Rule-Based On-the-fly Web Spambot Detection Using Action Strings

Pedram Hayati
Anti Spam Research Lab
Digital Ecosystems & Business Intelligence Institute, Curtin Business School, Curtin University, Perth, Western Australia, Australia
p.hayati@curtin.edu.au

Vidyasagar Potdar
Anti Spam Research Lab
Digital Ecosystems & Business Intelligence Institute, Curtin Business School, Curtin University, Perth, Western Australia, Australia
v.potdar@curtin.edu.au

Alex Talevski
Anti Spam Research Lab
Digital Ecosystems & Business Intelligence Institute, Curtin Business School, Curtin University, Perth, Western Australia, Australia
a.talevski@curtin.edu.au

William F. Smyth
Algorithms Research Group,
Department of Computing & Software
McMaster University, Hamilton, Ontario, Canada L8S 4K1
smyth@mcmaster.ca[1]

ABSTRACT

Web spambots are a new type of internet robot that spread spam content through Web 2.0 applications like online discussion boards, blogs, wikis, social networking platforms etc. These robots are intelligently designed to act like humans in order to fool safeguards and other users. Such spam content not only wastes valuable resources and time but also may mislead users with unsolicited content. Spam content typically intends to misinform users (scams), generate traffic, make sales (marketing/advertising), and occasionally compromise parties, people or systems by spreading spyware or malwares.

Current countermeasures do not effectively identify and prevent web spambots. Proactive measures to deter spambots from entering a site are limited to question / response scenarios. The remaining efforts then focus on spam content identification as a passive activity. Spammers have evolved their techniques to bypass existing anti-spam filters.

In this paper, we describe a rule-based web usage behaviour action string that can be analysed using Trie data structures to detect web spambots. Our experimental results show the proposed system is successful for on-the-fly classification of web spambots hence eliminating spam in web 2.0 applications.

1. INTRODUCTION

A fake profile in an online communities, an unsolicited comment in blog, a commercial unwelcome thread in an online discussion boards etc are example of new form of spamming techniques called spam 2.0 [1, 2]. Spam 2.0 offers a far more attractive proposition for spammers as compared to traditional spam specifically email spam.

Spammers can discover web 2.0 applications and use automated tools to distribute spam information that is targeted at a demographic of their choice with very little resistance. A single spam 2.0 attack may reach many targeted and domain specific users and online messages typically cannot be deleted by regular users and persist until an administrator deals with them often impacting many users in the meantime.

Spam 2.0 posts also have a parasitic nature. They may exist on legitimate and often official websites. If such information persists, the trust in such pages is diminished, spam is effectively promoted by trusted sources, many users can be mislead or lead to scams and computer malware and such legitimate sites may be blacklisted which then deprives all others of legitimate content. As a result of the success and impact rates of spam 2.0, it is far more popular amongst spammers and has far greater negative socio-economic impact.

Such spamming techniques promote the use of Web Spambots (or simply spambots): a web crawler that navigates the World Wide Web with the sole purpose of planting unsolicited content on external websites. To date, as Web 2.0 platforms are getting more prevalent, spam 2.0 are becoming more and more widespread. In order to avoid being spuriously used in this way (and to eliminate the clutter and wasted time created by spambots), websites seek tools to recognize and deter spambots as they arrive. Usually the tools used for this purpose are based on challenge-response techniques, such as Completely Automated Public Turing test to

CEAS 2010 – Seventh annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference, July 13-14, 2010, Redmond, Washington, US

[1] The work of the third author was supported in part by the Natural Sciences & Engineering Research Council of Canada.

tell Computers and Human Apart (CAPTCHA), a well-known technique typically in the form of an image used to block web crawlers access to a website [3]. However such techniques are about to expire as it decreases human users' convenience and computers are getting more and more powerful day by day [2, 4, 5].

