLLOWBACK	666,328	SougoFollow	197,525
TFBJP	527,176	ipad	195,375
gameinsight	510,504	FOLLOWBACK	177,109
android	341,240	THF	165,762
OPENFOLLOW	332,857	FOLLOWNGAIN	149,916
FF	293,748	500aday	146,005
androidgames	286,706	AUTOFOLLOW	141,214
RETWEET	250,992	MUSTFOLLOW	138,040
RT	235,137	TEAMHITFOLLOW	136,043
IPADGAMES	232,335	MUSIC	129,852

Table 1 lists the top-20 most popular hashtags in the 14 million tweets. Observe that the most popular hashtag #TEAMFOLLOWBACK appears in more than half million tweets out of the 14 million. Moreover, 40% of the top-20 most popular hashtags contain the word "follow". Many of the usernames among the popular mentions contain the word "follow". In fact, a quick check of the top-20 most mentioned usernames shows that 11 usernames[5] have been suspended by Twitter. Among the suspended usernames, nearly half contain the word "follow".

Figure 1(c) plots the per-tweet distribution of hashtags and mentions. Note that the y-axis is in log-scale. Both distributions show the same interesting pattern: (i) the number of tweets having 1 to 4 hashtags/mentions drops significantly along the increase of the number of hashtags/mentions; (ii) the rate of dropping becomes much smaller for tweets containing 4 to 10 hashtags/mentions; and (iii) the number of tweets again drop significantly with the increase of contained hashtags/mentions from 10 onwards. As expected, very few tweets contain more than 14 mentions or more than 18 hashtags, given the 140-character length limit of tweets.

4. TWEET ANNOTATION

There are three major challenges in manually labeling millions of tweets: (i) *large number of instances*, *i.e.,* it is infeasible to manually check every single tweet to determine its label as in many other labeling efforts [8], (ii) *limited amount of information carried by each tweet* because of its shortness, and (iii) *lack of standard criteria* for the classification of spam and ham tweets.

To address the first two challenges, we propose to group the near-duplicate tweets into clusters. Spammers keep changing their behavioral patterns to avoid detection; however, they keep on posting similar promotional tweets exploiting popular keywords. Hence, clustering of near-duplicate tweets captures promotional tweets effectively despite change in behavioral patterns of spammers. Because the tweets in each near-duplicate cluster are nearly identical to each other, the same label applies to all the tweets in the same cluster. The near-duplicate clustering also helps in partially addressing the second challenge. One of the major characteristics

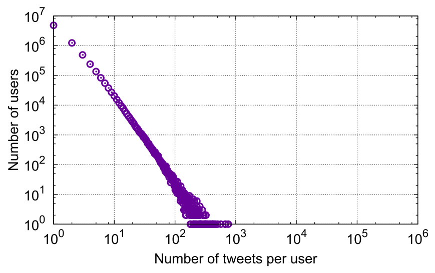
[1] `https://www.hashtags.org/trending-on-twitter/` Note that the trending hashtags identified by Hashtags.org are based on the site's proprietary algorithm.

[2] We used Twitter database server code available at `http://140dev.com/`.

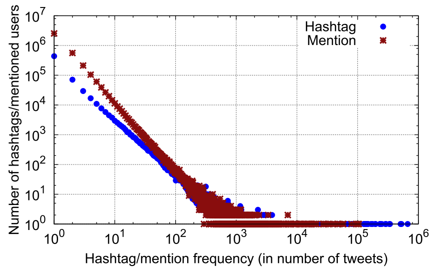
[3] We used language detection library for Java available at `http://code.google.com/p/language-detection/`

[4] `https://blog.twitter.com/2012/shutting-down-spammers`

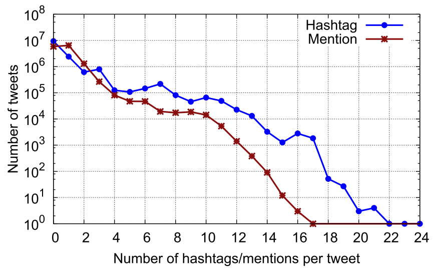
[5] In Twitter, a user may change her username at any time. Mentioning the same username at different time points may refer to different users.



(a) User frequency distribution



(b) Hashtag/mention frequency distribution



(c) No. of hashtags/mentions per tweet

Figure 1: Distributions of users and hashtag/mention usages in the 14 million English tweets

of spammers is to post near-duplicate content in repetitive manner. If a larger number of near-duplicate tweets are from very few users, then it is relatively easier for us to label spam tweets even if each tweet contain very limited semantics. During the labeling process, we may also access all tweets by a particular user in our data collection to get more contextual information. Regarding the last challenge, we reference to the existing studies on spam detection (*e.g.,* tweets about quick money gain, adult content are considered spammy). In the context of hashtag-oriented spam detection, we also consider tweets containing semantically unrelated hashtags and quick followership gain as spam tweets.

Figure 2 illustrates the four main steps in the annotation process. We now brief the four major steps. More details of the steps and the labeling criteria are described in the next sub-sections.

Heuristics-based Tweet Selection. It is reasonable to assume that majority of tweets are ham. To minimize human labeling effort, it is necessary to label only the tweets that are more likely to be spams. To this end, we selected the tweets that contain most popular hashtags (our assumption is that these hashtags are likely to be hijacked for spamming). We also select the tweets that contain money gain, adult content related keywords, and promoting purposes.

Cluster-based Manual Annotation. Selected tweets are grouped into near-duplicate clusters. A subset of clusters are then labeled with spam or ham, and all the tweets in the same cluster share the same label. The near-duplicate clusters are obtained by *MinHash* algorithm [3]. With an initial set of clusters labeled, we adopt k nearest-neighbor approach to "grow" the labels as in [8]. A cluster is labeled by kNN if the prediction of the label is very confident (based on the percentage of the nearest neighbors in the same class), and manually labeled otherwise. Spam tweets may be posted by some spam user accounts. For the top 10,000 users who posted the largest number of tweets in our dataset, we conduct user-based tweet clustering (*i.e.,* all tweets in each cluster are posted by the same user) and label the clusters. Similarly, based on the domain names of the links embedded in tweets, we cluster the tweets that contain links from the same domain, and label the clusters. The top 10,000 most frequent domains in our dataset are considered.

