Statistical Features-Based Real-Time Detection of Drifted Twitter Spam

Chao Chen, Yu Wang, Jun Zhang, Yang Xiang, Wanlei Zhou, *Senior Member, IEEE*, and Geyong Min

***Abstract*— Twitter spam has become a critical problem nowadays. Recent works focus on applying machine learning techniques for Twitter spam detection, which make use of the statistical features of tweets. In our labeled tweets data set, however, we observe that the statistical properties of spam tweets vary over time, and thus, the performance of existing machine learning-based classifiers decreases. This issue is referred to as "Twitter Spam Drift". In order to tackle this problem, we first carry out a deep analysis on the statistical features of one million spam tweets and one million non-spam tweets, and then propose a novel *Lfun* scheme. The proposed scheme can discover "changed" spam tweets from unlabeled tweets and incorporate them into classifier's training process. A number of experiments are performed to evaluate the proposed scheme. The results show that our proposed Lfun scheme can significantly improve the spam detection accuracy in real-world scenarios.**

***Index Terms*— Social network security, twitter spam detection, machine learning.**

I. Introduction

TWITTER has become one of the most popular social networks in the last decade. It is rated as the most popular social network among teenagers according to a recent report [17]. However, the exponential growth of Twitter also contributes to the increase of spamming activities. Twitter spam, which is referred to as unsolicited tweets containing malicious link that directs victims to external sites containing malware downloads, phishing, drug sales, or scams, *etc.* [1], not only interferes user experiences, but also damages the whole Internet. In September 2014, the Internet of New Zealand was melt down due to the spread of malware downloading spam. This kind of spam lured users to click links which claimed to contain Hollywood star photos, but in fact directed users to download malware to perform DDoS attacks [29].

Consequently, security companies, as well as Twitter itself, are combating spammers to make Twitter as a spam-free platform. For example, Trend Micro uses a blacklisting service called Web Reputation Technology system to filter spam URLs for users who have its products installed [27]. Twitter also implements blacklist filtering as a component in their detection system called BotMaker [21]. However, blacklist fails to protect victims from new spam due to its time lag [18]. Research shows that, more than 90% victims may visit a new spam link before it is blocked by blacklists [37]. In order to address the limitation of blacklists, researchers have proposed some machine learning based schemes which can make use of spammers' or spam tweets' statistical features to detect spam without checking the URLs [14], [40].

Machine Learning (ML) based detection schemes involve several steps. First, statistical features, which can differentiate spam from non-spam, are extracted from tweets or Twitter users (such as account age, number of followers or friends and number of characters in a tweet). Then a small set of samples are labelled with class, *i.e.* spam or non-spam, as training data. After that, machine learning based classifiers are trained by the labelled samples, and finally the trained classifiers can be used to detect spam. A number of ML based detection schemes have been proposed by researchers [1], [35], [39], [43].

However, the observation in our collected data set shows that the characteristics of spam tweets are varying over time. We refer to this issue as "Twitter Spam Drift". As previous ML based classifiers are not updated with the "changed" spam tweets, the performance of such classifiers are dramatically influenced by "Spam Drift" when detecting new coming spam tweets. Why do spam tweets drift over time? It is because that spammers are struggling with security companies and researchers. While researchers are working to detect spam, spammers are also trying to avoid being detected. This leads spammers to evade current detection features through posting more tweets or creating spam with the similar semantic meaning but using different text [34], [39].

In this work, we firstly illustrate the "Twitter spam drift" problem through analysing the statistical properties of Twitter spam in our collected dataset and then its impact on detection performance of several classifiers. By observing that there are "changed" spam samples in the coming tweets, we propose a novel **Lfun** (**L**earning **f**rom **un**labelled tweets) approach, which updates classifiers with the spam samples from the

Manuscript received August 6, 2015; revised February 28, 2016 and August 1, 2016; accepted October 9, 2016. Date of publication October 26, 2016; date of current version January 30, 2017. This work was supported by the ARC Linkage Project under Grant LP120200266. The work of J. Zhang was supported by the National Natural Science Foundation of China under Grant 61401371. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Hafiz Malik. *(Corresponding author: Y. Wang.)*

C. Chen is with the University of Electronic Science and Technology of China, Chengdu 610051, China, and also with the School of Information Technology, Deakin University, Burwood, VIC 3125, Australia (e-mail: chao.chen@deakin.edu.au).

Y. Wang, J. Zhang, Y. Xiang, and W. Zhou are with the School of Information Technology, Deakin University, Burwood, VIC 3125, Australia (e-mail: y.wang@deakin.edu.au; jun.zhang@deakin.edu.au; yang.xiang@deakin.edu.au; wanlei.zhou@deakin.edu.au).

G. Min is with the College of Engineering, Mathematics, and Physical Sciences, University of Exeter, Exeter, EX4 4QF, U.K. (e-mail: g.min@exeter.ac.uk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIFS.2016.2621888

1556-6013 © 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

unlabelled incoming tweets. In summary, our contributions are listed below:

- We collect and label a real-world dataset, which contains 10 consecutive days' tweets with 100k spam tweets and 100k non-spam tweets in each day (2 million tweets in total). This dataset is available for researchers to study Twitter spam.[1]
- We investigate the "Twitter Spam Drift" problem from both data analysis and experimental evaluation aspects. To the best of our knowledge, we are the first to study this problem in Twitter spam detection.
- We propose a novel Lfun approach which learns from unlabelled tweets to deal with "Twitter Spam Drift". Through our evaluations, we show that our proposed Lfun can effectively detect Twitter spam by reducing the impact of "Spam Drift" issue.

The rest of this paper is organized as follows. Section II presents a review on machine learning based methods for Twitter spam detection. In Section III, the collection and labelling of the data used in our work is introduced. Meanwhile, the "Spam Drift" problem is illustrated and justified. Then we introduce our Lfun approach in Section IV, and analyse the performance benefit of our approach. Section V evaluates our Lfun approach and compares it with four traditional machine learning algorithms. Finally, Section VII concludes this work and introduces our future work.

II. Related Work

Due to the increasing popularity of Twitter, spammers have transferred from other platforms, such as email and blog, to Twitter. To make Twitter as a clean social platform, security companies and researchers are working hard to eliminate spam. Security companies, such as Trend Micro [27], mainly rely on blacklists to filter spam links. However, blacklists fail to protect users on time due to the time lag. To avoid the limitation of blacklists, some early works proposed by researchers use heuristic rules to filter Twitter spam. Reference [41] used a simple algorithm to detect spam in #robotpickupline (the hashtag was created by themselves) through these three rules: suspicious URL searching, username pattern matching and keyword detection. Reference [23] simply removed all the tweets which contained more than three hashtags to filter spam in their dataset to eliminate the impact of spam for their research.

Later on, some works applied machine learning algorithms for Twitter spam detection. References [1], [24], [35], [38] made use of account and content based features, such as *account age, the number of followers/followings, the length of tweet, etc.* to distinguish spammers and non-spammers. Wang *et al.* proposed a Bayesian classifier based approach to detect spammers on Twitter [38], while Benevenuto *et al.* detected both spammers and spam by using Support Vector Machine [1]. In [35], Stringhini *et al.* trained a Random Forest classifier, and used the classifier to detect spam from three social networks, Twitter, Facebook and MySpace. Lee *et al.* deployed some honeypots to get spammers' profiles, and extracted the statistical features for spam detection with several ML algorithms, such as Decorate, RandomSubSpace and J48 [24].

Features used in previous works [1], [24], [35], [38] can be fabricated easily through purchasing more followers, posting more tweets, or mixing spam with normal tweets [39]. Thus, some researchers [34], [39] proposed robust features which rely on the social graph to avoid feature fabrication. Song *et al.* extracted the *distance* and *connectivity* between a tweet sender and its receiver to determine whether it was spam or not [34]. After importing their features into previous feature set, the performance of several classifiers were improved to nearly 99% True Positive and less than 1% False Positive. While in [39], Yang *et al.* proposed more robust features, such as *Local Clustering Coefficient, Betweenness Centrality* and *Bidirectional Links Ratio*. By comparing with four existing works [1], [24], [35], [38], their feature set can outperform all the previous works.

Instead, [25] and [36] solely relied on the embedded URLs in tweets to detect spam. A number of URL based features were used by [36], such as the *domain tokens, path tokens and query parameters of the URL*, along with some features from the landing page, DNS information, and domain information. In [25], the authors studied the characteristics of Correlated URL Redirect Chains, and further collected relevant features, like *URL redirect chain length, Relative number of different initial URLs etc.* These features also showed their discriminative power when used for classifying spam.

