SPAM ANALYSIS AND DETECTION FOR USER GENERATED CONTENT IN ONLINE SOCIAL NETWORKS

DISSERTATION

Presented in Partial Fulfillment of the Requirements for
the Degree Doctor of Philosophy in the
Graduate School of The Ohio State University

By

Enhua Tan, B.Engr., M.Engr.

Graduate Program in Computer Science and Engineering

The Ohio State University

2013

Dissertation Committee:

Xiaodong Zhang, Advisor
Feng Qin
Ten H. (Steve) Lai
ABSTRACT

Recent years have witnessed the success of a number of online social networks (OSNs) and explosive increasing of social media. These social networking and social media sites have attracted a significant number of participants that contribute various types of contents on the Internet, which are generally referred as user generated content (UGC). A well designed UGC network can utilize the wisdom of crowds to collect, organize, and vote user contributed content to generate high quality knowledge with a relatively low cost. However, the open environment of UGC system also makes it easy to be polluted and attacked by spammers and malicious users. How users participate in UGC networks, especially how users contribute content and share content with their friends and other users, is fundamental to spam detection and high quality knowledge discovery. In this dissertation, we investigate two important research issues: (1) discovering user content generation patterns in OSNs, focusing on publicly available content (knowledge sharing), and (2) detecting spam in user generated content based on our discovered patterns.

With the access to three large OSN user activity logs, including Yahoo! Blogs, Yahoo! Answers, and Yahoo! Del.icio.us, for a duration of up to 4.5 years, we are able to well analyze the patterns of content generation patterns of social network users in detail. Our analysis consistently shows that users’ posting behavior in these networks exhibits strong daily and weekly patterns, but the user active time in these OSNs
does not follow commonly assumed exponential distributions. We also show that the user posting behavior in these OSNs follows stretched exponential distributions instead of widely accepted power law distributions. Our discovery lays a foundation for user behavior analysis in social networks, and serves as a ground truth for anomaly detection and anti-spam.

Applying the user posting behavior distribution pattern, we further conducted a comprehensive analysis of spamming activities on a large commercial social blog UGC site in 325 days covering over 6 million posts and nearly 400 thousand users. Observing power law distribution instead of our discovered stretched exponential distribution on user contributions, we find it actually indicates serious UGC spam attack activities. Our analysis shows that UGC spammers exhibit unique non-textual patterns, such as posting activities, advertised spam link metrics, and spam hosting behaviors. Based on these non-textual features, we show with commonly used classification methods that a high detection rate could be achieved offline. These results further motivate us to develop a runtime scheme, BARS, to detect spam posts based on these spamming patterns. The experimental results demonstrate the effectiveness and robustness of BARS.

To timely detect spam in large social network sites, it is desirable to discover self-tuned, unsupervised schemes that can save the training cost of supervised classification schemes. Identifying the limitations of existing unsupervised detection schemes due to assumptions of spammer behaviors that no longer hold, we design an unsupervised spam detection scheme, called UNIK. Instead of picking out spammers directly, UNIK leverages both the connection-based social graph and the content-based user-link graph to remove non-spammers from the network first, and then clusters
spammers with the landing pages they are trying to advertise. Based on highly accurate detection results of UNIK, we further analyze a number of spam campaigns. The result shows that different spammer clusters demonstrate distinct characteristics, implying the ability of UNIK to automatically extract spam signatures.
To my family.
ACKNOWLEDGMENTS

I would like to first thank my advisor Prof. Xiaodong Zhang for his continuous support along these years. Prof. Zhang has a great vision on research topics and is also in the front of industry trends. He always focuses on solving real system problems and making impact to advance technology. With his guidance, I have improved my research ability to complete my PhD study. Prof. Zhang is more than just an advisor: his detail-driven approach and hard work ethic have greatly influenced my work approach and attitude. The role model of Prof. Zhang has been an important part for me to accomplish this milestone in my career path. Prof. Zhang also has given a lot of advices on my presentation skills, which is also vital for me to present this dissertation and any other future work.

Dr. Lei Guo has been my mentor and collaborator from the beginning of my PhD research. Working with him has been my privilege to get access to the most active and critical mind. He has given me very valuable help during my graduate study.

I would also like to thank Prof. Songqing Chen for collaborating with me on a number of research topics. His dedicated and out-of-the-box thinking has been an important resource for my research.

My lab mates Dr. Shansi Ren, Dr. Xiaoning Ding, Dr. Feng Chen, Mr. Tian Luo, Mr. Kaibo Wang, Ms. Shuang Liang, and Dr. Rubao Li have given me the most precious memory of my PhD study, and I owe them a lot for their supports and
friendships. Their talented and constructive discussions have always been useful in the first place.

I would like to thank other committee members of this dissertation: Prof. Feng Qin, Prof. Ten H. (Steve) Lai, for their helpful advices.

In the end, I would like to thank my wife Yujun Zhou for her support all these years. Her encouragement has always inspired me to work hard every day. My parents Ronglong Tan and Huolan Wu have been my safest harbor and role models. I sincerely appreciate their patience on every single bit of me.
VITA

2001 ........................................B.Engr. in Computer Science & Technology, University of Science and Technology of China

2004 ........................................M.Engr. in Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences

2004-2005 ................................. Ph.D. student in Computer Science, College of William and Mary

2006-present ............................... Ph.D. student in Computer Science and Engineering, The Ohio State University

PUBLICATIONS

Research Publications


FIELDS OF STUDY

Major Field: Computer Science and Engineering
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>ii</td>
</tr>
<tr>
<td>Dedication</td>
<td>v</td>
</tr>
<tr>
<td>Acknowledgments</td>
<td>vi</td>
</tr>
<tr>
<td>Vita</td>
<td>viii</td>
</tr>
<tr>
<td>List of Tables</td>
<td>xiv</td>
</tr>
<tr>
<td>List of Figures</td>
<td>xv</td>
</tr>
</tbody>
</table>

Chapters:

1. Introduction                                                         | 1    |
   1.1 Background and Motivation                                         | 1    |
   1.2 Research Contributions                                            | 5    |
       1.2.1 Analysis of User Content Generation Patterns in Social Networks | 5    |
       1.2.2 Analysis and Detection of Spam in User Generated Content     | 6    |
       1.2.3 Social Graph based Unsupervised Spam Detection in User Generated Content | 7    |
   1.3 Organization of the Dissertation                                 | 8    |

2. Analysis of Patterns of User Content Generation in Online Social Networks | 9    |
   2.1 Introduction                                                      | 9    |
   2.2 Related Work                                                      | 13   |
   2.3 User Activity and UGC in OSNs                                    | 15   |
       2.3.1 Overview of User Activity and UGC                            | 16   |
2.3.2 User/content Increase Rate in OSNs .............................................. 17
2.3.3 User Activity Over Time and User Lifetime ................................. 21
2.4 Distribution of User Contributions .................................................. 23
    2.4.1 Original and Non-original UGC ............................................. 23
    2.4.2 Stretched Exponential Distribution of User Contribution ............ 26
    2.4.3 Model Validation .............................................................. 30
2.5 Implications of User Contribution Distributions ............................... 32
    2.5.1 The “80-20” Rule and Top-k Users in OSNs ............................. 32
    2.5.2 Discussions ........................................................................ 35
2.6 Conclusion ..................................................................................... 38