In this paper we propose a rule-based combinatorial approach that distinguishes automatically between spambots and regular human users based on their usage patterns:

- Introduce the idea of using web usage behaviour to investigate spambots behaviour on the Web,
- Propose a new concept – Action Strings, a feature set extracted from web usage behaviour to model spam users vs. human users behaviour,
- Illustrate a novel rule-based classifier using Trie structure [6-8] for fast and on-the-fly spambot detection,

Here we describe the experience with a prototype system developed by us. In the next section we discuss about two major concepts – web usage data and trie structure that we used in our proposed system

2. BACKGROUND

2.1 Web Usage Data

Users browsing websites navigate through web objects such as web pages, image, files, documents etc. Such data can be recorded and later mined to extract valuable information [9]. This tracking data is referred to as web usage data. Web usage data has been widely employed in many studies to understand visitor browsing patterns, make personalised websites, improve web performance and to implement decision support and recommendation systems [10] [11]. Web usage data may contain the following fields:

- The visited URL,
- The referrer URL,
- Timestamp,
- IP address,
- Browser/Operating System identity.

In our work, we associate two additional attributes to each entry – a unique session identity (session ID) and a user account name of a user who makes a request. The former makes it possible to track a user while the user is visiting a website, while the latter includes their username in each record to make it possible to track user activities over a period of time.

2.2 Trie

A trie structure is a way to store and retrieve information. Its main advantages over other techniques are: ease of updating and handling, shorter access time, and removing redundancies in the data structure [6]. A trie is in the form of a tree structure where each node contains the value of the key it is associated with.

In this paper we construct a trie structure which consists of human and spambot behaviour patterns. The trie gives fast on-the-fly pattern matching ability and we use that for spambot pattern detection.

3. PROBLEM

To set the scene for this paper, we begin with a brief overview of the problems in spam 2.0 and spambot detection. A number of solutions have been proposed to classify spam in web 2.0 environments such as: opinion spam detection [12], spam in social networks [13-15], spam in video sharing websites [1, 16]. Most of these solutions have focussed on one particular form of spam. By extracting features from the content of spam 2.0, such techniques try to make spam content separate from legitimate content. However most of them did not come up with satisfactory classification results. Hence, this leads us to investigate different approaches to identify spam 2.0.

This research continues in line with our previous work on combating spam 2.0 [2]. Our main idea is to actively investigate spambot – as an origin of spam 2.0 instead of seeking to discriminate attributes inside spam content.

The concept of spambot identification has not been investigated thoroughly. Hence, we first provide a formal definition for this problem.

Similar to the spam classification problem [17] spambot detection can be formalised as a binary classification problem as follows

$$U = \{u_1, u_2, ..., u_{|U|}\} \tag{1}$$

Where U is set of users visiting a website, u_i is the i^{th} user in a set.

$$C = \{c_h, c_s\} \tag{2}$$

Where C refers to the overall set of users, c_h refers to human user class and c_s refers to spambot user class. the binary classification $\phi(u_i, c_j)$ is

$$\phi(u_i, c_j) : U \times C \rightarrow \{0,1\} \tag{3}$$

Where

$$\phi(u_i, c_j) = \begin{cases} 1 & u_i \in c_s \\ 0 & otherwise \end{cases} \tag{4}$$

Since in spambot detection problem each u_i belongs to one and only one class, the decision function can be simplified as $\phi(u_i)_{spam} : U \rightarrow \{0,1\}$.

4. HUMAN BEHAVIOUR VS. SPAMBOT BEHAVIOUR

The main assumption of our proposed method is that human web usage behaviour is intrinsically different from spambot behaviour. The reason is that spambots have different intentions. Spambots visit the website mainly to spread spam content rather than to consume the content. Hence by investigating and mining web usage data it is possible to distinguish spambots from human users. This viewpoint is different from previous studies which have focused on content and meta-content based features [1].