However, all the above mentioned works do not consider the "Spam Drift" problem. Their detection accuracy will decrease as time goes on, since spammers are changing strategies to avoid being detected. Egele *et al.* proposed a historical model based spam detection scheme, whose detection accuracy would not be affected by "Spam Drift" [12]. They built several models, like Language model and Posting Time model, for each user. Once the model behaved abnormally, there might be a compromise of this account, and this account was likely to be used to spread spam by attackers. This method can only detect whether an account was compromised or not, but cannot identify the spamming accounts which were created by spammers fraudulently.

Different to related works, we are going to thoroughly study the "Spam Drift" problem. In addition, we will propose an innovative scheme called Lfun which learns from the unlabelled tweets and can tackle this issue in identifying Twitter spam. Thus, our work can make great contributions to the research area of Twitter spam detection.

III. Problem of Twitter Spam Drift

A. 10-day groundtruth

A labelled dataset is important for classification tasks, such as Twitter spam detection. In this work, we used Twitter's Streaming API to collect tweets with URLs in a period of 10 consecutive days. While it is possible to send spam without embedding URLs on Twitter, the majority of spam contains URLs [7], [12], [16]. We have inspected hundreds of spam tweets by hand and only find a few tweets without URLs which could be considered as spam. In addition, spammers

[1] You can download our dataset from http://nsclab.org/nsclab/resources/

TABLE I

EXTRACTED FEATURES

Feature No.	Feature Name	Description
f1	account_age	The age (days) of an account since its creation until the time of sending the most recent tweet
f2	no_follower	The number of followers of this twitter user
f3	no_following	The number of followings/friends of this twitter user
f4	no_userfavourites	The number of favourites this twitter user received
f5	no_lists	The number of lists this twitter user added
f6	no_tweets	The number of tweets this twitter user sent
f7	no_retweets	The number of retweets this tweet
f8	no_hashtag	The number of hashtags included in this tweet
f9	no_usermention	The number of user mentions included in this tweet
f10	no_urls	The number of URLs included in this tweet
f11	no_char	The number of characters in this tweet
f12	no_digits	The number of digits in this tweet

mainly use embedded URLs to make it more convenient to direct victims to external sites to achieve their goals, such as phishing, scams, and malware downloading [43]. Therefore, we only focus on spam tweets with URLs.

Currently, researchers use two ways to build ground-truth, manual inspection and blacklists filtering. While manual inspection can label a small number of training data, it is very time- and resource-consuming. A large group of people are needed to check tens of thousands of tweets. Although HIT (human intelligence task) websites can help label the tweets, it is also costly and sometimes the results are doubtful [3]. Others apply existing blacklisting services, such as Google SafeBrowsing and URIBL [15] to label spam tweets. Nevertheless, these services' API limits make it impossible to label a large amount of tweets.

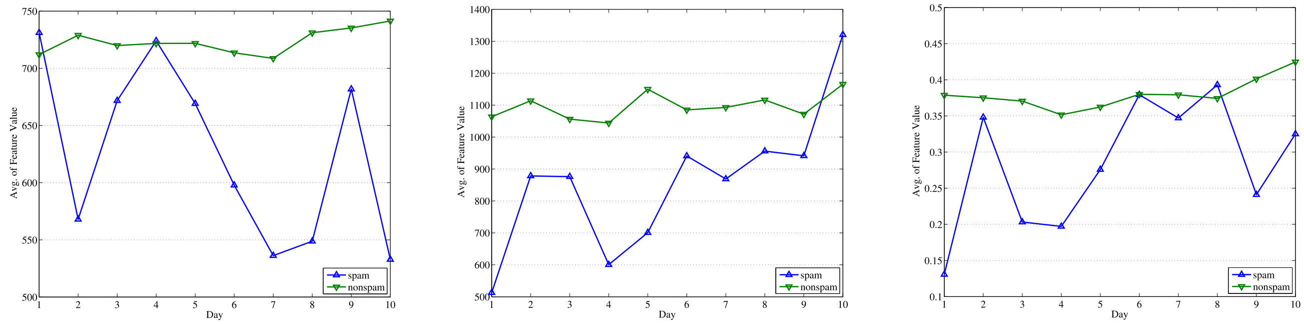
We apply Trend Micro's Web Reputation Technology to identify which tweets are deemed spam [4]. Trend Micro's WRT system maintains a large dataset of URL reputation records, which are derived from their customers' opt-in URL filtering records. WRT system is dedicated to collecting the latest and the most popular URLs, to analysing them, and then to providing Trend Micro customers with real-time protection while they are surfing the web. Hence, through checking URLs with the WRT system, we are able to identify whether a URL is malicious or not. We define those which contain malicious URLs as Twitter spam. WRT system is reliable as the protection rate of it is 100%, as stated in AV Comparatives' testing report [8]. In addition, we have done a manual inspection of hundreds of tweets to confirm the reliability of WRT. In our collected data, we labelled one million spam tweets and one million non-spam tweets for 10 days, with 100k spam tweets and 100k non-spam tweets for each day.

Feature extraction is a key component in machine learning based classification tasks [42]. Some studies [1], [35], [38] have applied a few features which make use of historical information of a user, such as tweets that the user sent in a period of time. While these features may be more discriminative, it is not possible to collect them due to the restrictions of Twitter's API. Other researchers [34], [39] applied some social graph based features, which are hard to be evaded. Nevertheless, It is significantly expensive to collect those features, as they cannot be calculated until the social graph is formed. Thus, those expensive features are not suitable for real-time detection, despite that they have more discriminative power in separating spammers and legitimate users. The longer time a spam tweet exists, the more chance it can be exposure to victims. Thus, it is very important to detect spam tweets as early as possible. To reduce the loss caused by spam, real-time detection is in demand. Consequently, we only focus on extracting light-weight features which can be used for timely detection as in [19]. These features can be straightforwardly extracted from the collected tweets' JSON data structure [28] with little computation. We have totally extracted 12 features from our dataset as listed in TABLE I.

B. Problem Statement

In the real world, the statistical features of spam tweets are changing in unpredicted ways over time. As a result, machine learning based detection system becomes inaccurate. The issue is referred to as "Spam Drift" problem in our previous paper [5]. Here, we present an investigation of "Spam Drift" problem from the aspect of the change of mean value of each feature from day to day.

Fig. 1 shows the changing trend of average value of each feature for two classes in 10 days. In general, the variation of average value of feature from spam tweets is greater than that of non-spam tweets. Fig. 1a shows that, the average value of Account Age for spam tweets ranges from 530 to 730, and the variation is dramatic. However, it deviates from 710 to 740 for non-spam tweets, which is relatively stable. It is due to the fact that spammers are creating a large number of new accounts to send spam once their old account are blocked. For instance, we have 3 spammers with account age of 2 days, 6 days, 10 days in the first day, the average value of Account Age is (2 +6 +10)/3 = 6 days. In the second day, if the spammer whose account age is 2 days is detected and removed, the average value of Account Age is (6+10)/2 = 8 days, which increases. In addition, spammers may also generate new accounts with 0 day Account Age to spread spam after some of their accounts are block, which can lead the decrease of average value of Account Age. That is why the average value of Account Age is fluctuating. Naturally, spammers tend to keep following new friends as they want to be exposed to public more frequently, whereas for non-spammers, their number of followings are not changing too much once they have built their friend circle, as we can see from Fig. 1c.



(a) (b) (c)

Fig. 1. Changes of average values of features. (a) Account age. (b) No. of followings. (c) No. of user mentions per tweet.