3. Spammer Behavior Analysis and Detection in User Generated Content .... 39
    3.1 Introduction ............................................................................. 39
    3.2 Measurement-based Analysis ..................................................... 42
        3.2.1 Dataset Overview ............................................................. 42
        3.2.2 Weekly and Daily Spamming Patterns ................................. 45
        3.2.3 Spammer Posting Patterns ................................................. 47
        3.2.4 Distribution of Posting Contributions .................................. 49
        3.2.5 Link Patterns .................................................................. 51
        3.2.6 Content Characterizations .................................................. 53
        3.2.7 Hosting Behaviors ............................................................. 54
        3.2.8 Link Spam in UGC Sites ..................................................... 55
    3.3 Offline Spammer Detection ........................................................... 56
        3.3.1 Features and Classifiers ...................................................... 56
        3.3.2 Spammer Classification ...................................................... 59
        3.3.3 Spammer Detection on Entire Dataset .................................. 60
    3.4 BARS: Blacklist-assisted Runtime Spam Detection ......................... 61
        3.4.1 ML & Blacklist Interaction .................................................. 62
        3.4.2 Anti-Detection Prevention .................................................. 64
        3.4.3 User Clustering ................................................................. 64
    3.5 BARS Evaluation ....................................................................... 65
        3.5.1 Performance Evaluation ...................................................... 65
        3.5.2 Anti-Detection Attacks Evaluation ...................................... 68
        3.5.3 Discussions ..................................................................... 70
    3.6 Related Work ........................................................................... 71
    3.7 Conclusion ................................................................................. 73

4. Social Graph based Unsupervised Spam Detection in User Generated Con-
   tent ................................................................................................. 74
    4.1 Introduction ............................................................................. 74
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.2 Motivation</td>
<td>78</td>
</tr>
<tr>
<td>4.2.1 Dataset</td>
<td>78</td>
</tr>
<tr>
<td>4.2.2 Evaluation of AutoRE</td>
<td>80</td>
</tr>
<tr>
<td>4.2.3 Evaluation of FBCluster</td>
<td>81</td>
</tr>
<tr>
<td>4.3 Applying Sybil Defense</td>
<td>82</td>
</tr>
<tr>
<td>4.3.1 SD2: Sybil Defense based Spam Detection</td>
<td>82</td>
</tr>
<tr>
<td>4.3.2 Limitations of SD2</td>
<td>87</td>
</tr>
<tr>
<td>4.4 Design of UNIK</td>
<td>87</td>
</tr>
<tr>
<td>4.4.1 Overview</td>
<td>88</td>
</tr>
<tr>
<td>4.4.2 Generating Whitelist to Trim Edges</td>
<td>90</td>
</tr>
<tr>
<td>4.4.3 Applying the Threshold of Node Degree</td>
<td>91</td>
</tr>
<tr>
<td>4.5 Performance Evaluation</td>
<td>92</td>
</tr>
<tr>
<td>4.5.1 Evaluation of Whitelist and Edge-weight-sum Threshold</td>
<td>94</td>
</tr>
<tr>
<td>4.5.2 UNIK Evaluation</td>
<td>96</td>
</tr>
<tr>
<td>4.5.3 Comparisons with Other Schemes</td>
<td>97</td>
</tr>
<tr>
<td>4.5.4 Social Network Sybil Attack</td>
<td>98</td>
</tr>
<tr>
<td>4.5.5 Legitimate URL Inclusion Attack</td>
<td>99</td>
</tr>
<tr>
<td>4.5.6 Limitations of UNIK</td>
<td>100</td>
</tr>
<tr>
<td>4.6 Spam Cluster Analysis</td>
<td>101</td>
</tr>
<tr>
<td>4.7 Other Related Work</td>
<td>104</td>
</tr>
<tr>
<td>4.8 Conclusion</td>
<td>105</td>
</tr>
<tr>
<td>5. Concluding Remarks</td>
<td>106</td>
</tr>
<tr>
<td>Bibliography</td>
<td>108</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 $\chi^2$ test results ($\alpha = 0.05$)</td>
<td>30</td>
</tr>
<tr>
<td>2.2 Top-$k$ users in OSNs</td>
<td>34</td>
</tr>
<tr>
<td>2.3 Stretched exponential parameters for different UGC contents</td>
<td>36</td>
</tr>
<tr>
<td>3.1 Dataset summary</td>
<td>42</td>
</tr>
<tr>
<td>3.2 Top-20 outgoing link domains</td>
<td>43</td>
</tr>
<tr>
<td>3.3 Content comparison of different spam</td>
<td>54</td>
</tr>
<tr>
<td>3.4 Complete list of features used in evaluations. <em>link-posts</em> represents those posts with link(s) inserted.</td>
<td>58</td>
</tr>
<tr>
<td>3.5 Classification results of different methods</td>
<td>59</td>
</tr>
<tr>
<td>3.6 Classification results on entire dataset</td>
<td>60</td>
</tr>
<tr>
<td>4.1 Statistics of social blog dataset</td>
<td>93</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Daily number of new posts and cumulative number of posts with time</td>
<td>16</td>
</tr>
<tr>
<td>2.2</td>
<td>Weekly and diurnal patterns of UGC posting in OSNs</td>
<td>18</td>
</tr>
<tr>
<td>2.3</td>
<td>Daily number of new users over time in OSNs</td>
<td>19</td>
</tr>
<tr>
<td>2.4</td>
<td>User/post increase rate and overall contribution per user in OSNs</td>
<td>20</td>
</tr>
<tr>
<td>2.5</td>
<td>User activity over time</td>
<td>21</td>
</tr>
<tr>
<td>2.6</td>
<td>The average posting intervals of bloggers</td>
<td>24</td>
</tr>
<tr>
<td>2.7</td>
<td>Weekly/daily patterns of “cut-and-paste” bloggers</td>
<td>24</td>
</tr>
<tr>
<td>2.8</td>
<td>Forwarded posts in different blog categories</td>
<td>25</td>
</tr>
<tr>
<td>2.9</td>
<td>Stretched exponential distribution of user posts in OSNs</td>
<td>27</td>
</tr>
<tr>
<td>2.10</td>
<td>User posts in Blog (users that join system in the same week)</td>
<td>31</td>
</tr>
<tr>
<td>2.11</td>
<td>Cumulative contribution ratio of top users in OSNs</td>
<td>32</td>
</tr>
<tr>
<td>2.12</td>
<td>Contribution of top users in SE and power law networks</td>
<td>33</td>
</tr>
<tr>
<td>2.13</td>
<td>Top user selection</td>
<td>33</td>
</tr>
<tr>
<td>3.1</td>
<td>A spam blog example</td>
<td>44</td>
</tr>
<tr>
<td>3.2</td>
<td>Blog post stats over time (stack graph)</td>
<td>45</td>
</tr>
</tbody>
</table>
3.3 Weekly and daily patterns of XYZ spam and other blog posts . . . . 46
3.4 User activities: for users joined in the same week . . . . . . . . . . . 47
3.5 User posting interval . . . . . . . . . . . . . . . . . . . . . . . . . . 48
3.6 Distribution of posting contributions . . . . . . . . . . . . . . . . . 49
3.7 Ratio of #link-posts per user . . . . . . . . . . . . . . . . . . . . . . 50
3.8 Link patterns . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 52
3.9 Cumulative distribution of content metadata . . . . . . . . . . . . . . 53
3.10 Hosting Behaviors . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 55
3.11 Inlink #domains of sample spam links . . . . . . . . . . . . . . . . . 56
3.12 Classification results using each feature only . . . . . . . . . . . . . . 57
3.13 Runtime spam detection performance . . . . . . . . . . . . . . . . . 66
3.14 Runtime spam detection FP/FN of ML model: posts with history length less or greater than \( T_{\text{history}} = 10 \) . . . . . . . . . . . . . . . . . . 67
3.15 BARS performance: tuning \( T_{\text{history}} \) . . . . . . . . . . . . . . . . . . 68
3.16 Runtime spam detection results: manually inserting non-spam URLs to spam posts . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 69
4.1 Statistics of a social blog dataset . . . . . . . . . . . . . . . . . . . . . 79
4.2 Evaluating AutoRE . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 80
4.3 Conductance of the user graph (social graph + user-link graph) of original dataset . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 86
4.4 Conductance of the user graph with simulated sybil attacks . . . . . 86
4.5 UNIK workflow . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 87
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.6</td>
<td>An example of edge trimming in UNIK</td>
<td>88</td>
</tr>
<tr>
<td>4.7</td>
<td>Whitelist and edge-weight-sum threshold evaluation</td>
<td>94</td>
</tr>
<tr>
<td>4.8</td>
<td>Evaluation of UNIK</td>
<td>96</td>
</tr>
<tr>
<td>4.9</td>
<td>Evaluation of UNIK under attacks</td>
<td>98</td>
</tr>
<tr>
<td>4.10</td>
<td>Spammer cluster size and activity over time</td>
<td>102</td>
</tr>
<tr>
<td>4.11</td>
<td>Spammer cluster posting activity</td>
<td>103</td>
</tr>
<tr>
<td>4.12</td>
<td>Top-4 clusters: host sharing intensity</td>
<td>104</td>
</tr>
</tbody>
</table>
CHAPTER 1