Reliable Ham Tweet Prediction. By the assumption that majority tweets are ham, a large number of tweets, particularly the unselected tweets left by the heuristics based selection, remain unlabeled. Here, we adopt the approach in *learning from positive examples only* in classification [25] to determine reliable ham tweets.

Label Prediction by EM Algorithm. After getting a reasonably large number of reliable labels of both spam and ham tweets, the EM algorithm by Nigam *et al.* [30] for learning from labeled and unlabeled data is adopted to predict the labels of the remaining tweets.

4.1 Heuristics-based Tweet Selection

We heuristically target on the tweets that are most likely to be spam, with the assumption that most tweets are ham. More specifically, we select three sets of tweets: (i) tweets containing any of the top-100 most popular hashtags in our collection, (ii) tweets containing adult content related keywords, and (iii) tweets containing keywords related to quick money gain, lucky draw, free gift *etc.*.

The first set of tweets is selected based on the heuristic that spammers are likely to exploit popular hashtags to make their content visible to broader audience. Table 1 lists the top-20 most popular hashtags, which is a subset of the 100 selected hashtags. The second set of the selected tweets aligns with most other spam detection tasks in email and web domains. A list of keywords related to adult content is used for this purpose.[6] For the third set of tweets selection, we used the following 23 keywords to capture the tweets related to quick money gain, lucky draw, and free gift: awesome, business, hurray, earn, bargain, deal, adventure, money, click, facebook, want, incredible, gift, money, amazon, ebay, amazing, prize, lucky, check out, shopping, sale, giveaway. Note that, there are tweets repeatedly posted by users with similar content, for promoting purposes related to major websites like Facebook, Amazon, and eBay. Therefore, these website names are included as keywords.

As the result, 7.10 million tweets posted by 3.93 million users were selected by keyword match. A tweet is selected if the tweet contains any of the selected hashtags or keywords in its content.

4.2 Cluster-based Manual Annotation

4.2.1 *Near-Duplicate Clustering*

To label the tweets more effectively, we group the near-duplicate tweets into clusters using MinHash algorithm [3]. The preprocessing of the tweets includes removal of links, removal of mentions (*i.e.,* @username), removal of the '#' symbols but keep the words in the hashtags, and removal of stop-words using the stop-word list provided in Lucene[7].

Both MinHash and SimHash are widely used for detecting near-duplicate documents [32]. Here, we adopt the MinHash algorithm on tweet's shinglings for duplicate detection [26]. We consider two tweets t_1 and t_2 are (near-)duplicates if minimum hash values of their *uni-gram*, *bi-gram*, and *tri-gram* representations are the same. More specifically, let $h(\cdot)$ be a hash function, and let tweet t be a n-word sequence $\langle w_1, w_2, \ldots w_n \rangle$. The minimum hash value of uni-gram representation of t is $h(w_i)$ iff $h(w_i) \le h(w_j)$ for any $1 \le i, j \le n$ and $i \ne j$. The minimum hash value of bi-gram

[6]We used the stem version of the keywords under pornographic category listed at `http://nymphs.org/RevealDirtyWordList.txt` In total there are 80 keywords in the list, and after stemming, there are 65 word stems.

[7]`http://lucene.apache.org/core/`

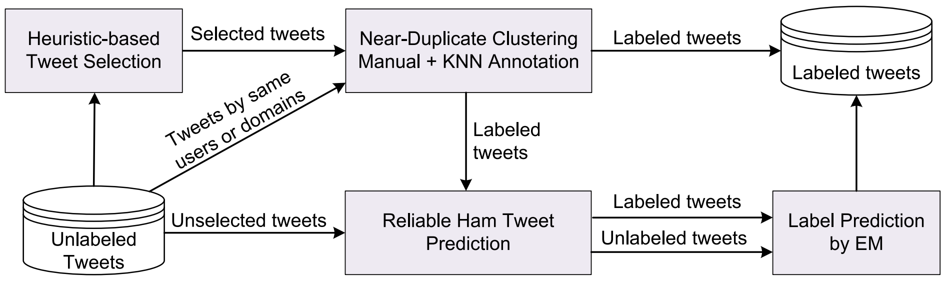


Figure 2: Overview of the four major steps in labeling spam/ham tweets

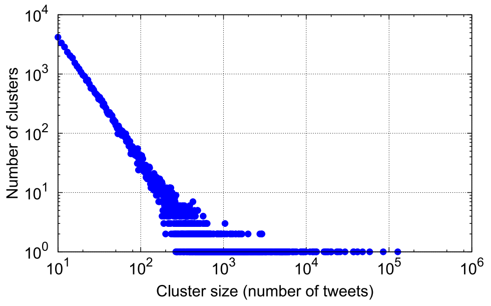


Figure 3: Size distribution of the near-duplicate clusters

Table 2: Example tweets from two near-duplicate clusters

1. • RT @username: #Money SEC chief White remains mum on money market fund reforms LINK1 #TonyRocha
   • #Forex: SEC chief White remains mum on money market fund reforms LINK1 LINK2
   • UPDATE 1-U.S. SEC chief White remains mum on money market fund rules LINK1 #news
   • SEC Chief White Remains Mum On Money Market Fund Rules: The top securities regulator remained tight-lipped on ... LINK1
2. • RT @username: [Retweet Only If You Want New Follower] #TEAMFAIRYROSE #OpenFollow #Follow #TFBJP #500aday #RT #R_Family #MaxVIP
   • RT @username: [Retweet Only If You Want New Followers] #TEAMFAIRYROSE #FOLLOWNGAIN #Follow #TFBJP #500aday #RT #R_Family #THF

Table 3: Measures of intra-/inter-cluster samples of tweets

Measure	Intra-cluster	Inter-cluster
Jaccard coefficient	0.9169 ± 0.0026	0.0187 ± 0.0006
Cosine similarity	0.9510 ± 0.0017	0.0350 ± 0.0012
Length difference	0.3933 ± 0.0196	5.5895 ± 0.0673

representation of t is $h(w_i w_{i+1})$ iff $h(w_i w_{i+1}) \leq h(w_j w_{j+1})$ for any $1 \leq i, j \leq n-1$ and $i \neq j$. The minimum hash value of tri-gram representation is computed in a similar manner. For simplicity, let $h_1(t)$, $h_2(t)$, and $h_3(t)$ be the minimum hash values of uni-gram, bi-gram, and tri-gram representations of tweet t. Then t_1 and t_2 are near-duplicates iff $h_1(t_1) = h_1(t_2)$, $h_2(t_1) = h_2(t_2)$ and $h_3(t_1) = h_3(t_2)$. In our Java implementation, we used the hashCode() method in String class to get the hash value of a n-gram (*i.e.,* a word or a shingling). The clustering process is computationally effective, and can be achieved in one scan of the tweets.