TABLE II

KL DIVERGENCE OF SPAM AND NONSPAM TWEETS OF TWO CONSECUTIVE DAYS

	D1 VS D2		D2 VS D3		D3 VS D4		D4 VS D5		D5 VS D6		D6 VS D7		D7 VS D8		D8 VS D9		D9 VS D10	
f1	0.36	0.04	0.34	0.03	0.44	0.04	0.24	0.03	0.26	0.03	0.27	0.03	0.29	0.05	0.26	0.03	0.34	0.04
f2	0.24	0.1	0.22	0.1	.26	0.1	0.19	0.1	0.21	0.1	0.21	0.1	0.17	0.1	0.38	0.1	0.35	0.1
f3	0.28	0.07	0.22	0.07	0.32	0.07	0.15	0.07	0.22	0.07	0.2	0.07	0.2	0.08	0.26	0.08	0.23	0.08
f4	0.16	0.07	0.13	0.07	0.14	0.08	0.14	0.07	0.17	0.07	0.19	0.07	0.13	0.07	0.27	0.08	0.19	0.08
f5	0.02	0.01	0.02	0.01	0.03	0.01	0.02	0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.05	0.01	0.05	0.01
f6	0.98	0.35	0.52	0.35	0.63	0.35	0.36	0.35	0.45	0.34	0.4	0.34	0.45	0.35	0.5	0.35	0.52	0.36
f7	0.1	0.04	0.08	0.03	0.04	0.04	0.04	0.04	0.05	0.03	0.07	0.04	0.06	0.04	0.1	0.04	0.08	0.04
f8	0.19	0	0	0	0.04	0	0.03	0	0.02	0	0.03	0	0.01	0	0.04	0	0.02	0
f9	0.09	0	0.03	0	0.01	0	0.02	0	0.01	0	0.01	0	0	0	0.04	0	0.01	0
f10	0	0	0.03	0	0.03	0	0.01	0	0.1	0	0	0	0.01	0	0.32	0	0.27	0
f11	0.26	0.01	0.06	0.01	0.06	0.01	0.11	0.01	0.1	0	0.09	0	0.26	0.01	0.28	0.01	0.2	0.02
f12	0.04	0	0	0	0.02	0	0.03	0.01	0.03	0	0.04	0	0.04	0	0.46	0	0.46	0

As expected, most of the other features have the same trend: the average value of one feature varies for spam tweets, while it is stable for non-spam tweets.

To sum up, the characteristics of spam tweets is varying from day to day, while that of non-spam tweets is not changing much, as we see from Fig. 1. "Spam Drift" is a crucial issue in Twitter spam detection, which is in great need to be solved.

C. Problem Justification

In previous section, we simply compare some representative statistics, such as the mean values of features to show the "Spam Drift" problem. To further illustrate the changing of the statistical features in a dataset, a natural approach is to model the distribution of the data [10]. There are two kinds of approaches: parametric and non-parametric. Parametric approaches are very powerful when the specific distribution of the dataset, like Normal Distribution, is already known. However, the distribution of the Twitter spam data is unknown, thus it is not possible to apply parametric approaches. Consequently, non-parametric methods, such as statistical tests, which make no assumptions of the dataset distributions are used by researchers [13].

The statistical tests are to compute the distance of two distributions to determine the change. One of the most common measures to compute the distance of distributions is Kullback-Leibler (KL) Divergence [10], [32]. The suitability of KL Divergence to be used in measuring distributions can be found in [10]. In [26], Juan *et al.* also use KL Divergence to model language models of tweets. KL Divergence, which is also known as relative entropy is defined as

$$D_{kl}(P\|Q) = \sum_{i} P(i)log\frac{P(i)}{Q(i)}.$$

It is used to compare two probability distributions. We need to map data points into distributions to apply the formula. According to [9], let $\mathbf{s} = \{x_1, x_2, \ldots, x_n\}$ be a multi-set from a finite set $\mathbf{F}$ containing numerical feature values, and denote $N(x|\mathbf{s})$ the number of appearances of $x \in \mathbf{s}$, thus the relative proportion of each x is donated by

$$P_{\mathbf{s}}(x) = \frac{N(x|\mathbf{s})}{n}.$$

However, the ratio of p/q is undefined if $Q(i) = 0$. As suggested by [22], the estimate $P_{\mathbf{s}}$ is replaced as,

$$P_{\mathbf{s}}(x) = \frac{N(x|\mathbf{s}) + 0.5}{n + |F|/2}.$$

when $|F|$ is number of elements in the finite set $\mathbf{F}$. The distance between two day's tweets, $D1$ and $D2$ is,

$$D(D1\|D2) = \sum_{x\in\mathbf{F}} P_{\mathbf{D1}}(x)log\frac{P_{\mathbf{D1}}(x)}{P_{\mathbf{D2}}(x)}.$$

We compute the KL Divergence of each feature of spam and non-spam tweets in two adjacent days, which is listed in TABLE II. The shadowed ones are the KL Divergence of features of non-spam tweets, while the others are the KL Divergence of features of spam tweets. KL Divergence

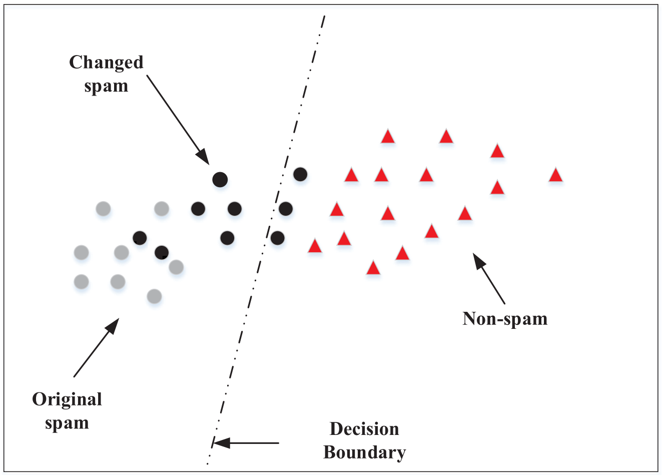


Fig. 2. Illustration of "Spam Drift".

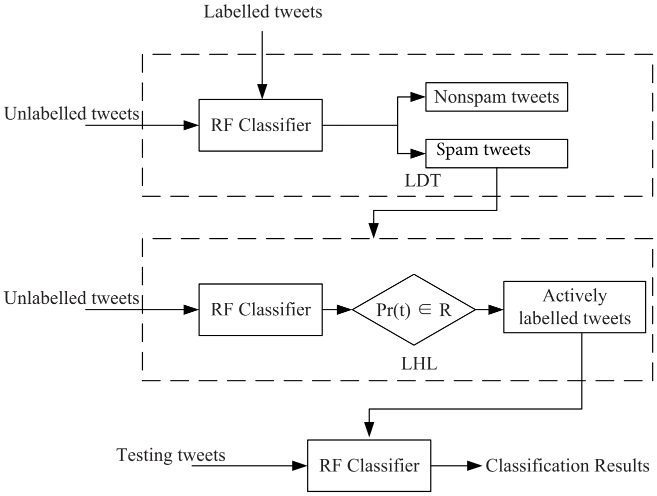


Fig. 3. Lfun framework.

indicates the dissimilarity of two distributions. The larger the value is, the more different the two distributions are. As shown in Table II, the KL Divergence of spam tweets in two adjacent days are much larger than that of the non-spam tweets for more than half the features. Taking f1 ("account_age") for example, the KL Divergence of spam between Day 1 and Day 2 is 0.36, while it is only 0.04 for non-spam, which indicates that the distribution of f1 of spam in Day 1 is much different to it in D2, compared with non-spam tweets' distribution. From these KL Divergence values, we can see that the distribution of spam tweets' features is changing unpredictably from day to day. Nevertheless, the distribution of training data is unchanged. As the knowledge structure which learns from the unchanged training data is not updated while being used to classify new incoming tweets, the performance of classifiers becomes inaccurate. As it is illustrated in Fig. 2, while the spam changes, the decision boundary is not updated. Consequently, more spam tweets are misclassified as non-spam.

IV. Proposed Scheme: *Lfun*

Existing machine learning based spam detection methods suffer from the problem of "Spam Drift" due to the change of statistical features of spam tweets as time goes on. When "spam drifts", the old classification model is not updated with "changed" spam samples, as a result, the classification results will gradually become inaccurate. To solve this problem, obtaining the "changed" samples to update the classification model is very important. By observing that there are such samples in the unlabelled incoming tweets which are very easy to collect, we propose a scheme called "Lfun" to address "Spam Drift" problem.

This section presents our Lfun scheme to deal with the drift problem in Twitter spam detection. Fig. 3 illustrates the framework of our proposed scheme. There are two main components in this framework: LDT is to learn from detected spam tweets and LHL is to learn from human labelling. In "Drifted Spam Detection" scenario, we have already got a small amount of labelled spam and non-spam tweets. However, there are not enough samples of "changed" spam. It is extraordinary expensive to have human label a large amount of "changed" tweets. Consequently, we make use of the above mentioned two components to automatically extract "changed" spam tweets from a set of unlabelled tweets, which are very easy to collected from Twitter. Once getting enough labelled "changed" spam tweets, we implement the scheme which employs a sufficiently powerful algorithm, Random Forest, to perform classification. Our Lfun scheme is summarised in Algorithm 1.