Introduction

1.1 Background and Motivation

Recent years have witnessed the success of a number of online social networks (OSNs) and social media sites, such as blogging, question answering, bookmarking, picture sharing, video sharing, personal networking, and professional networking sites. The typical sites include Blogspot, Twitter, Yahoo! Answers, Flickr, YouTube, MySpace, Facebook, and LinkedIn, to name a few. These social media and social networking sites have attracted a significant number of participants that contribute various types of contents to the Internet, which is generally referred as user generated content (UGC), owing to the pervasive broadband and mobile Internet accesses and the ever-increasing bandwidth available to end users [1]. For example, Facebook has over 1 billion monthly active users, more than 100 petabytes of stored photos and videos [2]. Twitter has over 500 million registered users, and generates over 340 million tweets daily [3]. Yahoo! Answers has over 250 million users worldwide, and close to 1 million questions and answers posted every day [4, 5].

Users are basic elements of these social media and social networking sites. In general, a user’s activities in such networks include authoring content, viewing, and
networking. According to their different purposes, existing OSNs could be classified into two categories, the networking-oriented OSNs and the knowledge-sharing-oriented OSNs. The former, such as Facebook and LinkedIn, emphasizes more on the networking perspective, and the social relationship is the basis of these OSNs. Hence, we call them networking-oriented OSNs. In these OSNs, content sharing is mainly among friends, but many users may also share their content publicly. The latter, such as (micro)blog networks, question/answer networks, and viral video networks, emphasizes more on knowledge or content sharing. Thus, we call them knowledge-sharing-oriented OSNs. The network in these OSNs is not driven by the underlying social relationships. Instead, the network is formed through the users’ common interests on the shared UGC. With the evolution of these networks, today most social media and social networking sites support both content sharing and social relation functions to improve their experiences and enhance their impacts.

UGC, as the main drive of user engagement in social media and social networking sites, has the following merits compared to traditional media and information from authorities and official sites. First, UGC volume is huge comparing with other forms of information dissemination, and often has much larger coverage than traditional media and expert created content. Furthermore, many UGC systems support a built-in mechanism to collect polls or aggregate user opinions to help identify useful and high quality knowledge, thus the user comments or votes themselves are also a source of knowledge, for example, Amazon product reviews. Second, by utilizing the wisdom of crowds [73, 42], a well organized UGC system could provide high quality contents as good as or even better than traditional media content, taking Wikipedia as the best example. Third, the update and propagation of UGC is much faster than traditional
media and other online media, making it an influential information carrier on the Internet. For example, retweet on Twitter reaches 1,000 users on average [48], and it often occurs earlier than media report on breaking news such as an earthquake event. Fourth, UGC is contributed by a huge amount of social network users and is free to be used for the open world. For example, Yahoo! Answers has provided over 1 billion answers for different questions, contributed by end users [5].

Like the dark side of the moon, the open environment of UGC networks also makes the generated content easy to be polluted or attacked by spammers and malicious users. UGC spam in online social networks is explosively increasing and has become an effective vehicle for malware and illegal advertisement distributions. In order to increase the click-through rate, spammers have utilized a number of methods to attract users. By posting spam articles repeatedly to a UGC site, spam content can be shown in the striking positions of the front page on the UGC site, such as the top article list and the most recent article list. By inserting popular terms to the title or the content, spammers can make their posts highly ranked when a user searches these keywords in a UGC system. Spam content not only pollutes the content contributed by normal users, resulting in bad user experiences, but also can mislead or even trap users. Furthermore, spam in UGC sites causes a lot of Internet resources and users’ time being wasted. For example, it has been estimated that 75% posts shown in the top-50 search results for commercial queries at Blogspot.com are actually spam [75]. Another study [60] shows that more than 8% of randomly sampled pages in Blogspot are spam, while many smaller blog sites without strong anti-spam team even have more than 50% spam.
Spamming has been widely studied in email systems, which has also been well controlled in top email service providers such as Yahoo! mail and Gmail. However, UGC spam has its unique characteristics different from email spam. As a result, existing email spam filtering approaches are not effective for UGC spam:

First, currently there is barely any system level protection to prevent spamming in UGC sites, like the firewall of a protected network that only allows SMTP traffic of authorized email servers in order to forbid botnet-based email spam attacks.

Second, spamming in UGC networks is much easier than sending junk emails because a spammer does not need to collect recipient IDs such as the email addresses of the targeted users and does not need to send spam emails with a botnet. Moreover, the audience of UGC can be the general public, online social connections, or members belonging to a group. Therefore, the negative effects of spam in UGC can be much larger, and demanding lower detection latency and higher accuracy.