Result and Evaluation. Because our purpose is to label tweets more effectively, the clusters containing fewer than 10 tweets are ignored from manual annotation. The remaining 40,549 clusters contain 3.13 million tweets posted by 1.52 million users. Figure 3 plots the size distribution of these 40,549 clusters. Again, the plot demonstrates a power-law distribution with many clusters each contains 10 or slightly more tweets and very few extremely large clusters. The largest cluster contains 127,559 near-duplicate tweets, mainly due to hashtags #gameinsight, #android, and #androidgames. An example tweet is "RT @username: I've collected 103,501 gold coins! LINK1 #android #androidgames #gameinsight".[8]

In manual annotation, all the tweets in the same near-duplicate cluster will be annotated with the same label (*i.e.,* spam or ham). For this purpose, we must ensure that the tweets in the same cluster are indeed similar to each other, or in other words, are near-duplicate. Table 2 shows two randomly selected near-duplicate clusters containing 4 and 2 example tweets respectively, from each cluster. The examples show that tweets in the same clusters are indeed very similar to each other. However, there is a need to quantitatively evaluate the quality of the clusters in terms of tweet similarity. To this end, we randomly selected 4000 clusters (*i.e.,* about 10% all clusters) and measured the intra-cluster and inter-cluster tweet similarities. For intra-cluster tweet similarity, two tweets are randomly selected from each cluster; for inter-cluster tweet similarity, two tweets are randomly selected from two different clusters without replacement.

Table 3 reports tweet similarity in terms of Jaccard coefficient and cosine similarity. Cosine similarity is computed based on the bag-of-words model with *TFIDF* weighting. The length differences between the randomly selected pairs of tweets in number of words are also reported. Shown in Table 3, tweets from the same cluster show high similarity in terms of both Jaccard coefficient and cosine similarity with values above 0.91 and 0.95 respectively. On the other hand, inter-cluster similarity of tweets is dramatically lower at 0.019 and 0.035 respectively for Jaccard coefficient and cosine similarity. In terms of length difference, tweets from the same cluster are nearly identical with length difference of 0.39 word, while a pair of tweets from different clusters differ on average 5.59 words.

In summary, the clustering process is able to produce clusters with high intra-cluster similarity and low inter-cluster similarity. Tweets in the same cluster are of very similar length.

4.2.2 *Manual Annotation of Near-Duplicate Clusters*

Labeling tweets to be spam or ham is non-trivial for two reasons. First, there is no very specific definition of "spam". However, many definitions, including the description in Wikipedia[9], contain the phrase "unsolicited message", and the words "advertisement"

[8] We mask the usernames and links in all example tweets in this paper.

[9] http://en.wikipedia.org/wiki/Spamming

Table 4: Initial manual annotation of near-duplicate clusters

Label	No. of clusters	No. of tweets	No. of users
Spam	1,104	1.63 million	0.61 million
Ham	1,000	0.19 million	0.17 million

and "repeatedly". Second, because of the length limit, tweets are very short, and often do not come with context. The notion of near-duplicate clusters partially addresses these two challenges. Each near-duplicate cluster lists the near-"repeated" tweets and it is easy to identify whether the repeated tweets are from very few users or from some specific apps. Because of the grouping of similar tweets, it is also relatively easy to identify whether these similar tweets were posted because of some important events/topics (*e.g.,* the 4 tweets in the first cluster in Table 2) or were posted for advertisement purposes (*e.g.,* the 2 tweets in the second cluster in Table 2). In other words, we are able to infer the context of the tweets in a collective manner. By considering the tweets in each near-duplicate cluster, the users who posted these tweets, and also the usage of hashtags and links, we are able to label the near-duplicate clusters.

We label ***tweets in a duplicate-cluster to be spam*** if the tweets in this cluster satisfy ***one of the five conditions***: (i) for quick money gain or quick gain of followers; (ii) repeatedly posted for advertisement, promoting products/services, and request for retweet or offering benefits; (iii) about adult content; (iv) contain many unrelated but popular hashtags; (v) automatically posted by bots/apps for multiple times with near identical content. Note that, many bots/apps post legitimate tweets. For example, by linking Twitter and Foursqaure accounts, users have the option to share check-ins made in Foursqaure to Twitter. Such tweets will not be labeled as spam because these tweets are informative (*i.e.,* reporting location information of users). On the other hand, recall that in Section 4.2.1, we report that the largest cluster contains 127,559 near-duplicate tweets in the form of "RT @username: I've collected 103,501 gold coins! LINK1 #android #androidgames #gameinsight". The only difference between any two tweets is the number of gold coins and the links. Moreover, we observe that many users post only tweets in this form in his/her full user profile in Twitter. We consider such tweets less informative and label them as spam.

With the above criteria, we *selectively* label the near-duplicate clusters, because labeling more than 40 thousand near-duplicate clusters is infeasible. The labeling is conducted in two phases. In the first phase, the top-1000 largest clusters ranked by number of tweets in each cluster are labeled. Because of the power-law distribution (see Figure 3), the labeling of the 1000 largest clusters cover a large number of tweets. In the second phase, we aim to cover more diverse tweets by considering the keywords we used in Section 4.1. For each of the keywords used for tweet selection, we label around 10 largest clusters containing this keyword if the keyword is not covered in the top-1000 clusters labeled earlier. As examples, most clusters containing keywords "love" and "news" are ham clusters while clusters containing "gameinsight" and "iphonegame" are mostly spam. In the labeling process, we also encounter some cases where we cannot confidently label the tweets to be ham or spam even after manual inspection. In this case, we assign an "unknown" label to this cluster.