A. Learning from Detected Spam Tweets

LDT is used to deal with a classification scenario where there is a sufficiently robust algorithm, but in lack of more data [30]. By learning from a large number of unlabelled data, LDT can obtain sufficient new information, which can be used to update the classification model.

In a LDT learning scenario, we are given a labelled data set $T_l = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$, containing m labelled tweets, where $x_i \in \mathbb{R}^k (i = 1, 2, \ldots, m)$ is the feature vector of a tweet, $y_i \in \{spam, non-spam\}$ is the category label of a tweet. We are also given a large data set $T_u = \{x_{m+1}, y_{m+1}), (x_{m+2}, y_{m+2}), \ldots, (x_{m+n}, y_{m+n})\}$ containing n unlabelled tweets ($n >> m$). Then a classifier φ is trained by T_l. φ can be used to divide T_u into spam T_{spam} and non-spam $T_{non-spam}$. Labelled spam tweets from T_u will be added into the labelled data set T_l to form a new training data set.

The basic of LDT is to find a function $\varphi : \mathbb{R}^k \longrightarrow \{spam, non-spam\}$ to predict the label $y \in \{spam, non-spam\}$ of new tweets when trained by T_{l+spam}, which is the combination of the labelled data set T_l and spam tweets T_{spam} identified from T_u. Particularly, the unlabelled data set T_u used in LDT does not have to share the same distribution with the labelled data set T_l [20]. In addition, only detected spam tweets will be added into the training data. The reason is that, we've already gained sufficient information of non-spam tweets, as the statistical properties are not changing for non-spam tweets. It is not necessary for us to gain more information about non-spam tweets.

Algorithm 1 Lfun Algorithm

```
Require: labelled training set $\{\psi_1, ..., \psi_N\}$,
    unlabelled tweets $T_{unlabelled}$,
    a binary classification algorithm $\Phi$,
Ensure: manually labelled selected tweets $T_m$
$T_{labelled} \leftarrow \bigcup_{i=1}^{N} \psi_i$
// Use $\Phi$ to create a classifier $Cls$ from $T_{labelled}$:
$Cls \leftarrow \Phi : T_{labelled}$
// $T_{unlabelled}$ is classified as $T_{spam}$ and $T_{non-spam}$:
$T_{spam} + T_{non-spam} \leftarrow T_{unlabelled}$
// Merge spam tweets $T_{spam}$ classified by $Cls$ into
$T_{labelled}$:
$T_{ex} \leftarrow T_{labelled} + T_{spam}$
// use $T_{ex}$ to re-train the classifier $Cls$ :
$Cls \leftarrow \Phi : T_{ex}$
// determine the incoming tweet's suitability for
selection:
$U \leftarrow \emptyset$
for $i = 1$ to $k$ do
    if $U_i$ meet the selection criteria $S$ then
        $U \leftarrow (U \cup U_i)$
    end if
end for
// manually labelling each $u_i$ in $U$
$T_m \leftarrow \emptyset$
for $i = 1$ to $k$ do
    manually label each $u_i$
    $T_m \leftarrow (T_m \cup u_i)$
end for
```

However, the spam tweets detected by the classifier that is trained using T_l also have the same or similar distribution of old spam. We need samples from "changed spam" to calibrate the classifier. We then use LHL (in Section IV-B) to get "changed spam" samples.

B. Learning from Human Labelling

In a supervised spam detection system, a learning algorithm, such as Random Forest, must be trained by sufficient labelled data to obtain more accurate detection results. However, labelled instances are very expensive and time-consuming to obtain. Fortunately, we have a huge number of unlabelled tweets which can be easily collected. The LHL in our Lfun is best suited where there are numerous unlabelled data instances, and human annotator anticipating to label many of them to train an accurate system [33]. LHL aims to minimize the labelling cost by using different learning criteria to select most informative samples from unlabelled data to be labelled by a human annotator [44]. We also import active learning in our Lfun scheme.

Now let us define our learning component in a formal way. In supervised Twitter spam detection, we are given a labelled training data set $T_{training} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$, containing m labelled tweets, where $x_i \in \mathbb{R}^k (i = 1, 2, \ldots, m)$ is the feature vector of a tweet, $y_i \in \{spam, non - spam\}$ is the category label of a tweet. The label y_i of a tweet x_i is donated as $y = f(x)$. The task is then to learn a function $\hat{f}$ which can correctly classify a tweet to spam or non-spam. We use generalisation error to measure the accuracy of the learned function:

$$Error(\hat{f}) = \sum_{x \in T_{training}} \mathcal{L}\Big(f(x), \hat{f(x)}\Big) P(x).$$

In practice, $f(x)$ is not available for testing data instances. Therefore, it is usual to estimate the generalisation error by the test error:

$$Error(\hat{f}) = \sum_{x \in T_{testing}} \mathcal{L}\Big(f(x), \hat{f(x)}\Big) P(x),$$

where $T_{testing}$ refers to the testing tweets, and prediction error can be measured by a loss function $\mathcal{L}$, such as mean squared error (MSE) [31]:

$$\mathcal{L}_{MSE}\Big(f(x), \hat{f(x)}\Big) = \Big(f(x) - \hat{f(x)}\Big)^2.$$

The learning criteria is set to select the most useful instances $X_{selected}$ and add them to the training set $T_{training}$ for achieving some certain objectives. Let us consider this objective as the minimization of generation error of a learned function trained by $T_{training}$. So the learning criteria can be donated as

$$Error(T_{training} \cup \{X_{selected}\}).$$

The goal of this kind of learning is to select instances $X_{selected}$ which can minimize the generalisation error $Error(X_{selected})$:

$$argmin\ Error(X_{selected}).$$

As a result, good selection criteria must be estimated to minimize the error. In Lfun scheme, we apply the selection criteria, called "Probability Threshold Filter Model", to select the most informative tweets to tackle "Spam Drift". In order to achieve this, Random Forest (RF) is used to determine the probability of a tweet whether it belongs to spam or not. Random Forest [2] can generate many classification trees after being trained with T_{ex} from Asymmetric Self-Learning. When classifying a new incoming tweet, each tree in the forest will give a class prediction. Then forest chooses the classification result which has the most votes. In our case, we set the number of trees to m, if n trees vote for the class "spam", the probability of the tweet to be classified as "spam" is $Pr = \frac{n}{m}$.

Through our empirical study, the mis-classification mostly occurred when $Pr \in [0.4, 0.7]$. So we set the threshold τ to $Pr \in [0.4, 0.7]$. After we pre-filter some candidate tweets to be labelled using the "Probability Threshold Filter Model", the number of tweets is still too many. We then randomly select a smaller number of tweets from the candidate tweets (we set it to be 100 in our experiments) to be manually labelled. As shown in Fig. 3, the manually labelled tweets, along with T_{ex} will be used to train a new classifier, which can tackle "Spam Drift" problem.

C. Performance Benefit Justification

We study the performance benefit of the proposed Lfun scheme by providing the theoretical analysis in this section. Fig. 4 illustrates the performance benefit by using simulation.

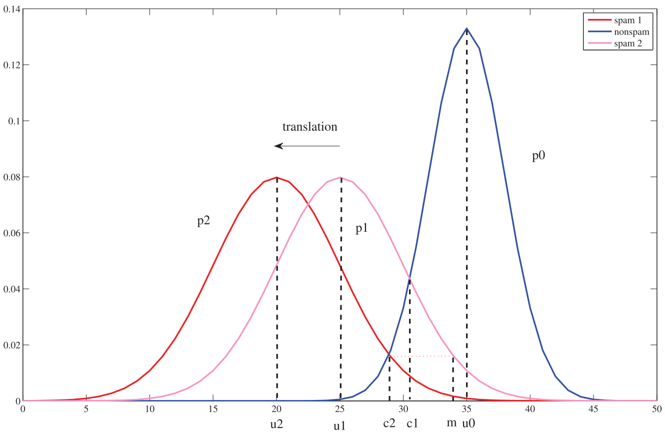


Fig. 4. Performance benefit illustration.