123.123.123.123 - - [10/Jul/2009:00:19:25 +0800] "GET /forum/index.php?action=post;board=1 HTTP/1.0" 200 7852 "http://example.com/forum/index.php?action=post;board=1" "Opera/9.0 (Windows NT 5.1; U; en)"

123.123.123.123 - - [10/Jul/2009:00:19:30 +0800] "GET /forum/index.php?board=1 HTTP/1.0" 200 6248 "http://example.com/forum/index.php?board=1" "Opera/9.0 (Windows NT 5.1; U; en)"

(a) Spambot HTTP header requests

111.111.111.111 - - [15/Aug/2009:07:05:10 +0800] "GET /forum/index.php HTTP/1.0" 200 60 "http://example.com/" "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-GB; rv:1.9.0.13) Gecko/2009073022 Firefox/3.0.13 (.NET CLR 3.5.30729)"

111.111.111.111 - - [15/Aug/2009:07:06:12 +0800] "GET /forum/index.php?board=1 HTTP/1.0" 200 6248 "http://example.com/forum/index.php" "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-GB; rv:1.9.0.13) Gecko/2009073022 Firefox/3.0.13 (.NET CLR 3.5.30729)"

111.111.111.111 - - [15/Aug/2009:07:08:32 +0800] "GET /forum/index.php?action=post;board=1 HTTP/1.0" 200 7852 "http://example.com/forum/index.php?board=1" "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-GB; rv:1.9.0.13) Gecko/2009073022 Firefox/3.0.13 (.NET CLR 3.5.30729)"

111.111.111.111 - - [15/Aug/2009:07:10:27 +0800] "GET /forum/index.php?topic=1397 HTTP/1.0" 200 524 "http://example.com/forum/index.php?action=post;board=1" "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-GB; rv:1.9.0.13) Gecko/2009073022 Firefox/3.0.13 (.NET CLR 3.5.30729)"

(b) Human HTTP header requests

Figure 1. Series of spambot http requests (a) vs. human http requests (b) in an online forum.

Figure 1 illustrates a simple series of HTTP requests sent from both humans and spambots to submit new content to an online forum. Each entry consists of an IP address, timestamp, request method, the requested URL, the response status code, the referrer URL and browser identity. There are some obvious differences between spambot and human requests. For example, the human's sequence of requests is more expected or on average, spambot sessions last for some seconds while human sessions last for 5 minutes etc. This factor shows that spambots follow pre-programmed and optimised procedures. Such key differences in web usage behaviour can be utilised to distinguish humans and spambots.

5. ACTION STRINGS

In order to model web usage data into a behavioural model, we propose an action as a set of user efforts to achieve certain purposes (E.q. 5). For example, in an online forum, in order to create a new user account, the user needs to navigate to the registration page, complete the registration form and submit this information. This procedure can be formulated as "Register a new user account" action. Actions abstract web usage data. Actions can be a suitable discriminative feature to model user behaviour and can also be extendible to many other Web 2.0 platforms.

Given a set of webpages $W = \{w_1, w_2, ..., w_{|W|}\}$, A is defined as a set of Actions, such that

$$A = \{a_i \mid a_i \subset W\} = \{\{w_l, ..., w_k\}\} \quad 1 \le l, k \le |W| \tag{5}$$

Respectively s_i is defined as

$$s_i = (a_j) \quad 1 \le i \le |T|; \quad 1 \le j \le |A| \tag{6}$$

s_i refers to a sequence of actions called *Action Strings*, performed in session i and T is total number of sessions.