Third, it is more difficult to get enough training data as machine learning based spam filtering approaches always require high human labor to label the training set. Unlike email systems, which can use user marked spam as the training data, it is hard to do so in a large social network given the large volume of content generation every day without a strong technical anti-spam team. It is highly desirable to avoid or least mitigate the labeling work cost with the common patterns of user content generation.

In this dissertation, we advance the understanding of user content generation in online social networks, and advance the spam detection technology based on these findings. Firstly, we analyze three large online social networks with knowledge sharing, discovering the common patterns of content generation, which provides a baseline to determine how deviated spam content can cause to the content generation.
pattern. Secondly, we identify spammer behaviors in a public social network that is significantly polluted by spam content, and propose methods to detect spammers in runtime. Lastly, we exploit the distinction between non-spammer and spammer behavior patterns to avoid the work of training data preparation for machine learning, and propose an unsupervised spam detection solution.

1.2 Research Contributions

1.2.1 Analysis of User Content Generation Patterns in Social Networks

A number of studies have been conducted to analyze various properties of OSNs, mainly focusing on the formation and evolution of the networks as well as the information propagation over the networks. However, in OSNs with knowledge sharing, such as blogs, Twitter and question/answering networks, issues on how users participate in the network and how users “generate/contribute” knowledge are vital to the success of the network. Similarly, for social networking based OSNs, it is also important to have rich and diversified content to attract users spending more time online. Despite of these, little about UGC patterns has been reported in the research literature. We empirically study workloads with duration of up to 4.5 years from three well recognized knowledge-sharing OSNs, including Yahoo! Blogs, Yahoo! Answers, and Yahoo! Del.icio.us to examine these properties. Our analysis shows (1) the posting behaviors of common users and spammers both exhibit strong but different diurnal patterns. (2) The active time of common users does not follow the widely accepted exponential distribution, and the active time distribution of spammers is quite distinct. (3) We also discover that the posting behavior of common users in these OSNs follows stretched exponential distribution, indicating that the influence
of a small number of core users cannot dominate the network, in contrast to that of spammers.

1.2.2 Analysis and Detection of Spam in User Generated Content

Spam content in social networks is surging with an explosive increase of user generated content on the Internet. Spammers often insert popular keywords or simply copy and paste recent articles from the Web with spam links inserted, attempting to bypass content-based detection. Based on the user content generation patterns we have found, including diurnal activity pattern and the stretched exponential distribution of users’ posting pattern, we have conducted a comprehensive analysis of spamming activities on a large commercial UGC site in 325 days covering over 6 million posts and nearly 400 thousand users. Based on our discovered stretched exponential distribution on user contributions, we find when we observe a power law distribution, it actually indicates that serious UGC spam attacks are going on in this social network. Our analysis shows that UGC spammers exhibit unique non-textual patterns, such as posting activities, advertised spam link metrics, and spam hosting behaviors. Based on these non-textual features, we show with commonly used classification methods that a high detection rate could be achieved offline. These results further motivate us to develop a runtime scheme, BARS, to detect spam posts based on these spamming patterns. The experimental results demonstrate the effectiveness and robustness of BARS.
1.2.3 Social Graph based Unsupervised Spam Detection in User Generated Content

To timely detect spam in large social network sites with least effort, it is desirable to design self-tuned, unsupervised schemes that can quickly adapt to the dynamically changed spamming content and new spamming campaigns, and thus save the training cost of supervised classification schemes. Existing unsupervised schemes heavily rely on bursty spamming behaviors which are no longer hold nowadays as spammers are constantly changing to avoid detection. Thus, the performances of these schemes are often poor with the attack of intelligent spammers in practice. Motivated by our observations that non-spammers and spammers usually form different social relationship clusters, we propose a sybil defense based spam detection scheme, SD2, which remarkably outperforms existing schemes by utilizing the social networking relation among users. In order to make it highly robust in facing more advanced spam attacks, we further design an unsupervised spam detection scheme, called UNIK. Instead of picking out spammers directly, UNIK works by deliberately removing non-spammers from the network and then identify suspicious spammer clusters, leveraging both the social graph and the user-link graph. The underpinning of UNIK is that while spammers constantly change their patterns to evade detection, non-spammers do not have to do so and thus have a relatively non-volatile pattern. UNIK has comparable performance to SD2 when it is applied to a large social network site, and outperforms SD2 significantly when the level of spam attacks increases. Based on detection results of UNIK, we further analyze a number of spam campaigns in this social network
site. The result shows that different spammer clusters demonstrate distinct characteristics, implying the volatility of spamming patterns and the ability of UNIK to automatically extract spam signatures.

1.3 Organization of the Dissertation

The rest of this dissertation is organized as follows. Chapter 2 presents our methodology of analyzing user content generation patterns in social networks and the implications of such patterns. Chapter 3 presents our analysis of spammer behavior patterns, offline spammer classification in UGC, and a runtime system BARS to detect spam in time. Chapter 4 presents the limitations of existing unsupervised spam detection schemes, and proposes two unsupervised schemes SD2 and UNIK based on user clustering behaviors in social networks, with evaluations and implications of these schemes. Chapter 5 concludes this dissertation.
CHAPTER 2

Analysis of Patterns of User Content Generation in Online Social Networks

2.1 Introduction

User Generated Content (UGC), defined as the content contributed by end users, has become pervasive in modern Internet, especially with the wide usage of networking-oriented Online Social Networks (OSNs), such as Facebook [6], LinkedIn [7], and knowledge-sharing-oriented OSNs, such as Yahoo! Answers [4], Twitter [8], Flickr [9].

The rapid development of these OSNs has attracted significant attentions from research community. A number of studies [23, 52, 16, 17, 47] have been conducted to examine various properties of different OSNs. For example, Cheng et al. [23] studied the YouTube videos and found that the links to related videos generated by uploaders’ choices have clear small-world characteristics, which indicates that the videos have strong correlations with each other. In work [52], based on four large online social networks, the authors studied the evolution of social networks and showed that the combination of the gap distribution with the node lifetime leads to a power law out-degree distribution that accurately reflects the real network in all four cases. In
blogspace, researchers in [16, 17] have studied the link propagation and information epidemics.

However, these existing studies mainly focused on how users are connected and thus how the networks are formed, as well as how a social network graph evolves over time, such as [52]. Users who have a large number of connections are the backbone of social networks, and play an important role on information propagation. But in knowledge-sharing-oriented OSNs, how users participate in the network and how users generate and share content play a key role in attracting viewers, since the user participation and contribution in these OSNs drive the growth of these social network communities and the success of their business. Therefore, understanding the patterns of user participation and user posting behavior in these knowledge-sharing-oriented OSNs is very imperative to social network industry and researchers, in order to identify and distinguish active users from spamming users, attract new users and keep existing users, predict hot spots and the trends of topics in user communities, and perform efficient resource management in the underlying supporting system.