Result and Analysis. After the two phase labeling, 1.82 million tweets in 2,104 near-duplicate clusters are manually labeled, reported in Table 4. Observe that, although the number of spam clusters is similar as the number of ham clusters, the number of spam tweets contained in these clusters is significantly larger than the number of ham tweets. This is expected as spam tweets are mostly posted repeatedly. More specifically, the median number of tweets in the labeled spam clusters is 437 while the median number of tweets in ham clusters is 26. The median number of users in the two sets of clusters are 383 and 25 respectively. It is also observed that 84% of labeled spam clusters have at least one user posting the same tweet multiple times. On the other hand, only 25% of labeled ham clusters have at least one user posting the same tweet multiple times. More interestingly, 47,454 users who posted at least one spam tweet have 0 followers. Without any followers, these users do not pollute the timelines of other users. However, their tweets may have adverse effect on keyword-based or hashtag-based tweet search. The number of users who posted at least one ham tweet and have 0 followers is 4,822, about 10% of those who posted at least one spam tweet.

4.2.3 kNN Annotation

Using the 2,104 manually labeled clusters as ground truth, we are able to "grow" labels in a more effective manner with the help of kNN classifier. A similar approach has been adopted in creating benchmark dataset for a web image dataset [8].

The most important issue here is to ensure the quality of labels predicted by the kNN classifier. We address this issue by assigning a label to a cluster only if 80% of its k-nearest neighbors are from the same class (*i.e.,* spam or ham). Moreover, if this newly labeled cluster c_k causes misclassification of a manually labeled cluster c_m if the same kNN rule were applied to c_m, then we consider this newly labeled cluster c_k to be a *difficult cluster*. The *difficult clusters* are then manually inspected and labeled. If a newly labeled cluster does not cause misclassification of any existing manually labeled cluster, then the label of this cluster assigned by kNN is considered as a *confident* label. The labeling process is conducted in batch mode and thus confidently labeled clusters in each batch are added as labeled examples for the next batch.

Algorithm 1 summarizes the main steps in kNN-based near duplicate cluster labeling. The algorithm takes the set of manually labeled cluster C_ℓ and unlabeled clusters C_u as input, and returns clusters with confident labels C_k as output, in four main steps.

Step 1. A subset of clusters $C_{mk} \subset C_\ell$ is obtained such that within C_{mk} each cluster's label is consistent with the label predicted based on the kNN rule (Lines 2 - 5). More specifically, for each cluster c, its k nearest neighbors are obtained from the labeled set C_ℓ using function $C_{NN} = kNN(c, k, C_\ell)$, where C_{NN} denotes the set of c's nearest neighbors, $kNN(c, k, C_\ell)$ finds the k nearest neighbors of c within the set C_ℓ. In Line 4, the function $majorityLabel(C_{NN})$ returns the *number of neighbors* from the majority class in C_{NN} (*e.g.,* either spam or ham). The clusters in C_{mk} will be used to identify the difficult clusters labeled by kNN in later steps.

Step 2. The algorithm classifies unlabeled clusters in a batch mode (Lines 10 - 13). Each batch is set to be 20% of the size of the manually labeled examples (Line 10). For the first batch, the set of training examples is initialized to be the set of manually labeled clusters C_ℓ (Line 7). In the subsequent batches, the training examples will include the newly labeled examples (Lines 22 - 23).

Step 3. The algorithm verifies whether the newly labeled examples cause misclassification of some clusters in C_{mk} (Lines 15 - 21). More specifically, the k nearest neighbors of each cluster in C_{mk} is now searched from the set of training examples C_t and the newly labeled batch C_b using the function $kNN(c, k, C_t \cup C_b)$. If the number of neighbors from majority class is smaller than 80% of k, then we consider that the neighbors which are from the newly labeled batch cause the misclassification. These clusters will then

Algorithm 1: kNN Annotation

```
    Input  : Set of manually labeled clusters $C_\ell$
             Set unlabeled near-duplicate clusters $C_u$
             Number of nearest neighbors $k$
    Output: Set of labeled near-duplicate clusters $C_k$
 1  $C_{mk} = \{\}$ // clusters consistent with $k$NN result
 2  foreach cluster $c \in C_\ell$ do
 3      $C_{NN} = kNN(c, k, C_\ell)$ // find nearest neighbors
 4      if $majorityLabel(C_{NN}) \geq 0.8 \times k$ then
 5          $C_{mk} = C_{mk} \cup \{c\}$
 6  $C_d = \{\}$
 7  $C_t = C_\ell$ // set of training examples
 8  $C_b = \{\}$ // a batch of labeled clusters
 9  for unlabeled cluster $c \in C_u$ do
10      if $|C_b| < 0.2 \times |C_\ell|$ then
11          $C_{NN} = kNN(c, k, C_t)$
12          if $majorityLabel(C_{NN}) \geq 0.8 \times k$ then
13              $C_b = C_b \cup \{c\}$
14      else // verify the current batch
15          foreach cluster $c \in C_{mk}$ do
16              $C_{NN} = kNN(c, k, C_t \cup C_b)$
17              if $majorityLabel(C_{NN}) < 0.8 \times k$ then
18                  foreach neighbor cluster $c_n \in C_{NN}$ do
19                      if $c_n \in C_b$ then
20                          $C_b = C_b - \{c_n\}$
21                          $C_d = C_d \cup \{c_n\}$
22          $C_t = C_t \cup C_b$ // add as training examples
23          $C_b = \{\}$// start a new batch
24  Manually label clusters in $C_d$ and add to $C_t$ in Line 7
25  Repeat the labeling process (Lines 9 - 24) till no more clusters
    can be labeled
26  Return $C_k = C_t - C_\ell$;
```

be removed from the batch and considered as *difficult clusters*. The difficult clusters are then manually labeled in the next step. The remaining confidently labeled clusters in C_b will be added to C_t for the next batch labeling (Line 22).