We use three normal distributions (listed below) to simulate this: w_0 represents the distribution of non-spam, while w_1 and w_2 represents the distribution of spam before and after using our Lfun approach, respectively.

$$\begin{cases} w_0 \sim N(\mu_0, \sigma_0^2) \\ w_1 \sim N(\mu_1, \sigma_{12}^2) \\ w_2 \sim N(\mu_2, \sigma_{12}^2) \end{cases}$$

The PDFs (probability distribution functions) [11] of these three distributions, w_0, w_1 and w_2 are illustrated as p_0, p_1 and p_2 in Fig. 4. We assume that only the mean μ_1 of w_1 changes to μ_2, but the variance σ_{12} is not changing.

As p_1 translated to p_2, we can always find m, which can make

$$m - c_2 = \mu_1 - \mu_2, \tag{1}$$

and

$$p_1(m) = p_2(c_2). \tag{2}$$

As $c_2 < c_1$, we have

$$p_0(c_2) < p_0(c_1). \tag{3}$$

We also have

$$p_0(c_1) = p_1(c_1), \;\; p_0(c_2) = p_2(c_2). \tag{4}$$

From Equation. 3 and Equation. 4, we get

$$p_1(c_1) > p_2(c_2). \tag{5}$$

From Equation. 2 and Equation. 5, we can have

$$p_1(c_1) > p_1(m). \tag{6}$$

As a result,

$$m > c_1. \tag{7}$$

Taking into account Equation. 7 and Equation. 1, we can have $c_1 - c_2 < \mu_1 - \mu_2$. So,

$$c_2 - \mu_2 > c_1 - \mu_1. \tag{8}$$

The error rate of classification before Lfun,

$$\begin{aligned} P_1(error) &= P(x > c_1) + P(x < c_2) \\ &= \int_{c_1}^{\infty} p_1(t)dt + \int_{-\infty}^{c_1} p_0(t)dt \\ &= 1 - \phi(\frac{c_1 - \mu_1}{\sigma_{12}}) + \phi(\frac{c_1 - \mu_0}{\sigma_0}). \end{aligned}$$

Similarly, we have the error rate after using Lfun

$$P_2(error) = 1 - \phi(\frac{c_2 - \mu_2}{\sigma_{12}}) + \phi(\frac{c_2 - \mu_0}{\sigma_0}).$$

The difference of $P_1(error)$ and $P_2(error)$,

$$\begin{aligned} P_1(error) - P_2(error) = &\Big[\phi(\frac{c_2 - \mu_2}{\sigma_{12}}) - \phi(\frac{c_1 - \mu_1}{\sigma_{12}})\Big] \\ &+ \Big[\phi(\frac{c_1 - \mu_0}{\sigma_0}) - \phi(\frac{c_2 - \mu_0}{\sigma_0})\Big], \end{aligned} \tag{9}$$

while

$$\phi(x) = \frac{1}{\sqrt{2\pi}} \int_0^x e^{-t^2/2} dt. \tag{10}$$

The differentiation of Equation 10 is $\phi'(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2} > 0$. So, we can have $\phi(a) > \phi(b)$ when $a > b$. From Equation. 8, we know $\frac{c_2 - \mu_2}{\sigma_{12}} > \frac{c_1 - \mu_1}{\sigma_{12}}$. Consequently,

$$\phi(\frac{c_2 - \mu_2}{\sigma_{12}}) > \phi(\frac{c_1 - \mu_1}{\sigma_{12}}). \tag{11}$$

As $c_1 > c_2$, we have $\frac{c_1 - \mu_0}{\sigma_0} > \frac{c_2 - \mu_0}{\sigma_0}$. Then, we know

$$\phi(\frac{c_1 - \mu_0}{\sigma_0}) > \phi(\frac{c_2 - \mu_0}{\sigma_0}). \tag{12}$$

Substitute Equation. 11 and 12 into 9, we will have

$$P_1(error) - P_2(error) > 0. \tag{13}$$

Obviously, our proposed approach can effectively reduce the probability of error from Equation 13.

V. Performance Evaluation

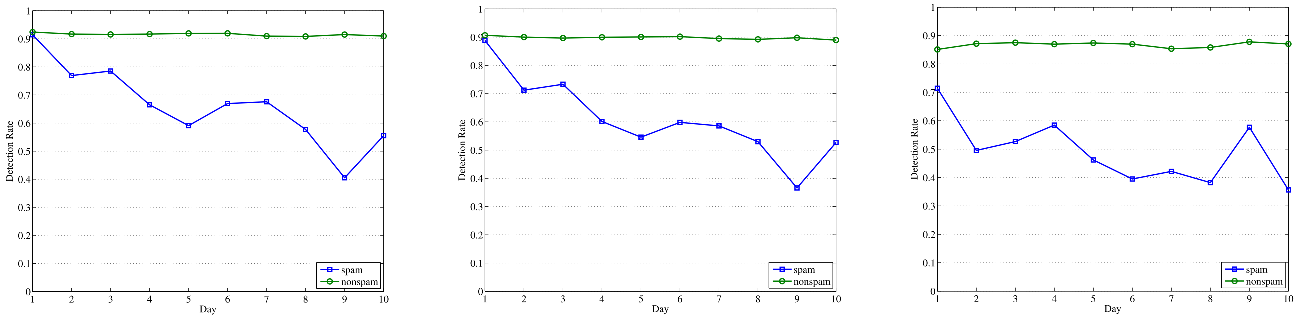
In this section, we evaluate the performance of the proposed Lfun scheme in detecting "drifted" Twitter spam. All the experiments are carried out on our real-world 10 consecutive days' tweets with each day containing 100k spam tweets and 100k non-spam tweets.

As in existing works [39], we also use *F-measure* and *Detection Rate* to measure the performance. Despite that both of the metrics are used to evaluate all the classes' performance, we only focus on the F-measure and Detection Rate of spam class. F-measure is an evaluation metric which combines precision and recall to measure the per-class performance of classification or detection algorithms. It can be calculated by

$$F - measure = \frac{2 * Precision * Recall}{Precision + Recall}.$$

Detection Rate is defined as the ratio of those tweets correctly classified as belonging to class *spam* to the total number of tweets in class *spam*, it can be calculated by

$$DetectionRate = \frac{TP}{TP + FN}.$$



(a)

(b)

(c)

Fig. 5. Trend of detection rate. (a) Random forest. (b) C4.5 Decision tree. (c) Bayes network.

In the evaluation, we have designed three sets of experiments in order to show the *impact of spam drift* (in Section V-A) firstly, then the benefit of our proposed Lfun (in Section V-B) and the comparisons with other traditional machine learning algorithms (in Section V-C). We repeat the experiments for 100 times with different random training samples and report the average values on all the 100 runs.

A. Impact of Spam Drift

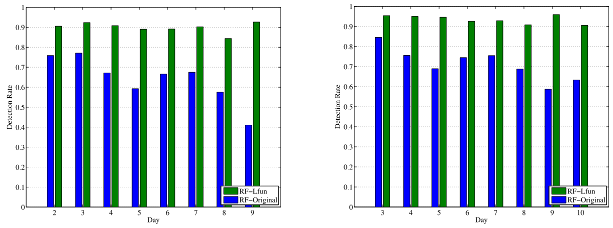
In order to evaluate the impact of "Spam Drift" problem, we perform a number of experiments in this section. It is aiming to show that the performance of a traditional classifier, for example C4.5 Decision Tree, varies over time when "Spam Drift" exists.

During these experiments, Day 1 data is divided into two parts, half for training pool where training data can be extracted from, and another half for testing purpose. We create a classifier by using a supervised classification algorithm, and train it with 10k spam and 10k non-spam tweets which are randomly sampled from the training pool of Day 1. Then the classifier is used to classify the testing data in Day1, as well as the testing samples in Day 2 to Day 10.

Fig. 5 shows the Detection Rate of both spam and non-spam tweets on three classifiers, Random Forest, C4.5 Decision Tree and Bayes Network. We can see that, the Detection Rate of non-spam is very stable, it keeps above 90% for Random Forest and C4.5 Decision Tree, and near 90% for Bayes Network, despite the change of testing data. However, when it comes to spam tweets, the Detection Rate fluctuates dramatically, and the overall trend is decreasing. The Detection Rates for Random Forest and C4.5 Decision Tree are 90% in the first day, but they could decrease to less then 40% in the 9th day. This phenomenon also applies with Bayes Network, the Detection Rate decreases from 70% on 1st day to less than 50% for most of the other testing days.