6. RULE-BASED ON-THE-FLY SPAMBOT DETECTION METHOD

Figure 2 provides an overview of the monitoring strategy in algorithmic form. Each user (u) has multiple sessions and each session contains a series of performed actions (s_i). For every new incoming u our algorithm goes down through the trie to find matching nodes. If the probability for that specific action string (1- $P_H(s_i)$) is higher than a given threshold, our algorithm classifies the incoming u as a spambot.

for every new user *u* do
    location ← root of trie
    for every action *i* do
        location ← (*location*, *i*)
        $P_H(s_i)$ ← eval(*location*)
        if 1− $P_H(s_i)$ > *threshold* then
            zap(*u*)
    until *u* exits

Figure 2. Overview of user monitoring strategy

6.1 Framework

We propose a rule-based on-the-fly spambot detection framework by using the trie data structure. Our framework consists of five steps as follows:

6.1.1. Step1: Monitoring web usage data. Track web usage data and the navigation stream. For each incoming HTTP

request, IP address, visited URL, referrer URL, browser identity, timestamp, session ID and user account name are tracked.

6.1.2. Step2: Data cleaning and preparation. Once data is aggregated through Step 1, it needs to be cleansed of irrelevant information such as visitors who did not create a user account in the system. Next, data needs to be grouped into meaningful user navigation clusters. This task is known as Transaction Identification [9]. As discussed in Section 2.1, we accompany each tracking record with a unique session ID. Each session ID can assist us to track a user's navigation. In our proposed framework we group data based on session IDs. Therefore, requests made by a user in a single visit are clearly identifiable. For example, a user may visit a system three times in a day, each time our proposed framework assigns a unique session ID to the user. Hence, it is possible to track user requests in each visit. Moreover, defining transactions at the session level can be utilised for fast and on-the-fly classification as well as to provide in-depth insights into the user behaviour.

6.1.3. Step3: Action extractions and formulation. Table 1 provides a summary of different actions that can be performed by users in our system. Each action – defined in Section 5 – comes along with an index key that is used to formulate a user's interaction type. Such action lists can be extended to other web 2.0 applications and can be defined beforehand for a specific platform. Upon new incoming traffic, our framework assigns associated actions. Later, by putting each action index key together, action strings can be build up as defined in Section 5. Action strings grow over time as the user performs actions, as well as preserving the order of the action sequence. As mentioned in Section 4, such a sequence can be utilised as a discriminative feature to distinguish humans and spambots

Table1. Action Index Key and Action Lists

Action Key	Description
A	View root page
B	View topics
C	Start new topic
D	View topic (view their own created topic)
E	View user profile information
F	Edit user information
G	Start new poll
H	User authentication (Login)
I	Reply to topic
J	View recently submitted topics
K	View statistic
L	Spell checking
M	Send private message
N	Search
O	Get new topics
P	Update topic
Q	Preview topic

6.1.4. Step4: Building up a trie data structure. Once the action strings for both human and spambots have been created, our framework builds a trie data structure based on each action string. Each trie edge contains an action key index and each node contains the probability of a specific action string being either human or spambot. This probability is computed through E.q 7 and E.q. 8.

$$P_H(s_i) = \frac{f_H(s_i)}{f_H(s_i) + f_B(s_i)} \quad (7)$$

$$P_B(s_i) = \frac{f_B(s_i)}{f_H(s_i) + f_B(s_i)} \quad (8)$$

where s_i is an action string, $f_H(s_i)$ is the frequency of human action strings that have prefix s_i. $f_B(s_i)$ is the frequency of spambots with prefix s_i, $P_H(s_i)$ is the probability of s_i being human and $P_B(s_i)$ of its being spambot, respectively.

Zero probability means that that particular action string has never occurred before. For example Figure 3 illustrates a trie data structure for a set of {ABC, ABCD, ABDE, ABC} action strings for a human and {ABD, ABD, ABDE} action strings for a spambot.

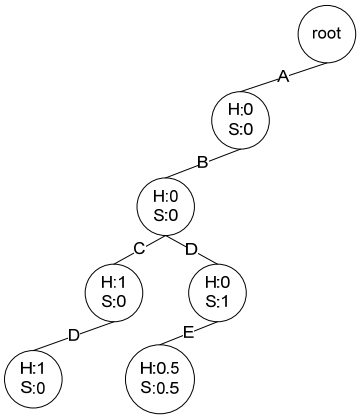

Figure 3. Simple trie data structure for set of {ABC, ABCD, ABDE, ABC} for a human and {ABD, ABD, ABDE} for a spambot. S and H represent probability of action string belonging to spambots and humans respectively.