Step 4. Observe that the predicted labels of unlabeled clusters in C_u are dependent on the order of labeling because the training examples C_t are updated after each batch of labeling. For this reason, we run the labeling process (Lines 9 - 23) multiple times. Before each run, we manually label the difficult clusters resulted from the last run (Line 24) and add them to C_t in Line 7 because these labels are manually assigned. Note that, for concise presentation, we do not explicitly add this "outer loop" in Algorithm 1. Instead, this step is stated in Line 25 and the iteration continues until there is no unlabeled cluster which has 80% of neighbors from same class. Note that some detailed control structures in the algorithm are ignored, for example, the last batch in each run may not reach the size of $0.2 \times |C_\ell|$.

For nearest neighbor search, the similarity between two clusters is the cosine similarity with *TFIDF* weighting of the word feature vectors for the two clusters, by aggregating all the tweets in each cluster. The value of k is set to 19 empirically in our kNN based annotation process. The size of each batch is about 420.

Annotation Result. As the result of kNN-based annotation, 15,571 spam clusters and 15,913 ham clusters are labeled. In the labeling process, 1,178 clusters were identified to be difficult clusters and were then manually labeled. Among the latter, 826 clusters were labeled ham and 275 were labeled spam clusters; the remaining 77 clusters were difficult to label even after manual inspection. These 77 clusters are labeled "unknown" in our data collection.

After manual and kNN-based annotation, we have labeled more spam tweets than ham tweets. Because spam clusters often contain much repeated tweets, the number of spam tweets is significantly larger than the number of ham tweets. The reason is that, the manual annotation process was targeted on spam tweets by using the most suspicious keywords and most popular hashtags. The kNN-based annotation was guided by the then labeled clusters. In other words, most ham tweets are left untouched in the dataset.

4.2.4 User- and Domain-based Cluster Annotation

We now label repeated tweets posted by spammers and tweets promoting specific links. Instead of restricting to the tweets that match keywords (Section 4.1), we consider all tweets in this step.

User-based Cluster Annotation. In this step, tweets posted by the same user are grouped into near-duplicate clusters. Then the clusters with more than 10 tweets are labeled. The top 10,000 users who posted the largest number of tweets in our dataset are considered. We observe that most user-based clusters are small clusters with fewer than 10 tweets. That is, most of users do not post many similar tweets. As the result, 980 user-based clusters each has more than 10 tweets need to be labeled, consisting of about 25 thousand tweets in total. Among these 980 clusters, 723 are labeled spam (containing about 20 thousand tweets) and 120 clusters are labeled ham (about 2 thousand tweets). Note that, highly similar ham tweets may be posted by the same user repeatedly (*e.g.,* updates of bus services and weather reports). The remaining 137 are labeled unknown, after looking into the content of the tweets and user profiles (*e.g.,* number of followers/followees, number of tweets) .

Because we consider all tweets in this step, some of the tweets in user-based cluster may have already have been labeled in the earlier steps. If the labels are all from manual annotation, then the cluster is considered labeled. If the labels are given in the kNN annotation, these labels are considered as reference and may be corrected during this manual annotation process.

Domain-based Cluster Annotation. Recall that a tweet may embed links to external sources. Here we aim to label the spam tweets that promote links from specific domains. To this end, the tweets that contain links from any of the top 10,000 most frequent domains in our collection are selected and then grouped into near-duplicate clusters based on domains (all the tweets in the same cluster contain links from the same domain). There are 9,038 such clusters each contains at least 10 tweets. Out of these 9,038 clusters, 4,398 are spam clusters (containing about 221 thousand tweets), and 3,785 are ham clusters (about 121 thousand tweets). The remaining 855 clusters could not be labeled even after manual inspection. Similar to that in user-based cluster annotation, the labels from earlier steps are utilized in this annotation process.

During the annotation process, we observe that domain-based cluster annotation is effective in capturing distributed spam activities in promoting online resources. Examples include links related to topics of working from home, best deals, and coupon codes. Note that, in our labeling process, clusters with tweet content that is clearly ham and links that are from major trustworthy sites (*e.g.,* BBC, CNN, NBA) are labeled ham without looking the content of the web pages. Clusters containing links from less popular domains are labeled by looking into the user profiles and the content of the linked pages. However, it is found that in most cases, tweets provide good descriptions of the links. It is uncommon to have similar tweets linking to very different web pages.

Until this step most ham tweets are left untouched in the dataset. In the next section, we aim to detect the ham tweets from the remaining unlabeled data.

4.3 Reliable Ham Tweet Detection

The remaining unlabeled tweets in our tweet collection may contain a large number of ham tweets, but also contain many spam tweets yet to be identified. Given the large number of remaining unlabeled tweets, this problem can be formulated as a *PU Learning problem*, *i.e.,* learning from Positive and Unlabeled examples [25].

Inspired by the approach for building text classifiers from positive and unlabeled examples proposed in [25], we utilized the *positive words* in spam tweets to identify reliable ham tweets. Positive words are the words that appear more frequently in spam tweets than other tweets. Example positive words are ‘followers’, ‘gain’, ‘gameinsight’, ‘please’, ‘win’, ‘iphone’ etc. More specifically, let $rdf(w, D_s) = \frac{df(w,D_s)}{|D_s|}$ be the relative document frequency of word w in the set of spam tweets D_s labeled so far, where $df(w, D_s)$ denotes the number of tweets containing word w in set D_s. Let $rdf(w, D_h \cup D_u)$ be the relative document frequency of word w in the set of labeled ham tweets D_h and the set of unlabeled tweets D_u. Then w is a positive word if $rdf(w, D_s) > rdf(w, D_h \cup D_u)$. Reliable ham tweets are those tweets that do not contain any positive words. In our implementation, words that are shorter than 3 characters in length are ignored. Note that, number of characters in a hashtag does not include the ‘#’ symbol.