B. Performance of Lfun

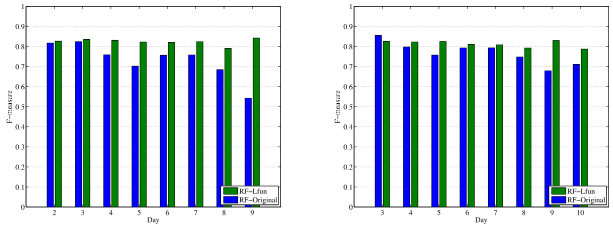
We evaluate the performance of Lfun here, by using F-measure and Detection Rate. The number labelled training samples from old day (*i.e.* Day 1 and Day 2 in this case) is 5000. The number of manually labelled samples during Lfun is set to 100.



(a)

(b)

Fig. 6. Detection rate of Lfun. (a) Day 1 training, Day 2 to 9 testing. (b) Day 2 training, Day 3 to 10 testing.



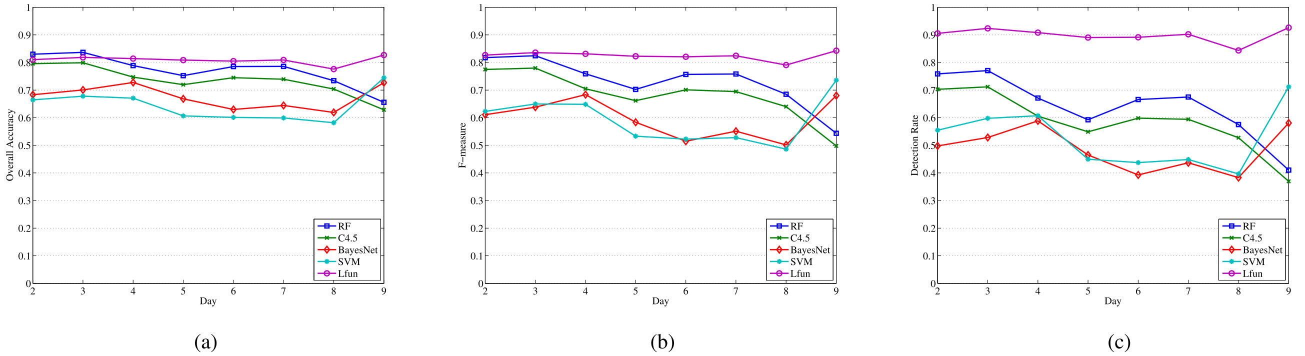
(a)

(b)

Fig. 7. F-measure of Lfun. (a) Day 1 training, Day 2 to 9 testing. (b) Day 2 training, Day 3 to 10 testing.

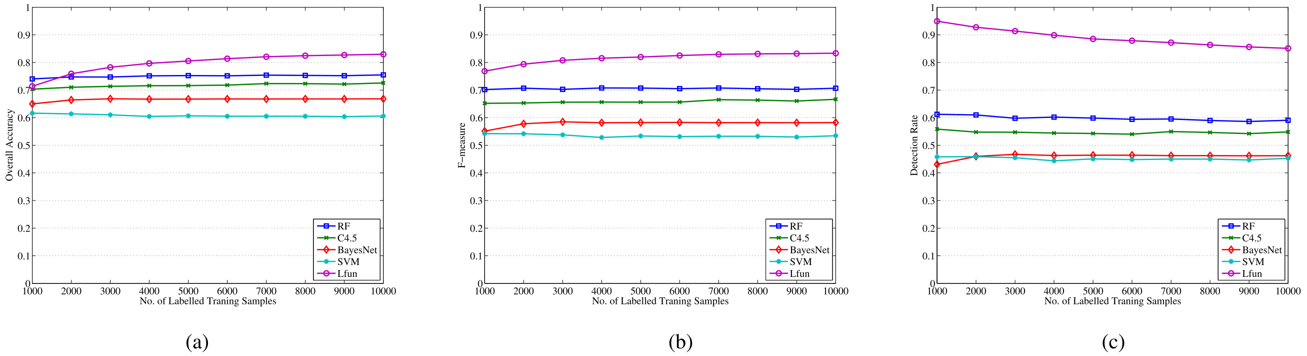
Fig. 6 shows the Detection Rate of Lfun, when Day 1 data (Fig. 6a) or Day 2 data (Fig. 6b) is used for training and the rest days are used for testing. We can see from Fig. 6a that, the Detection Rates of original Random Forest are relatively low. For example, the Detection Rate when testing on Day 9 is only around 40%. However, our RF-Lfun can reach over 90% Detection Rate on the same day. While Random Forest can only achieve Detection Rate ranging from 45% to 80%, our RF-Lfun can rise as high as 90% Detection Rate. This also happens when training data is from Day 2, and testing data is from Day 3 to Day 10, as illustrated in Fig. 6b. The highest Detection Rate of Random Forest is around 85%, but that of RF-Lfun is over 95%. Generally, our Lfun can detect most of the spam tweets even with "Spam Drift". The reason is that, our Lfun brings more samples of "changed spam tweets" to update the training process.

Fig. 7 shows the F-measure of Random Forest using Lfun approach compared with it without using Lfun. We can see that, the F-measure of original Random Forest keeps



(a) (b) (c)

Fig. 8. Comparisons with other Algorithms (changing testing days). (a) Overall accuracy. (b) F-measure. (c) Detection rate.



(a) (b) (c)

Fig. 9. Comparisons with other algorithms (training on day 1 and testing on day 5). (a) Overall accuracy. (b) F-measure. (c) Detection rate.

decreasing from 80% to 55% as the testing data changes from Day 2 to Day 9 in Fig. 7a. However, once it is applied with our Lfun approach, the F-measure becomes stable, which is always greater than 80%, except on Day 8. Similarly, when the training data is from Day 2, F-measure of Random Forest is decreasing as well. But F-measure of our Lfun-RF is not fluctuating, as shown in Fig. 7b. Nevertheless, the proposed Lfun can effectively improve the F-measure and the improvement is up to 25% in the best case.

C. Comparisons With Other Algorithms

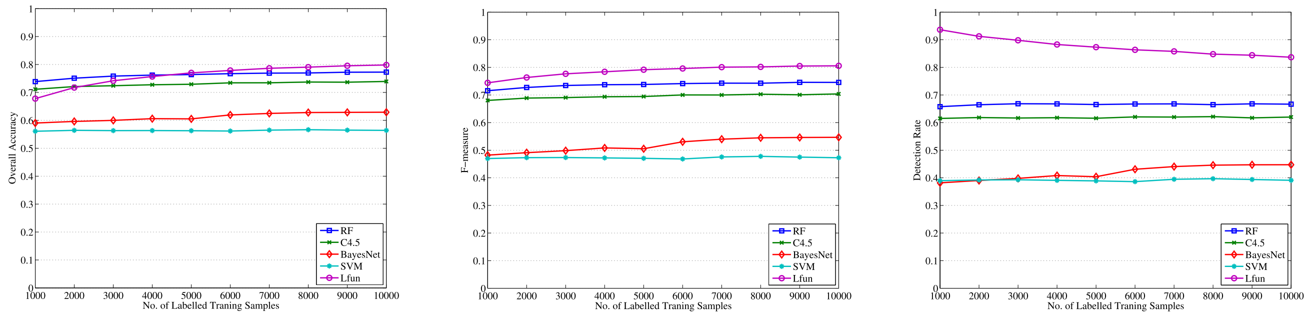
In this section, we compare our Lfun approach with four traditional machine learning algorithms (*Random Forest, C4.5 Decision Tree, Bayes Network and SVM*) to detect spam tweets in the "drift" scenario. There are two sets of experiments carried out. One set is to evaluate the performance while training data is from Day 1, and testing data are varying from Day 2 to Day 9. Another set is to evaluate the performance when training and testing data are from two specified days, but the number of labelled training data is changing from 1000 to 10000.

1) Comparisons With Changing Days: Fig. 8 demonstrates the experimental results in terms of overall accuracy, F-measure and detection rate of Lfun compared to other algorithms, when the testing days are varying. We can see from Fig. 8a that, the overall accuracy of Lfun outperforms all the other algorithms, followed by Random Forest, C4.5 Decision Tree, Bayes Network and SVM. In terms of F-measure (see Fig. 8b), our Lfun is also the best among all the algorithms. For example, it is over 30% higher than C4.5 Decision Tree when testing data is from Day 9. Furthermore, the performance of Lfun is much better in terms of detection rate. Fig. 8c show that, the detection rate of Lfun is above 90% for most of the days. However, the detection rate of all the others is below 80%. Especially, Bayes Network has the lowest detection rate, which is below 50%. In general, our Lfun is the best among all the algorithms evaluated by all the three metrics.