6.1.5. Step5: Classification. New incoming action string s_i is validated through trie structure that has been built in Step4. After validation s_i can fall into two categories – Match and Not-Matched. In the former case, our framework looks at $P_H(s_i)$ and if $1 - P_H(s_i) > Threshold$, s_i would be classified as spambot. In the latter situation no decision is made; this is a focus of our future work.

6.2 Performance Measurement

We used Matthews Correlation Coefficient (MCC) method to measure the performance of our proposed framework [18]. MCC is one of the best performance measurement methods of binary classifications especially when the data among two classes of data is not balanced [19]. It considers true and false positives and returns a value between -1 and +1. If the return value is closer to +1 the classification result is better and the decision can be

considered to have greater certainty. However, if the result value is close to 0 it shows the output of the framework is similar to random prediction. A result value closer to -1 shows a strong inverse ability of the classifier. MCC is defined as follows;

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (9)$$

In Eq. 9, TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives and FN is the number of false negatives.

7. EXPERIMENTAL RESULTS & DISCUSSION

7.1 Dataset

According to our understanding, there is no publicly available collection, which contains both human and spambot web usage data for web 2.0 platform. Hence, we use the same dataset that we collected in our previous study [2]. We collected human user data from a human moderated online forum and our spambot data from our HoneySpam 2.0 project [2], which runs same online forum application with the same configurations as human forum. Each entry in our dataset contains visited URL link, referrer URL link, timestamp, browser identity, and IP address; and it is associated with session ID and forum username. Our dataset contains 16594 entries consisting of 11039 spambots records and 5555 human records. In the action extraction and formulation step we come up with 34 individual actions. We used 2/3 of the data for feeding the trie in Building up a trie data structure step and use the remaining 1/3 for evaluation purpose.

7.2 On-The-Fly Detection

Our proposed framework has a real-time detection or on the fly detection feature. The system creates action strings as they happen, i.e. based on the user webpage behaviour. The aim is to identify the spam behaviour pattern in the least amount of actions and then flag that transaction as spambot. We make a window over test action strings, run our classifier and increase the window's size by one character for the next run. Such situations can simulate real world practices where user action strings grow over the time.

7.3 Results

We make five random datasets (*DS1* to *DS5*) and run our framework based on each dataset. The window size ranges from 2 to 10 characters (the length of the longest action string in our dataset is 10). Additionally we vary the threshold from -0.05 to 0.05. We measure the performance of our framework by using MCC for each experiment.

Figure 4 illustrates the accuracy of our proposed system for different window sizes on our five random datasets. The best accuracy of 96% was achieved on DS1 with window size equal to or larger than 4 (Table 3), while overall accuracy is 93% over all datasets. Table 2 summarises average accuracy for our 5 random datasets.

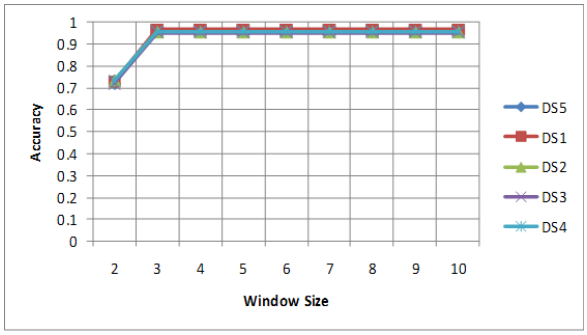

Figure 4. Accuracy of our proposed system for different windows size on 5 random datasets. Threshold maintains at zero for all experiments.