The user participation in terms of active time in peer-to-peer and social networks has been assumed to follow exponential distributions in modeling [35, 52], and it has been also assumed that there is strong correlation between user active time and user contribution. It had been reported that in Wikipedia, most of articles are contributed by a small number of users [73]. In [72], Voss found that both the number of Wikipedia articles a user edited and the number of authors for a Wikipedia article follow power law, for different languages. In [42], Kittur et al. analyzed the edit logs of Wikipedia from 2001 to 2006, and find a shift of the user contributions from a core group of elite users to common users, in terms of both number of edits and length of revised contents.
A similar shift of user contribution distribution for Del.icio.us social network was also reported in [42]. However, whether the user contribution in Wikipedia/Del.icio.us follows power law or not is not further analyzed. On the other hand, some studies have been conducted on the distribution of user participations in networking-oriented OSNs. For example, Seshadri et al. studied the mobile phone call graph as a social network and analyzed the distribution of the length of mobile calls [67]. They found it does not follow power law or lognormal. Instead, the double Pareto LogNormal fits the data very well. Gjoka et al. [30] studied the applications on Facebook and found that although the number of application installations increases with time, the average user activity decreases. These findings have put the commonly accepted power law assumption in doubt.

In this work, we set to study the user contributions and activities in knowledge-sharing-oriented OSNs empirically. For this purpose, we have analyzed three workloads of popular OSNs, including Yahoo! Blogs, Yahoo! Answers, and Yahoo! Del.icio.us for a duration of up to 4.5 years. In these OSNs, we are particularly interested in how users participate in the network, generate or post content, as well as the quality of the content. Our analysis shows that

1. User posting behavior in these OSNs shows strong daily and weekly patterns. While active users keep posting at a constant rate, the overall user lifetime does not follow the exponential distribution. Instead, there exist a few users actively participating in the network while some users keep inactive.

2. The distribution of different users’ posting activity does not follow power law distributions. Instead, our analysis across three workloads from both short terms (1 year) and long terms (about 4.5 years) consistently shows that it follows
the stretched exponential distribution, for which the individual contributions of top users are distributed much flatter than those in power law networks.

3. The stretched exponential distribution of user contributions in OSNs follows the “80-20” rule roughly, i.e., 20% users contribute 80% of total content in the network. However, this “80-20” rule of OSNs is significantly different from that in power law phenomena such as Pareto’s social wealth distribution, in that the cumulative contribution ratios of top-\( k \) users are much smaller than those in power law distributions. This indicates that in the OSNs we have studied, a small number of core users cannot dominate the network.

4. User contributions on different types of UGC objects in these OSNs have different characteristics under the stretched exponential model. Typically, the distributions of user contributions on high-quality content or effort-consuming content have a smaller stretch factor, meaning they are more skewed towards high contribution users.

Our results provides timely insights for the current social network industry and research communities. We believe that social networks are rapidly evolving with the advancement of Internet technologies and changing dynamics of user behavior. The analysis of this kind should be conducted from time to time so that we can timely catch the new dynamics of OSNs and respond accordingly.

Our discovery through the workload analysis has many implications and direct applications. In particular, since the highest quality of the content in these knowledge-sharing-oriented OSNs is highly skewed and thus data- or knowledge-mining in these OSNs should focus more on contents generated by these active users. Accordingly,
the hosting site of these OSNs can predict normal resource consumption based on our findings, as well as planning the advertisement placement, filtering spamming contents, dealing with click-fraud, etc.

The remainder of this chapter is organized as follows. We present related works in Section 2.2. In Section 2.3, we overview the social network systems studied in this work. Our detailed workload analysis is conducted in Section 2.4. We further analyze the implications of our findings in Section 2.5 and make concluding remarks in Section 2.6.

2.2 Related Work

Online social networks have attracted considerable attention recently. A number of studies have been conducted on different forms of social networks. For example, as one of the typical social network domain on the Internet, blogspace has constantly attracted researchers’ attention. Earlier work [16, 17, 47] had focused on the link propagation behaviors in blogsphere and studied the information epidemics. While cascading behaviors were rare in [17, 34], Leskovec et al. [55] analyzed about 45 thousands blogs and 2.2 million postings, reported that the size of the cascades distribution follows a perfect Zipf distribution with the slope value of $-2$. Accordingly, a flu-like epidemiological model is proposed to characterize the cascades in blogspace. Gruhl et al. [34] studied the dynamics of information propagation at both topic and individual levels in blogspace. They showed that topics are composed as a union of long term “chatter” topics and short-term “spike” topics. At the individual level, they propose a model for information diffusion based on the spread of infectious diseases [19].
Besides blogs, various other online social networks have also been studied. For example, Guo et al. [77] studied the instant messaging (IM) networks and found that the social network of IM users does not follow a power law distribution; instead, it can be characterized by a Weibull distribution. Leskovec et al. [53] studied Microsoft Messengers and found the average path length among users is 6.6.

The empirical results of social network community structure and evolution have also been reported. For example, Leskovec [54] et al. analyzed about 70 large networks and found that community structure is different from what have been reported, and proposed a “forest fire” generative model to characterize such structures. In work [52], based on four large online social networks, authors studied the evolution of social networks and showed that the combination of the gap distribution with the node lifetime leads to a power law out-degree distribution that accurately reflects the true network in all four cases.

On the other hand, analytical models have also been studied for social networks for a long time. For example, work [65] aimed to find the most influential nodes and build probabilistic models for viral marketing, and work [41] tried to find the most influential nodes in several of the most widely studied models in social network analysis. In [76], authors performed theoretical analysis on information cascades based on random graphs while in [51] authors analyzed topological cascade patterns in a large product recommendation network empirically.

Among these social networks, user activities and the UGC play a key role, particular to knowledge-sharing-oriented OSNs. Work [74] proposed fast algorithms to
characterize bursty patterns of user posting activities in blogspace. Such bursty patterns are often regarded as the result of heavy-tailed dynamics, and power law distributions [24, 72] are often used to characterize such attributes. In P2P networks, Stutzbach and Rejaie have also studied the churns caused by user activities [68].

Since user activity patterns in knowledge-sharing-oriented OSNs have not been well studied despite of their significance, we set to study such patterns and its implications in this work. Our empirical analysis provides several new findings that are different from the commonly accepted concepts about user activities and contributions in these OSNs.

2.3 User Activity and UGC in OSNs

In this study, we analyze three OSN workloads. The Blog workload contains the DB dump of the Yahoo! Blog system of Hong Kong, China, including all posts of articles and photos from 2005-10-26 to 2007-11-20. We will refer them as Blog Article (or Article) and Blog Photo (or Photo), respectively. The Bookmark workload is from the DB dump of Yahoo! Del.icio.us system of U.S., including all bookmarks posted from 2004-01-01 to 2008-07-31. The Answer workload contains the DB dump of Yahoo! Answers system of U.S., including all question and answer posts from 2005-06-26 to 2007-08-27. In our study, we only consider objects posted by users. User comments, such as comments of a blog article by the blogger’s friends, are not considered as UGC posts. Before we present the detailed analysis results, we first present an overview of user posting activities and the posted content in this section.
2.3.1 Overview of User Activity and UGC

In order to study the daily UGC objects generated in a social network, we plot the daily number of new objects and cumulative number of objects with time. As shown in Figures 2.1(a), 2.1(b), 2.1(c), and 2.1(d), in general, the number of daily new posts increases with time sub-linearly, and the cumulative number of all posts increases with time super-linearly. For Bookmark and Answer, the daily number of new posts shows two modes: the upper mode represents the daily posts during weekdays, and the
lower mode represents the daily posts during weekends. To further show these weekly patterns, we study the number of posts in different weekdays and weekends. We bin UGC posts by hours, and count the numbers of posts in the same hours by weeks, for the three OSNs. We then normalize the number of posts in each hour by the total number of posts across the entire trace duration. The upper plots of Figure 2.2 show the normalized weekly posts for Blog Article, Blog Photo, Bookmark, and Answer, respectively. We can see for Bookmark and Answer systems, the hourly number of posts in weekdays is much higher than the hourly number of posts in weekends, causing the “two modes” in the daily new post plot. For Blog systems, the hourly number of posts is similar during weekdays and weekends. This is because blogging is a kind of daily Web journaling or diary writing, so the daily user activities on blog posting do not change dramatically across different days in a week.