Result and Evaluation. This method is simple but found to be very effective in identifying ham tweets. In total 4.093 million tweets are identified as reliable ham tweets. We evaluated the quality of the labeled tweets by manually inspecting a randomly selected sample of 1000 reliable ham tweets. Among them, 32 tweets are incorrectly labeled as ham, hence the precision of the labels in reliable ham detection is 0.968. Following are three examples of reliable ham tweets detected in this step.

- Why can’t I sleep
- The first thunder sound scared me
- @username hey :)) do u know what channel is CNN on???? Lol i forgot it...

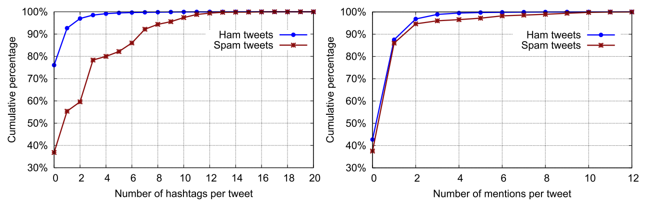
4.4 Annotation by EM-Algorithm

Out of the 14 million tweets in our data collection, more than 7 million tweets are now labeled in the earlier steps. With the 2.387 million spam tweets and 4.897 million ham tweets, we have a sizable collection of labeled examples to build a reliable classifier for the prediction of labels of the remaining unlabeled tweets. To this end, we adopt the Expectation-Maximization (EM) algorithm proposed for text classification using labeled and unlabeled documents [30]. EM has also been used to reduce human effort in evaluating IR systems [11]. A Naïve Bayes classifier is firstly constructed by utilizing the existing labeled tweets. In the E-Step, this classifier is used to predict the labels of the remaining unlabeled tweets. In the M-Step, the probabilities of the word features are re-estimated based on the predicted labels. Then the classifier in the E-Step is updated accordingly and is used to re-classify the tweets in the M-Step. This process continues till the number of changes in the predicted labels is below 0.01% of the total unlabeled tweets.

As the result, 5.790 million ham tweets and 0.951 million spam tweets are labeled in this step. To evaluate the quality of the labeling, we randomly selected 500 ham and 500 spam tweets labeled by EM for manual inspection. The precisions for spam and ham labels are 0.94 and 0.96 respectively.

Table 5: Statistics of the labeled tweets in HSpam14 dataset

Annotation step	Spam (million)	Ham (million)
Manual annotation	1.644	0.226
kNN-based annotation	0.502	0.455
User-based annotation	0.020	0.002
Domain-based annotation	0.221	0.121
Reliable ham tweet detection	-	4.093
EM-based annotation	0.951	5.790
Total	3.338	10.688



(a) Hashtags per tweet (b) Mentions per tweet

Figure 4: Cumulative percentage of hashtags and mentions per tweet, in spam tweets and ham tweets, respectively.

4.5 Summary

After all these steps, we have labeled almost all the 14 million English tweets. Table 5 reports the number of tweets labeled as ham and spam in each step. Note that, the numbers reported under manual annotation are slightly larger than that reported in Table 4 because the numbers here include the manual annotations conducted in kNN-based annotation step as well as the annotations made during the evaluation of subsequent steps. In summary, 3.338 million spam tweets and 10.688 million ham tweets are labeled in the HSpam14 dataset. There are two possible reasons for the relatively large number of spam tweets. First, the tweets in our collection were collected based on trending hashtags, which are more likely to be hijacked for spamming purpose. Second, many of the spam tweets are repetition of similar content, *e.g.,* 127,559 near-duplicate tweets are due to the hashtag #gameinsight.

5. ANALYSIS OF HSPAM14

We believe the quality of the labels are reasonably high based on the manual inspection after each step, if no manual annotation is involved during that step. In this section, we provide a preliminary analysis of the HSpam14 dataset. We mainly focus on the usage of hashtags in spam and ham tweets. Because of the way the tweets were collected, our collection does not contain full user profiles, which limit the user-based analysis.

Per-tweet Hashtag/Mention. Figures 4(a) and 4(b) plot the per-tweet usage distributions of hashtags and mentions in spam and ham tweets respectively. Observe that the mention distributions of spam and ham tweets are very similar to each other with ham tweets use slightly fewer number of mentions. For instance, 43% of ham tweets and 38% of spam tweets do not use mention. The hashtag per-tweet distributions are much more interesting. For ham tweets, 76% of tweets have no hashtags, and 97% of tweets have at most 2 hashtags. However, for spam tweets only 37% of them have no hashtags, and the cumulative percentage of tweets having 2 or fewer hashtags is 60%. The remaining 40% of tweets have 3 or

Table 6: The hashtags with the highest and lowest spammy-index. All hashtags listed in these two tables have frequency larger than 5000, and the hashtags are case-sensitive.

(a) Top-20 hashtags with highest spammy-index

Hashtag	$df(t, D_s)$	$df(t, D)$	$si(t)$
TEAMFOLLOWBACK	661,058	666,328	19.18
TFBJP	526,709	527,176	18.99
gameinsight	509,760	510,504	18.93
OPENFOLLOW	331,643	332,857	18.27
androidgames	286,101	286,706	18.09
android	335,699	341,240	18.06
IPADGAMES	231,725	232,335	17.78
SougoFollow	197,421	197,525	17.58
FOLLOWBACK	176,269	177,109	17.34
ipad	192,706	195,375	17.32
THF	165,407	165,762	17.30
FOLLOWNGAIN	149,708	149,916	17.17
500aday	145,696	146,005	17.12
RETWEET	240,128	250,992	17.10
TEAMHITFOLLOW	136,039	136,043	17.05
AUTOFOLLOW	140,686	141,214	17.04
MUSTFOLLOW	137,502	138,040	17.00
RT	224,640	235,137	16.98
HITFOLLOWSTEAM	127,262	127,264	16.96
Follow	122,575	126,720	16.35

(b) Top-20 hashtags with lowest spammy index

Hashtag	$df(t, D_s)$	$df(t, D)$	$si(t)$
musicjournals	4	6408	0.00
QnA	32	11868	0.01
occupygezi	21	13521	0.01
iCanAdmit	12	6714	0.01
thankyousiralex	25	5788	0.02
Aquarius	43	7270	0.03
blessed	42	6206	0.04
job	80	9786	0.05
BELIEVEtour	206	22758	0.07
Pisces	82	8011	0.07
nbaplayoffs	66	5596	0.07
Capricorn	104	7636	0.09
CFC	125	9271	0.09
jobs	333	26402	0.11
Sagittarius	202	9706	0.16
tbt	382	18157	0.18
TCOT	207	8553	0.19
MUFC	410	17791	0.20
Gemini	283	11156	0.21
Libra	160	5425	0.22

more hashtags. That is, the spam tweets use hashtags much more aggressively than ham tweets, probably aiming for more visibility.