Table2. Average Accuracy of datasets

	DS1	DS2	DS3	DS4	DS5
Accuracy	0.939	0.930	0.926	0.933	0.934

Table3. Accuracy of different windows sizes on our datasets

	DS1	DS2	DS3	DS4	DS5
Len 2	0.727	0.734	0.719	0.732	0.721
Len 3	0.965	0.955	0.952	0.958	0.960
Len 4	0.965	0.955	0.952	0.958	0.960
Len 5+	0.966	0.955	0.952	0.958	0.960

Figure 5 presents MCC value for different windows sizes. Overall average MCC on all datasets is 0.744. Table 4 shows the value of MCC on different datasets. The best MCC value 0.802 was achieved on DS1 for window size equal to or larger than 5 (Table 5).

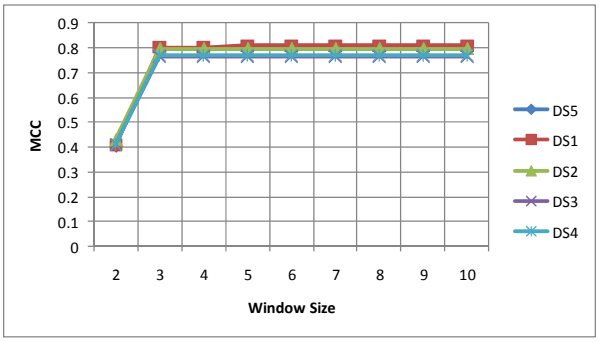

Figure 5. MCC values of different window size on 5 random datasets.

Table4. Average MCC of datasets

	DS1	DS2	DS3	DS4	DS5
MCC	0.762	0.754	0.725	0.729	0.749

Table5. MCC value of different windows size on our datasets

	DS1	DS2	DS3	DS4	DS5
Len 2	0.406	0.430	0.412	0.410	0.405
Len 3	0.802	0.795	0.764	0.769	0.792
Len 4	0.802	0.795	0.764	0.769	0.792
Len 5+	0.808	0.795	0.764	0.769	0.792

Figure 6 illustrates the performance of our proposed system based on different threshold at different window sizes (*Len2* to *Len6*).

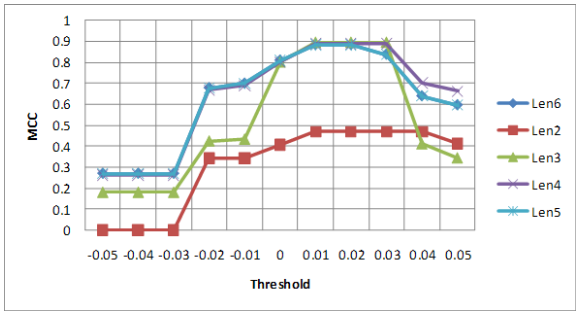

Figure 6. Performance of system based on different thresholds for different window size (Len2 to Len6)

The best MCC value of 0.88 is achieved for thresholds between 0 and 0.01 for window size equal to or larger than 4 (*Len4* to up).

The performance of our proposed system becomes better as window size grows. This shows that our system can predict better as user uses the system over time. However, the performance remains the same after some windows size. The reason is that our datasets are randomly selected and not all the action string combinations are included during Step 4, *Building up a trie data structure.* Hence some of the action strings are not included in trie structure and our system may not be able to match incoming unseen action string. The same behaviour can be seen in the accuracy of our results (Fig. 4), where accuracy remains the same after a certain window size (window size 4).

In Figure 6 it is obvious that moving the threshold toward negative values increases the number of false-negatives while moving toward positive values increases the number of false positives. Therefore, MCC on such values is low.

8. RELATED WORKS

Although the topic of spam has been investigated extensively, to the best of our knowledge research into spambot detection on Web 2.0 applications and spam 2.0 is quite young and has not received comprehensive attention. In this section we review some of the important literature in this area.