The bottom plots of Figure 2.2 further show the daily patterns of UGC posting, computed in a similar way as that of weekly patterns of UGC posting. For all three OSNs, the peak time for posting is about 23:00 in local time. However, the least active time for blog article/photo posting is 6:00, which is different from bookmark/answer posting (12:00-13:00), indicating different user activity patterns in these OSNs.

### 2.3.2 User/content Increase Rate in OSNs

To study the statistics of user contributions in OSNs, a common method is to consider the union of all users and all posts in the system during a measurement period. However, since users may join the system at different time in this period, this method may cause biased distributions of user activities and user contributions. For large scale measurements, the number of new users in this period can be non-trivial
and the joining rate can be bursty. Figure 2.3 shows the daily number of new users joined in Blog and Bookmark systems. In addition to the weekly and daily fluctuating patterns of newly joined users, the user joining events are bursty in daily, weekly, and even a larger time scale. For example, as shown in Figure 2.3(a), around the 280-th day in the trace collection duration, the number of new users is significantly higher than that before and after the 280-th day. For Bookmark, as shown in Figure 2.3(b),
there are a number of spikes, indicating bursts of user joining events spreading over a number of weeks.

Figure 2.4(a) shows the daily increase rate of users and article posts on the left y-axis, and the overall contribution per user on the right y-axis for the Blog system with time. The daily user (article post) increase rate is defined as the ratio of the number of newly joined users (newly posted articles) in a day over the cumulative number of users (articles) by the last day. The overall contribution per user by a day is defined as the ratio of the cumulative number of posts in the system by that day over the cumulative number of users that have joined the system by that day. In general both the post increase rate and the user increase rate decrease with time gradually, meaning the total numbers of both posts and users increase with time super-linearly but not exponentially. Furthermore, the post increase rate is greater than the user increase rate. We can see that both the daily user increase rate and the post increase rate around the 280-th day are greater than those in the neighboring days noticeably. However, the ratio of the post increase rate to the user increase rate in that day is
smaller than those in the neighboring days. This means the burst of these new users have less contributions than the average user in the system. As a result, the overall contribution per user does not increase on that day and the increase becomes linear after that. In contrast, another user increase burst around 570-th day does not affect the overall contribution per user much.

Figure 2.4(b) shows the weekly increase rate of users and posts on the left y-axis, and the overall contribution per user on the right y-axis for the Bookmark system with time. Similar to the user/post increase rate in Blog, the weekly user/post increase rate is defined as the ratio of the number of newly joined users in a week over the cumulative number of users by the last week. Similar to those in the Blog system, both the post increase rate and the user increase rate decrease with time gradually. However, from the figure, we can see that in the week scale, the user increase rate can be greater than the post increase rate, which is very rare in the Blog system. Compared with Figure 2.3(b), we can see the spikes where the user increase rate is greater than the corresponding post increase rate are caused by the bursts of user
joining events, and these joined users have less contributions than common users. Since each of such bursts can last for weeks, the user increase rate actually fluctuates quite big in this time scale, though the general trend is still descending. As a result, as shown in Figure 2.4(b), the overall contribution per user fluctuates in a time scale larger than weeks, and finally increases linearly at a low rate.

### 2.3.3 User Activity Over Time and User Lifetime

Previous studies assume user’s lifetime follows exponential distributions and user activity is uniform over its lifetime or decreases with time exponentially [35, 52]. Understanding user activity over time is important to model the formulation and evolution process of social relationships in a social network, as well as the access and creation traffic of UGC content in a network.

In order to study user posting activity over time, we compute the “author age” of each UGC object in the workloads. The author age of a UGC object is defined as the interval from the time when a user joins the network to the time when the object
is posted. A uniform distribution of author age of UGC object means user activity is uniform over time, while an exponential distribution of author age of UGC posts means user activity decreases with time exponentially.

Since all three OSNs are expanding over time, the daily number of UGC posting increases with time. To avoid biased estimation, we select users that joined the system in the same week, and extract the author age of each post by these users. Figure 2.5(a) shows the CDF distribution of the author age of posts in Blog, Bookmark, and Answer OSNs. For the Bookmark OSN, we can see this distribution is almost uniform, meaning most users’ bookmarking activities do not change over time due to regular Web browsing. For Blog, posts are a little more concentrated on small author ages than Bookmark, but the main body is close to uniform too since many users tend to post blogs regularly. For Answer, the number of posts with large author ages is even smaller, and becomes uniform when the author age of posting is greater than 200 days. Because answering questions is a kind of altruism behavior, a user may become lazy after providing such services for a certain time duration in the Answer system. Even so, the user activities in all these networks are still not exponentially decreasing with time.

Figure 2.5(b) shows the CDF of user’s active duration, the duration from the user joining time to the last user posting time in our traces. The figure indicates that in all three OSNs, there are a few users who either have short active durations or have long active durations (especially for Blog and Bookmark). If we assume a user will not return to the OSN after a long inactive time, a short active duration represents a short user lifetime. This means there are two kinds of users in OSNs: users with short active durations just try the social network system for a short duration and
then never or rarely post later; users with long active durations keep posting with time, leading to the uniform body in the CDF of author age of posts.

Our analysis of user activity over time and user lifetime indicates user’s lifetime does not follow exponential distribution, while user activity is quite uniform over its lifetime. However, the activity frequency of users may vary significantly, which can be characterized through the distribution of user contributions in OSNs. We present its study in the next section.

2.4 Distribution of User Contributions

The workload overview study in the last section provides us some insights on the user activities along time. In this section, we further examine user contributions because in knowledge-sharing-oriented social networks, the content contributed by users is the key to attract viewers and drive the growth of the network.

2.4.1 Original and Non-original UGC

Before we start, we shall clarify that we are more interested in the original content created by users. In general, there are three types of UGC objects in OSNs. The first is the original UGC objects, created by the user who posts them. The second is non-original content obtained through cutting-and-pasting, and the third is advertisements and spams. Since mainly viewers are attracted by the original UGC, we thus focus on the first type for the study. Therefore, first we need to differentiate and filter out the second and the third types of content from the workloads.