Hashtag Spammy Index. We propose a measure named *Spammy Index* to quantify the extent a hashtag is being used in spam tweets. Given a hashtag t, its spammy index, denoted by $si(t)$, is defined in the following equation.

$$si(t) = \log_2(df(t, D)) \times \frac{df(t, D_s)}{df(t, D)} \qquad (1)$$

In this equation, $df(t, D)$ denotes document frequency, or the number of tweets containing hashtag t in tweets collection D; D_s is the set of spam tweets where $D_s \subset D$. A hashtag is considered to be more spammy if (i) its document frequency is high, and (ii) the probability of being used in spam tweets is high.

The top-20 hashtags with the highest spammy indexes are reported in Table 6(a). For comparison, Table 6(b) lists the 20 hashtags with the lowest spammy indexes among all hashtags with frequency more than 5,000. The two tables show that spammy hashtags are more likely to use capital letters, probably to draw attention. Frequent words appear in spammy hashtags are "follow", "fb"(follow back) and "game". Hashtags with low spammy indexes are less likely to contain all capital letters and many of them are topic specific on music, sports, or functional tags like #job and #jobs. Comparing Table 6(a) and Table 1, most extremely popular hashtags are hijacked for spamming purpose. Popular hashtags such as #android, #ipad are hijacked for spamming purpose so they appeared as top 20 spammy hashtags. The popularity of these hashtags may attribute to the large number of near-duplicate tweets automatically generated by bot like #gameinsight.

For the completeness of the analysis, we also computed the Odds Ratio (OR) of the words (including both case-insensitive hashtags after removing the # symbol and words appear in the content of tweets) in distinguishing spam tweets from ham tweets. The top-20 words with highest OR values are reported in Table 7. The words identified by high OR values are consistent with the hashtags of high spammy indexes, with the dominating word "follow".

Hashtag Co-occurrence Network. The co-occurrence graph for popular hashtags with frequency larger than 25,000 in HSpam14 is plotted in Figure 5. The size of node is proportional to the hashtag frequency (in $\log_2$ scale). The color of node is proportional to the hashtag's spammy index (most spammy hashtags are in red, least spammy hashtags are in green, intermediate ones are in blue). We make two observations from this co-occurrence network. First, the spammy hashtags are more likely to co-occur with each other. Second, the less spammy hashtags are less likely to co-occur with other hashtags. This can be partially explained by the observation that 97% of ham tweets have at most 2 hashtags.

Table 7: Top-20 most spammy words by Odds Ratio

Rank	Word	Rank	Word
1	hitfollowsteam	11	followers
2	tribez	12	ifollowngain
3	maxvip	13	followbacksegur
4	r_family	14	teamfollowwacky
5	follow2befollowe	15	smurfberries
6	teamhitfollow	16	myretweets
7	extraretweets	17	reignd
8	onlyifyouare	18	mesiguestesigo
9	rt2club	19	90sbabyf
10	rt2supergain	20	harvested

6. CONCLUSION AND LIMITATION

In this paper, we make the first step for tweet-level spam detection focusing on hashtag-oriented tweet spam. The main contribution of this work is the creation of HSpam14 data collection, which contains 3.338 million spam tweets and 10.688 million ham tweets, collected in two months' time guided by trending hashtags. While the annotation was not made per tweet basis, the overall quality of the labels are good enough to perform analyses and studies on hashtag-oriented tweet spam detection. Our analysis on this dataset also shows different usage patterns of mentions and hashtags in spam and ham tweets, and shows that spammy hashtags are likely to co-occur with spammy hashtags.

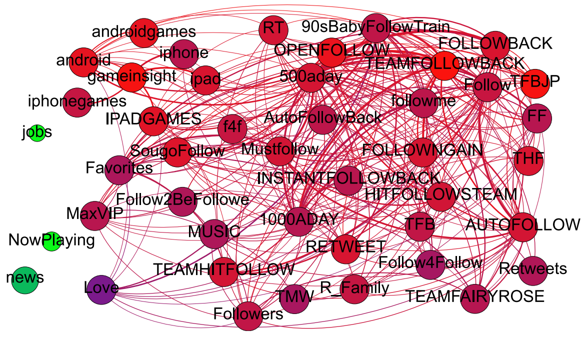


Figure 5: Co-occurrence of the most popular hashtags.

A limitation of this data collection is the lack of full profiles of the users. As the tweets in this collection were collected with the guide of trending hashtags, we do not have full user profiles (*e.g.,* all tweets published by each user, list of her followers/followees) in HSpam14 due to the restriction of Twitter steaming APIs. This may limit the study on user-level spam detection with this dataset. For the same reason, we only utilize very limited user-level features during the annotation process although many other user-level features (*e.g.,* temporal patterns, social connections) have demonstrated their effectiveness in identifying spammers. Another limitation could be the limited keywords used in heuristic-based tweet selection. The existing keywords may not catch spam in specific domains (*e.g.,* healthcare and medical). Part of our future work is to improve the recall of the spam tweets in specific domains.

Acknowledgements: This work was partially supported by Singapore MOE AcRF Tier-1 Grant RG142/14.

7. REFERENCES

[1] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida. Detecting spammers on twitter. In *CEAS*, 2010.

[2] F. Benevenuto, T. Rodrigues, V. Almeida, J. Almeida, and M. Gonçalves. Detecting spammers and content promoters in online video social networks. In *SIGIR*, pages 620–627, 2009.