Tan et al. [20] propose a web robot session identification method based on their navigational patterns. The main assumption in their proposed system is that web robot navigational patterns such as session length and set of visited webpages (width and depth of visited webpages) is different from those of humans. The aim of their study is on unknown and camouflaged web robots and web crawlers. Park et al. [21] provide a method for malicious web robot detection based on types of requests for web objects (e.g. Cascading Style Sheet files, image files) and existence of mouse/keyboard activity. However, both the above-mentioned studies did not focus on spambot detection in web 2.0 applications where a spambot can mimic human user behaviour.

A proactive spam filtering approach has been proposed by Göbel et al. [22]. Their proposed framework includes interaction with spam botnet controllers which can provide the latest spam messages. Later, it can present a template for current spam runs to improve spam filtering techniques.

Jindal and Liu [12] study opinion spam in review-gathering websites. They propose a machine learning approach based on 36 content-based features to differentiate opinion spam from legitimate opinion. Zinman and Donath [13] attempted to create a model to distinguish spam profiles from legitimate ones in Social Networking Services. Their machine learning based method uses content-based features to do the classification. Benevenuto et al. [16] provide a mechanism to identify video spammers in online social network by means of a Support Vector Machine classifier against content-based features. Heymann et al. [15] survey spam filtering techniques on the social web and evaluate a spam filtering technique on a social tagging system. Most of the above studies focus on one particular type of spam and are limited to the content attributes of that particular domain. Moreover, they do not study the source of the spam problem, i. e. the spambot.

Yu et al.[23] and Yiqun et al. [24] categorise spam webpages from legitimate webpages by employing user web access logs. Their framework relies on user web access logs as a trusted source for classifying webpages.

Last but not least, in HoneySpam 2.0 [2] we proposed our web tracking framework to track spambot data. We develop a framework for accumulating spambot web usage data rather than for detecting spambots.

9. CONCLUSION AND FUTURE WORKS

Research in the area of web spambot detection in Web 2.0 platform is quite young. Most of the current studies have focussed on one particular type of spam rather than a general solution to block spammers. We aim to detect web spambots as a source of spam problems on the Web 2.0 platform. Hence our solution can be extended to other web applications. This paper provides a rule-based on-the-fly web spambot detection method. Our method is based on web usage behaviour. We extract discriminative features called action strings from web usage data to classify spambot vs. human. We propose action as a set of user efforts to achieve certain purposes and action strings as a sequence of actions for a particular user in a transaction. In order to make a real-time and on-the-fly classification method we build a trie data structure based on action strings.

We evaluate our method against an online forum and achieved average accuracy of 93% on spambot detection. We measure the performance of our system by using MCC. The average MCC value of 0.744 is achieved on our 5 randomly selected datasets.

As future work, we intend to improve our detection method and develop an adaptive mechanism to make a real-time update to the system. Additionally, we will evaluate our proposed method against other Web 2.0 platforms such as social networking websites, wikis, and blogs.

10. REFERENCES

[1] P. Hayati and V. Potdar, "Toward Spam 2.0: An Evaluation of Web 2.0 Anti-Spam Methods " in *7th IEEE International Conference on Industrial Informatics* Cardiff, Wales, 2009.

[2] P. Hayati, K. Chai, V. Potdar, and A. Talevski, "HoneySpam 2.0: Profiling Web Spambot Behaviour," in *12th International Conference on Principles of Practise in Multi-Agent Systems*, Nagoya, Japan, 2009, pp. 335-344.