Among the three OSNs we study, Blog mainly contains original UGC and non-original content forwarded from other places. On the other hand, we find a small number of bloggers do post advertisements on their blogs. These blogs had been
identified and removed when we built search index for the system. In Bookmark, although a user can post her own bookmark for advertisements with a large number of buzz words as tags in order to be searched and ranked high, our bookmark management system can deal with this in a same way as our search engine for ranking. Similarly, spamming can be found and dealt with. In Answer, since a user can score the answers posted, a person who posts unrelated answers in order to earn credits will be ignored. Thus, we mainly need to filter out non-original content in Blog Article.

Figure 2.6 shows the average posting interval and the total number of posts for each blogger. We can see there are a small number of bloggers who post a large number of articles with small intervals. These bloggers just cut-and-paste entertainment news and political news from the Web to their own blogspaces. We call these blog posts as forwarded posts. Figure 2.7 shows the weekly and daily posting patterns of these “cut-and-paste” bloggers. In contrast to Figure 2.2, the figure shows no concentrated “peak time” for forwarded blog posting, indicating that these users are using “cut-and-paste” at any time in a day. Figure 2.8 further shows the number of such posts
in different article categories (the blog category is selected by the blogger when post). The “recreation” category accounts for most forwarded posts, and the “social events” category ranks the second.

Instead of identifying and removing all forwarded posts, we remove all articles of bloggers whose average posting interval is less than 1/5 day and have posted at least 100 posts altogether, corresponding to a total of 210 users (0.06%) and 1.92% articles.

Since our purpose is to study the distribution of the number of original posts by each
blogger, which is heavy-tailed, ignoring forwarded posts of users who post a small number of articles (in total) shall not affect the distribution much.

### 2.4.2 Stretched Exponential Distribution of User Contribution

Since we are interested in the contribution of each user on the original content in an OSN and the variance among different user’s contributions, it is natural to rank all users according to their contributions and then identify those with high contributions. If we sort each user by the number of posts in descending order, the function of a user’s post number to his/her rank order is called the *rank order distribution function* in the social network. If we normalize the rank order by dividing the total number of users in the social network, then the inverse function of a normalized rank order distribution function is identical to a *complementary cumulative probability distribution function* (CCDF). With the rank distribution, we can focus on those active users who contribute a large amount of high quality UGC content.

The well known Zipf distribution is a rank order distribution, also known as power law. The power law distribution can be expressed as $y_i \propto i^{-\alpha} \ (1 \leq i \leq n)$, where $y_i$ is the value, $i$ is the rank, and $\alpha$ is a constant. The power law distribution has been widely used in characterizing the Internet, WWW, and social networks.

To analyze the user contribution distribution in depth, Figure 2.9 shows the distribution of user posts for six types of UGC objects in these three OSNs. In each figure, the $x$ coordinate represents the reference rank of each user, plotted in log scale, while the $y$ coordinate represents the number of UGC objects posted by this user, plotted in both log scale (marked on the right of $y$-axis) and a powered scale (by a constant $c$, as marked on the left of $y$-axis). We call the combination of log scale in $x$
and powered scale in $y$ as the stretched exponential (SE) scale. Note for Blog Article shown in Figure 2.9(a), the “cut-and-paste” bloggers have been removed. Bookmark Imports in Figure 2.9(d) represents the bookmarks a user imports from her existing bookmark when joining the Bookmark network. Since in the Answer network, an asker can select an answer for her question as the best answer, we plot the best answers that each user contributes in Figure 2.9(f), separated from the overall answers in Figure 2.9(e).

These figures show that in log-log scale, the post rank distributions of users in OSNs have a flat head and a steep tail, which cannot be fitted with a straight line, indicating they are not power law. However, by selecting a proper constant $c$, all these workloads can be well fitted with a straight line in SE scale. The first several
points in Figure 2.9(c) and 2.9(d) are much higher than the line predicts, which is called “King effect” [49]. Such a rank distribution is called a stretched exponential distribution.

The stretched exponential distribution has been used to characterize the access patterns of Internet media traffic [36]. Its corresponding CCDF function is the Weibull function

$$P(X \geq x) = e^{-\left(\frac{x}{x_0}\right)^c},$$  \hspace{1cm} (2.1)

where $c$ and $x_0$ are constants. If we rank the $n$ elements in a data set in descending order of the data value $x_i (1 \leq i \leq n)$, we have $P(X \geq x_i) = i/n$. Substitute $x_i$ for $y_i$, the rank distribution function can be expressed as follows

$$y_i^c = -a \log i + b \hspace{0.5cm} (1 \leq i \leq n),$$ \hspace{1cm} (2.2)

where $a = x_0^c$ and $b = y_1^c$. Thus, the data distribution is a straight line in log-$y^c$ plot. If we assume $y_n = 1$, we have

$$b = 1 + a \log n.$$  \hspace{1cm} (2.3)

To get the parameters of a stretched exponential distribution, we use the maximum likelihood method (MLE): assuming a data set $\{x_1, x_2, ..., x_n\}$ follows some probability distribution with unknown parameters, the most probable parameters are parameters that make the product of the probability density functions of each element in the data set maximum. Denote the parameter vector as $\theta$, then

$$\theta = \arg \max_{\theta} \prod_{i=1}^{n} p_\theta(x_i).$$ \hspace{1cm} (2.4)

The probability density function of a Weibull distribution (stretched exponential) is

$$p(x) = c\frac{x^{c-1}}{x_0^c} e^{-\left(\frac{x}{x_0}\right)^c}.$$ \hspace{1cm} (2.5)
Thus, we have

\[
\begin{align*}
\frac{1}{c} &= \frac{\sum_{i=1}^{n}(y_i^c \log y_i - y_n^c \log y_n)}{\sum_{i=1}^{n}(y_i^c - y_n^c)} - \frac{1}{n} \sum_{i=1}^{n} \log y_i, \\
\frac{a}{n} &= \frac{1}{n} \sum_{i=1}^{n} (y_i^c - y_n^c). 
\end{align*}
\] (2.6)

We first get parameter \( c \) with the iteration method, then we get parameter \( a \). With Equation 2.2, parameter \( b \) can be estimated as

\[
b = \frac{1}{n} \sum_{i=1}^{n} (y_i^c + a \log i). \] (2.7)

However, in our study, the data to be fit (the number of UGC content a user creates) are positive integers, while the random variables in a Weibull distribution are real numbers. Since there is no data element smaller than one, the parameters given by the MLE method above may result in non-trivial errors in the stretched exponential plot. In order to minimize the model fitting errors caused by the discreteness of data values, especially data elements equal to one, an iterative fitting technique is utilized, described in the following.