[3] A. Broder. On the resemblance and containment of documents. In *Proc. Compression and Complexity of Sequences*, pages 21–29, 1997.

[4] C. Castillo, M. Mendoza, and B. Poblete. Information credibility on twitter. In *WWW*, pages 675–684, 2011.

[5] D. Chinavle, P. Kolari, T. Oates, and T. Finin. Ensembles in adversarial classification for spam. In *CIKM*, pages 2015–2018, 2009.

[6] A. Chowdhury, O. Frieder, D. Grossman, and M. C. McCabe. Collection statistics for fast duplicate document detection. *ACM Trans. on Info. Sys.*, pages 171–191, 2002.

[7] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia. Who is tweeting on twitter: Human, bot, or cyborg? In *Proc. Annual Computer Security Appln Conf.*, pages 21–30, 2010.

[8] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng. Nus-wide: A real-world web image database from national university of singapore. In *CIVR*, pages 48:1–48:9, 2009.

[9] G. V. Cormack. Email spam filtering: A systematic review. *Foundations and Trends in Inf. Ret.*, pages 335–455, 2008.

[10] E. Ferrara, O. Varol, C. Davis, F. Menczer, and A. Flammini. The rise of social bots, 2014. arXiv:1407.5225.

[11] N. Gao, W. Webber, and D. W. Oard. Reducing reliance on relevance judgments for system comparison by using expectation-maximization. In *ECIR*, pages 1–12, 2014.

[12] S. Ghosh, B. Viswanath, F. Kooti, N. K. Sharma, G. Korlam, F. Benevenuto, N. Ganguly, and K. P. Gummadi. Understanding and combating link farming in the twitter social network. In *WWW*, pages 61–70, 2012.

[13] C. Grier, K. Thomas, V. Paxson, and M. Zhang. @spam: The underground on 140 characters or less. In *Proc. ACM Conf. on Computer and Communications Security*, pages 27–37, 2010.

[14] P. Heymann, G. Koutrika, and H. Garcia-Molina. Fighting spam on social web sites: A survey of approaches and future challenges. *IEEE Internet Computing*, 11:36–45, 2007.

[15] X. Hu, J. Tang, and H. Liu. Leveraging knowledge across media for spammer detection in microblogging. In *SIGIR*, pages 547–556, 2014.

[16] X. Hu, J. Tang, and H. Liu. Online social spammer detection. In *AAAI*, pages 59–65, 2014.

[17] N. Jindal and B. Liu. Review spam detection. In *WWW*, pages 1189–1190, 2007.

[18] N. Jindal and B. Liu. Opinion spam and analysis. In *WSDM*, pages 219–230, 2008.

[19] K. Lee, J. Caverlee, and S. Webb. The social honeypot project: Protecting online communities from spammers. In *WWW*, pages 1139–1140, 2010.

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

[21] K. Lee, S. Webb, and H. Ge. The dark side of micro-task marketplaces: Characterizing fiverr and automatically detecting crowdturfing. In *ICWSM*, 2014.

[22] F. Li, M. Huang, Y. Yang, and X. Zhu. Learning to identify review spam. In *IJCAI*, pages 2488–2493, 2011.

[23] J. Li, C. Cardie, and S. Li. Topicspam: a topic-model based approach for spam detection. In *ACL*, pages 217–221, 2013.

[24] E.-P. Lim, V.-A. Nguyen, N. Jindal, B. Liu, and H. W. Lauw. Detecting product review spammers using rating behaviors. In *CIKM*, pages 939–948, 2010.

[25] B. Liu, Y. Dai, X. Li, W. S. Lee, and P. S. Yu. Building text classifiers using positive and unlabeled examples. In *ICDM*, pages 179–186, 2003.

[26] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*, volume 1. Cambridge University Press, 2008.

[27] J. Messias, L. Schmidt, R. Oliveira, and F. Benevenuto. You followed my bot! transforming robots into influential users in twitter. *First Monday*, 18, 2013.

[28] G. Mishne, D. Carmel, and R. Lempel. Blocking blog spam with language model disagreement. In *AIRWeb*, 2005.

[29] A. Mukherjee, B. Liu, J. Wang, N. Glance, and N. Jindal. Detecting group review spam. In *WWW*, pages 93–94, 2011.

[30] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Mach. Learn.*, pages 103–134, 2000.

[31] M. Ott, Y. Choi, C. Cardie, and J. T. Hancock. Finding deceptive opinion spam by any stretch of the imagination. In *HLT*, pages 309–319, 2011.

[32] A. Shrivastava and P. Li. In defense of minhash over simhash. In *AISTATS*, pages 886–894, 2014.

[33] N. Spirin and J. Han. Survey on web spam detection: Principles and algorithms. *SIGKDD Explor. Newsl.*, 13(2):50–64, 2012.

[34] E. Tan, L. Guo, S. Chen, X. Zhang, and Y. Zhao. Unik: Unsupervised social network spam detection. In *CIKM*, pages 479–488, 2013.

[35] K. Thomas, C. Grier, D. Song, and V. Paxson. Suspended accounts in retrospect: An analysis of twitter spam. In *IMC*, pages 243–258, 2011.

[36] C. Wagner, S. Mitter, C. Körner, and M. Strohmaier. When social bots attack: Modeling susceptibility of users in online social networks. In *Workshop on Making Sense of Microposts*, 2012.

[37] G. Wang, C. Wilson, X. Zhao, Y. Zhu, M. Mohanlal, H. Zheng, and B. Y. Zhao. Serf and turf: Crowdturfing for fun and profit. In *WWW*, pages 679–688, 2012.

[38] Z. Wang, W. Josephson, Q. Lv, M. Charikar, and K. Li. Filtering image spam with near-duplicate detection. In *Proc. Conf. on Email and AntiSpam*, pages 600–603, 2007.

[39] J. Weng, E.-P. Lim, J. Jiang, and Q. He. Twitterrank: Finding topic-sensitive influential twitterers. In *WSDM*, pages 261–270, 2010.

[40] L. Zhu, A. Sun, and B. Choi. Detecting spam blogs from blog search results. *Inf. Process. Manage.*, 47(2):246–262, 2011.