2) Comparisons With Changing Labelled Training Samples: Fig. 9 and Fig. 10 report the evaluation results when the number of labelled training samples is changing. The training and testing data is from Day 1 and Day 5 in Fig. 9, while the training and testing data is from Day 4 and Day 8 in Fig. 10. We can see that the overall accuracy of Lfun increases from 70% to 80% with the increase of labelled training samples. It is better than the four algorithms in comparison, as the best of them (C4.5 Decision Tree) can only achieve less than 74% overall accuracy. When it comes to F-measure, the performance of Lfun is still the best; it is 10% higher than that of C4.5 Decision Tree and nearly 30% higher than that of SVM. In terms of detection rate, our Lfun is about 30% higher than the second best algorithm. Similarly in Fig. 10, Lfun outperforms all the other algorithms.

VI. Discussions

In research community, there are also some machine learning approaches related to our proposed method. For example, online learning and incremental learning. They are both common machine learning algorithms to continuously update



(a) (b) (c)

Fig. 10. Comparisons with other algorithms (training on day 4 and testing on day 8). (a) Overall accuracy. (b) F-measure. (c) Detection rate.

the prediction model with new training data for better future classification. They can generate a prediction model and put it into operation without much training data at first, but they require new training data to update the model. When it comes to online Twitter spam classification, it is very difficult to label enough training samples to update the model. The reasons are two-folds. Firstly, it is significantly time-consuming to label a large amount of tweets by human. Secondly, it is difficult to gain enough spam tweets even we have got a large number of human-labelled tweets, as the spam rate of Twitter is about 5% [6]. If there are not enough spam samples (Lfun does not need non-spam samples as non-spam tweets are not drifting) to retrain the model, it is not able to solve the "spam drift" issue.

Our Lfun approach has the same advantage of online learning and incremental learning, *i.e.*, it can be deployed without much training data at the beginning, but to be updated when new training data comes. Different to online and incremental learning, we incorporate both automated labelling and human labelling. The LDT component learns from the detected tweets. This competent is automatically updated with detected spam tweets with no human effort. To better adjust the prediction model, we also import LHL component, which learns from human labelling. To minimize human effort, LHL only samples a very small number of tweets for labelling, for example, 100 tweets in our experiments. In addition, it does not randomly pick up tweets to label, but to be in line with selection criteria called "Probability Threshold Filter Model" which can choose the most useful tweets. Benefiting from these two components, our Lfun approach can successfully deal with "spam drift", but with the least human effort.

VII. Conclusion and Future Work

In this paper, we firstly identify the "Spam Drift" problem in statistical features based Twitter spam detection. In order to solve this problem, we propose a Lfun approach. In our Lfun scheme, classifiers will be re-trained by the added "changed spam" tweets which are learnt from unlabelled samples, thus it can reduce the impact of "Spam Drift" significantly. We evaluate the performance of Lfun approach in terms of Detection Rate and F-measure. Experimental results show that both detection rate and F-measure are improved a lot when applying with our Lfun approach. We also compare Lfun to four traditional machine learning algorithms, and find that our Lfun outperforms all four algorithms in terms of overall accuracy, F-measure and Detection Rate.

There is also a limitation in our Lfun scheme. The benefit of "old" labelled spam is to eliminate the impact of "spam drift" to classify more accurate spam tweets in future days. The effectiveness of "old" spam has been proved by our experiments during a short period. However, the effectiveness will decrease as the correlation of "very old" spam becomes less with the new spam in the long term run. In the future, we will incorporate incremental adjustment to adjust the training data, such as dropping the "too old" samples after a certain time. It can not only eliminate unuseful information in the training data but also make it faster to train the model as the number of training samples decrease.

Acknowledgment

The authors would like to thank Trend Micro for providing us the service to label spam tweets.

References

[1] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida, "Detecting spammer on twitter," in *Proc. 7th Annu. Collaboration, Electron. Messaging, Anti-Abuse Spam Conf.*, Jul. 2010, p. 12.

[2] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[3] C. Castillo, M. Mendoza, and B. Poblete, "Information credibility on twitter," in *Proc. 20th Int. Conf. World Wide Web*, 2011, pp. 675–684.

[4] C. Chen, J. Zhang, X. Chen, Y. Xiang, and W. Zhou, "6 million spam tweets: A large ground truth for timely twitter spam detection," in *Proc. IEEE Commun. Inf. Syst. Security Symp. (ICCCISS)*, Jun. 2015, pp. 8689–8694.

[5] C. Chen, J. Zhang, Y. Xiang, and W. Zhou, "Asymmetric self-learning for tackling twitter spam drift," in *Proc. 3rd Int. Workshop Security Privacy Big Data (BigSecurity)*, Apr. 2015, pp. 237–242.

[6] C. Chen, J. Zhang, Y. Xiang, W. Zhou, and J. Oliver, "Spammers are becoming 'smarter' on twitter," *IT Prof.*, vol. 18, no. 2, pp. 14–18, Apr. 2016.

[7] E. M. Clark, J. R. Williams, C. A. Jones, R. A. Galbraith, C. M. Danforth, and P. S. Dodds, "Sifting robotic from organic text: A natural language approach for detecting automation on twitter," *J. Comput. Sci.*, vol. 16, p. 1–7, Sep. 2016.

[8] (2016). *Whole Product Dynamic Real-World Protection Test*, Av Comparatives, accessed on Aug. 1, 2015. [Online]. Available: http://www.av-comparatives.org/wp-content/uploads/2016/07/avc_prot_2016a_en.pdf

[9] I. Csiszar and J. Körner, *Information Theory: Coding Theorems For Discrete Memoryless Systems*. Cambridge, U.K.: Cambridge Univ. Press, 2011.

[10] T. Dasu, S. Krishnan, S. Venkatasubramanian, and K. Yi, "An information-theoretic approach to detecting changes in multi-dimensional data streams," in *Proc. Symp. Interface Statist., Comput. Sci., Appl.*, 2006, pp. 1–24.

[11] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. Hoboken, NJ, USA: Wiley, 2000.

[12] M. Egele, G. Stringhini, C. Kruegel, and G. Vigna, "Compa: Detecting compromised accounts on social networks," in *Proc. Annu. Netw. Distrib. Syst. Security Symp.*, 2013.

[13] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *J. ACM Comput. Surv.*, vol. 46, no. 4, p. 44, Apr. 2014.

[14] H. Gao, Y. Chen, K. Lee, D. Palsetia, and A. Choudhary, "Towards Online spam filtering in social networks," in *Proc. NDSS*, 2012, pp. 1–16.

[15] H. Gao, J. Hu, C. Wilson, Z. Li, Y. Chen, and B. Y. Zhao, "Detecting and characterizing social spam campaigns," in *Proc. 10th ACM SIGCOMM Conf. Internet Meas.*, 2010, pp. 35–47.

[16] H. Gao *et al.*, "Spam ain't as diverse as it seems: Throttling OSN spam with templates underneath," in *Proc. 30th Annu. Comput. Security Appl. Conf.*, 2014, pp. 76–85.

[17] A. Greig. (2013). *Twitter Overtakes Facebook as the Most Popular Social Network for Teens,According to Study*, DailyMail, accessed on Aug. 1, 2015. [Online]. Available: http://www.dailymail.co.uk/news/article-2475591/Twitter-overtakes-Facebook-popular-social-network-teens-according-study.html

[18] C. Grier, K. Thomas, V. Paxson, and M. Zhang, "@spam: The underground on 140 characters or less," in *Proc. 17th ACM Conf. Comput. Commun. Security*, 2010, pp. 27–37.

[19] A. Gupta, P. Kumaraguru, C. Castillo, and P. Meier, *TweetCred: Real-Time Credibility Assessment of Content on Twitter*. New York City, NY, USA: Springer, 2014.

[20] K. Huang, Z. Xu, I. King, M. Lyu, and C. Campbell, "Supervised self-taught learning: Actively transferring knowledge from unlabeled data," in *Proc. Neural Netw., IJCNN Int. Joint Conf.*, Jun. 2009, pp. 1272–1277.