[3] A. Luis von, B. Manuel, and L. John, "Telling humans and computers apart automatically," *Commun. ACM,* vol. 47, pp. 56-60, 2004.
[4] K. Chellapilla and P. Simard, "Using Machine Learning to Break Visual Human Interaction Proofs (HIPs)," in *NIPS*, 2004.
[5] Y. Jeff and A. Ahmad Salah El, "Usability of CAPTCHAs or usability issues in CAPTCHA design," in *Proceedings of the 4th symposium on Usable privacy and security* Pittsburgh, Pennsylvania: ACM, 2008.
[6] F. Edward, "Trie memory," *Commun. ACM,* vol. 3, pp. 490-499, 1960.
[7] R. M. Donald, "PATRICIA\—Practical Algorithm To Retrieve Information Coded in Alphanumeric," *J. ACM,* vol. 15, pp. 514-534, 1968.
[8] M. Kurt, "Compressed tries," *Commun. ACM,* vol. 19, pp. 409-415, 1976.
[9] R. Cooley, B. Mobasher, and J. Srivastava, "Web mining: information and pattern discovery on the World Wide Web," in *Tools with Artificial Intelligence, 1997. Proceedings., Ninth IEEE International Conference on*, 1997, pp. 558-567.
[10] C. Aggarwal, J. L. Wolf, and P. S. Yu, "Caching on the World Wide Web," *Knowledge and Data Engineering, IEEE Transactions on,* vol. 11, pp. 94-107, 1999.
[11] J. Dean and M. R. Henzinger, "Finding related pages in the World Wide Web," *Computer Networks,* vol. 31, pp. 1467-1479, 1999.
[12] J. Nitin and L. Bing, "Opinion spam and analysis," in *Proceedings of the international conference on Web search and web data mining* Palo Alto, California, USA: ACM, 2008.
[13] A. Zinman and J. Donath, "Is Britney Spears spam," in *Fourth Conference on Email and Anti-Spam* Mountain View, California, 2007.
[14] S. Webb, J. Caverlee, and C. Pu, "Social Honeypots: Making Friends with a Spammer Near You," in *Proceedings of the Fifth Conference on Email and Anti-Spam (CEAS 2008)*, Mountain View, CA, 2008.
[15] H. Paul, K. Georgia, and G.-M. Hector, "Fighting Spam on Social Web Sites: A Survey of Approaches and Future Challenges," *IEEE Internet Computing,* vol. 11, pp. 36-45, 2007.
[16] F. Benevenuto, T. Rodrigues, V. Almeida, J. Almeida, C. Zhang, and K. Ross, "Identifying Video Spammers in Online Social Networks," in *AIRWeb '08* Beijing, China, 2008.
[17] Z. Le, Z. Jingbo, and Y. Tianshun, "An evaluation of statistical spam filtering techniques," *ACM Transactions on Asian Language Information Processing (TALIP),* vol. 3, pp. 243-269, 2004.
[18] B. W. Matthews, "Comparison of the predicted and observed secondary structure of T4 phage lysozyme," *Biochim Biophys Acta,* vol. 405, pp. 442-451, 1975.
[19] P. Baldi, S. Brunak, Y. Chauvin, C. A. F. Andersen, and H. Nielsen, "Assessing the accuracy of prediction algorithms for classification: an overview," *Bioinformatics,* vol. 16, pp. 412-424, May 1, 2000 2000.
[20] P.-N. Tan and V. Kumar, "Discovery of Web Robot Sessions Based on their Navigational Patterns," *Data Mining and Knowledge Discovery,* vol. 6, pp. 9-35, 2002.
[21] K. Park, V. S. Pai, K.-W. Lee, and S. Calo, "Securing Web Service by Automatic Robot Detection," *USENIX 2006 Annual Technical Conference Refereed Paper,* 2006.
[22] G. Jan, bel, H. Thorsten, and T. Philipp, "Towards Proactive Spam Filtering (Extended Abstract)," in *Proceedings of the 6th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment* Como, Italy: Springer-Verlag, 2009.
[23] H. Yu, Y. Liu, M. Zhang, L. Ru, and S. Ma, "Web Spam Identification with User Browsing Graph," in *Information Retrieval Technology*, 2009, pp. 38-49.
[24] L. Yiqun, C. Rongwei, Z. Min, M. Shaoping, and R. Liyun, "Identifying web spam with user behavior analysis," in *Proceedings of the 4th international workshop on Adversarial information retrieval on the web* Beijing, China: ACM, 2008