[21] R. Jeyaraman. (2014). *Fighting Spam With Botmaker*, Twitter, accessed on Aug. 1, 2015. [Online]. Available: https://blog.twitter.com/2014/fighting-spam-with-botmaker

[22] R. Krichevsky and V. Trofimov, "The performance of universal encoding," *IEEE Trans. Inf. Theory*, vol. 27, no. 2, pp. 199–207, Mar. 1981.

[23] H. Kwak, C. Lee, H. Park, and S. Moon, "What is twitter, a social network or a news media?" in *Proc. 19th Int. Conf. World Wide Web*, 2010, pp. 591–600.

[24] K. Lee, J. Caverlee, and S. Webb, "Uncovering social spammers: Social honeypots + machine learning," in *Proc. 33rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*,2010, pp. 435–442.

[25] S. Lee and J. Kim, "Warningbird: A near real-time detection system for suspicious URLs in twitter stream," *IEEE Trans. Depend. Sec. Comput.*, vol. 10, no. 3, pp. 183–195, May 2013.

[26] J. M. Romo and L. Araujo, "Detecting malicious tweets in trending topics using a statistical analysis of language," *Expert Syst. Appl.*, vol. 40, no. 8, p. 2992–3000, 2013.

[27] J. Oliver, P. Pajares, C. Ke, C. Chen, and Y. Xiang, "An in-depth analysis of abuse on twitter," Trend Micro, Irving, TX, USA, Tech. Rep., Sep. 2014.

[28] I. Ounis, C. Macdonald, J. Lin, and I. Soboroff, "Overview of the TREC-2011 microblog track," in *Proc. 20th Text Retr. Conf. (TREC)*, 2011.

[29] C. Pash. (2014). *The lure of Naked Hollywood Star Photos Sent the Internet into Meltdown in New Zealand*, Bus. Insider, accessed on Aug. 1, 2015 [Online]. Available: http://www.businessinsider.com.au/the-lure-of-naked-hollywood-star-photos-sent-the-internet-into-meltdown-in-new-zealand-2014-9

[30] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning: Transfer learning from unlabeled data," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 759–766.

[31] N. Rubens, D. Kaplan, and M. Sugiyama, "Active learning in recommender systems," in *Recommender Systems Handbook*. New York City, NY, USA: Springer, 2011.

[32] R. Sebastiao and J. A. Gama, "Change detection in learning histograms from data streams," in *Proc. Aritficial Intell. 13th Portuguese Conf. Progr. Artificial Intell.*, 2007, pp. 112–123.

[33] B. Settles, "Active learning literature survey," *Univ. Wisconsin, Madison*, vol. 52, nos. 55–66, p. 11, 2010.

[34] J. Song, S. Lee, and J. Kim, "Spam filtering in twitter using sender-receiver relationship," in *Proc. 14th Int. Conf. Recent Adv. Intrusion Detection*, 2011, pp. 301–317.

[35] G. Stringhini, C. Kruegel, and G. Vigna, "Detecting spammers on social networks," in *Proc. 26th Annu. Comput. Security Appl. Conf.*, 2010, pp. 1–9.

[36] K. Thomas, C. Grier, J. Ma, V. Paxson, and D. Song, "Design and evaluation of a real-time URL spam filtering service," in *Proc. IEEE Symp. Security Privacy*, 2011, pp. 447–462.

[37] K. Thomas, C. Grier, D. Song, and V. Paxson, "Suspended accounts in retrospect: An analysis of twitter spam," in *Proc. ACM SIGCOMM Conf. Internet Meas. Cof.*, 2011, pp. 243–258.

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

[39] C. Yang, R. Harkreader, and G. Gu, "Empirical evaluation and new design for fighting evolving twitter spammers," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 8, pp. 1280–1293, Aug. 2013.

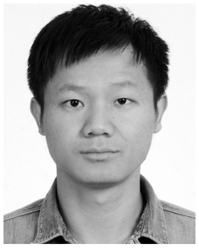
[40] C. Yang, R. Harkreader, J. Zhang, S. Shin, and G. Gu, "Analyzing spammers' social networks for fun and profit: A case study of cyber criminal ecosystem on twitter," in *Proc. 21st Int. Conf. World Wide Web*, 2012, pp. 71–80.

[41] S. Yardi, D. Romero, G. Schoenebeck, and D. Boyd, "Detecting spam in a twitter network," *First Monday*, vol. 15, nos. 1–4, pp. 1–13, Jan. 2010.

[42] J. Zhang, C. Chen, Y. Xiang, W. Zhou, and Y. Xiang, "Internet traffic classification by aggregating correlated naive bayes predictions," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 1, pp. 5–15, Jan. 2013.

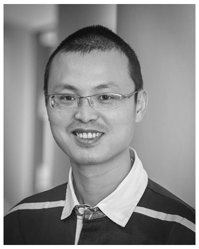
[43] X. Zhang, S. Zhu, and W. Liang, "Detecting spam and promoting campaigns in the twitter social network," in *Proc. Data Mining IEEE 12th Int. Conf. (ICDM)*, 2012, pp. 1194–1199.

[44] I. Zliobaite, A. Bifet, B. Pfahringer, and G. Holmes, "Active learning with drifting streaming data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 1, pp. 27–39, Jan. 2014.



Chao Chen received the B.Tech. degree (Hons.) from Deakin University, Australia, in 2012. He is currently pursuing the Ph.D. degree with the School of Information Technology, Deakin University. His research interests include network security and online social network security.

Yu Wang received the Ph.D. degree in computer science from Deakin University, Victoria, Australia. He is currently with the School of Information Technology, Deakin University. His research interests include network traffic modeling and classification, social networks, mobile networks, and network security.



Jun Zhang received the Ph.D. degree in computer science from the University of Wollongong, Wollongong, Australia, in 2011. He is currently with the School of Information Technology, Deakin University, Melbourne, Australia. He has authored over 50 research papers in refereed international journals and conferences, such as IEEE/ACM Transactions on Networking, IEEE Transactions on Image Processing, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Information Forensics and Security, IEEE Transactions on Systems, Man, and Cybernetics—Part B, and IEEE Transactions on Network and Service Management. His research interests include network and system security, pattern recognition, and multimedia retrieval.

Yang Xiang received the Ph.D. degree in computer science from Deakin University, Australia.

He is currently the Director of the Center for Cyber Security Research and also a Professor with the School of Information Technology, Deakin University. He has authored over 100 research papers in many international journals and conferences, such as IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE TRANSACTIONS ON INFORMATION SECURITY AND FORENSICS, and IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS. His research interests include network and system security, distributed systems, and networking. In particular, he is currently leading in a research group developing active defense systems against large-scale distributed network attacks. He is the Chief Investigator of several projects in network and system security, funded by the Australian Research Council. He has served as the Program/General Chair for many international conferences such as the ICA3PP 12/11, IEEE/IFIP EUC 11, IEEE TrustCom 11, IEEE HPCC 10/09, IEEE IC- PADS 08, and the NSS 11/10/09/08/07. He has been the PC member of over 50 international conferences in distributed systems, networking, and security. He serves as an Associate Editor of IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS and an Editor of *Journal of Network and Computer Applications*.

Wanlei Zhou (SM'09) received the B.Eng. and M.Eng. degrees from the Harbin Institute of Technology, Harbin, China, in 1982 and 1984, respectively, the Ph.D. degree from Australian National University, Canberra, in 1991, and the D.Sc. degree from Deakin University, Melbourne, Australia, in 2002. He was with a number of organisations including, the University of Electronic Science and Technology of China, Chengdu, China, Apollo/HP, Massachusetts, USA, the National University of Singapore, Singapore, and Monash University, Melbourne, Australia. He is currently the Chair Professor of Information Technology and also an Associate Dean with the Faculty of Science, Engineering and Built Environment, Deakin University, Melbourne, Australia. He has authored over 300 papers in refereed international journals and refereed international conferences proceedings. His research interests include network security, distributed and parallel systems, bioinformatics, mobile computing, and e-learning.

Geyong Min received the B.Sc. degree in computer science from the Huazhong University of Science and Technology, China, in 1995, and the Ph.D. degree in computing science from the University of Glasgow, U.K., in 2003. He is currently a Professor of High Performance Computing and Networking with the Department of Mathematics and Computer Science, College of Engineering, Mathematics, and Physical Sciences, University of Exeter, U.K. His research interests include future internet, computer networks, wireless communications, multimedia systems, information security, high performance computing, ubiquitous computing, modeling, and performance engineering.