We use the coefficient of determination of the data fit, also known as \( R^2 \), as an indicator of fitting errors

\[
\begin{align*}
SSE &= \sum_{i=1}^{n} w(i)(y_i - (a \log i + b)^\frac{1}{c})^2, \\
SST &= \sum_{i=1}^{n} w(i)(y_i - \bar{y_i})^2, \\
R^2 &= 1 - \frac{SSE}{SST},
\end{align*}
\] (2.8)

where \( SSE \) is the sum of weighted squares due to errors, \( SST \) is the total sum of weighted squares about the mean, and \( w(i) \) is the weight of data point \( y_i \). Since the stretched exponential fit is conducted in log scale on the x-axis, we select \( w(i) = (\log i)^\prime = 1/i \). The closer \( R^2 \) to 1, the better the model fits the data.
### Table 2.1: $\chi^2$ test results ($\alpha = 0.05$)

<table>
<thead>
<tr>
<th>Data set</th>
<th>$k$</th>
<th>$\chi^2$</th>
<th>$\chi^2_{(\alpha,k-c)}$</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>blog article</td>
<td>11</td>
<td>11.4031</td>
<td>14.067</td>
<td>√</td>
</tr>
<tr>
<td>blog photo</td>
<td>12</td>
<td>14.0723</td>
<td>15.507</td>
<td>√</td>
</tr>
<tr>
<td>bookmark post</td>
<td>10</td>
<td>11.4860</td>
<td>12.592</td>
<td>√</td>
</tr>
<tr>
<td>bookmark import</td>
<td>11</td>
<td>9.3672</td>
<td>14.067</td>
<td>√</td>
</tr>
<tr>
<td>all answer post</td>
<td>11</td>
<td>13.3397</td>
<td>14.067</td>
<td>√</td>
</tr>
<tr>
<td>best answer post</td>
<td>10</td>
<td>7.0005</td>
<td>12.592</td>
<td>√</td>
</tr>
</tbody>
</table>

We iteratively truncate the ranked sequence $\{y_i\} (1 \leq i \leq n)$ by removing the last $k$ elements that equals to one in the sequence, then estimate parameters for the truncated sequence using the MLE method, until the $R^2$ value with the estimated parameters is closest to 1 or larger than a threshold. To avoid bias on the fitting error estimation, all elements in the sequence (including the $k$ elements cut off) are considered when computing $R^2$.

#### 2.4.3 Model Validation

In order to evaluate the goodness-of-fit of the stretched exponential distribution on the data, we conduct Chi-square test as follows. We divide the data value range into $k$ bins ($k \geq 10$) as evenly as possible, with each bin has at least 5 data points (tail bins are merged when necessary). The Chi-square sum is computed as follows

$$\chi^2 = \sum_{i=1}^{k} \frac{(O_i - E_i)^2}{E_i},$$

where $O_i$ is the observed frequency of $i$-th bin, $E_i$ is the expected frequency of $i$-th bin, and $k$ is the number of bins.
Assume the significance level is $\alpha$ (in our test $\alpha = 0.05$), the assumed distribution is rejected when

$$\chi^2 > \chi^2_{(\alpha, k-c)}$$

(2.10)

where $\chi^2_{(\alpha, k-c)}$ is the Chi-square function, $k$ is number of bins and $c$ is the number of distribution parameters plus 1. The results of our test are presented in Table 2.1. All 6 data fittings pass the Chi-square test. We have also tried power law and lognormal fits on these data sets, however, none of them can pass the Chi-square test.

Figure 2.9 includes all users in the entire workload. As we have presented before, the number of users increases with time super-linearly. In order to eliminate the effect caused by users of different ages, we select users who join the system in the same week and study their contributions during an entire year. Our results show that the user contributions are still well fit with SE distributions with nearly the same stretch factor $c$ (examined by Chi-square test). Figure 2.10 shows the fitting results for Blog. Due to page limit, we omit other figures.
2.5 Implications of User Contribution Distributions

The stretched exponential distribution of user contributions has a number of implications that are different from those based on a power law model. We analyze some of these implications in this section.

2.5.1 The “80-20” Rule and Top-$k$ Users in OSNs

Figure 2.11 shows the cumulative contribution ratio of top users (over all contribution by all users) in OSNs. As shown in the figure, the cumulative contribution of top users in the tree OSNs roughly follows the so called “80-20” rule: in Blog, the top 20% users account for 80% posts; in Bookmark, the top 16.5% users account for 83.5% posts; in Answer, the top 13% users account for 87% posts.

However, the “80-20” rule of the stretched exponential distribution is different from that of the power law distribution. Consider the cumulative contribution ratio of top-$k$ users in a power law rank distribution $y_i \propto i^{-\alpha}$ and a stretched exponential
rank distribution $y_i^e = -a \log i + b \ (1 \leq i \leq n \text{ in both distributions})$, denoted as $T_{se}$ and $T_{pow}$, respectively. Figure 2.12 shows the comparison of $T_{se}$ and $T_{pow}$ with log scale in x axis. The parameters of the SE plot are based on the blog article data, while the skewness factor of the power law plot, $\alpha$, is set to 0.9. We can see although the two curves intersect at the “80-20” point, for a small $k$, the cumulative contribution ratio of top $k$ users in a stretched exponential network is much smaller than that in a power law network.

When $n \to \infty$, for a limited value of $k$, we can prove

$$
\frac{T_{se}}{T_{pow}} \approx \lim_{n \to \infty} \frac{k \sum_{i=1}^{k} \frac{1}{\Gamma(1 + \frac{1}{\alpha}) (1 - \alpha \log n)^{\frac{1}{\alpha}}}}{\Gamma \left( \frac{1}{\alpha} \right)} = 0.
$$

(2.11)

The analysis above indicates that in contrast to a power law distribution, a stretched exponential distribution is less skewed, meaning a small number of top users cannot dominate the network as those in power law networks. This can be reflected from the log-log plot of the user contribution rank distribution. As shown in
Table 2.2: Top-$k$ users in OSNs

<table>
<thead>
<tr>
<th>Data set</th>
<th>$a$</th>
<th>$c$</th>
<th>$n$</th>
<th>$k/n$</th>
<th>cumsum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blog Article</td>
<td>2.091</td>
<td>0.418</td>
<td>347,710</td>
<td>0.148</td>
<td>0.7329</td>
</tr>
<tr>
<td>Blog Photo</td>
<td>1.769</td>
<td>0.32</td>
<td>268,837</td>
<td>0.077</td>
<td>0.6397</td>
</tr>
<tr>
<td>Bookmark</td>
<td>2.215</td>
<td>0.34</td>
<td>1,727,353</td>
<td>0.083</td>
<td>0.6762</td>
</tr>
<tr>
<td>Answer</td>
<td>0.980</td>
<td>0.245</td>
<td>10,316,931</td>
<td>0.047</td>
<td>0.6371</td>
</tr>
</tbody>
</table>

the log-log plot of Figure 2.9, the user contribution distribution curve has two modes in general. The first mode is quite flat, corresponding to a small number of top users, where the change rate of user contribution decreases with the change rate of user rank $k$ slowly. The second mode is much steeper, corresponding to the majority of users, where the change rate of contribution decreases with the change rate of rank $k$ significantly.

This observation motivates us to identify those “core” users and the corresponding contributions in a social network with the top-$k$ analysis. For this purpose, we select a rank $k$ so that for users with a rank $\leq k$, the decrease rate of user contribution is smaller than the increase rate of user rank. Thus we have

$$\frac{dy(k)}{y(k)} + \frac{dk}{k} = 0, \text{ i.e., } \frac{d\log y(k)}{d\log k} = -1.$$  \hspace{1cm} (2.12)

Figure 2.13 shows a stretched exponential distribution curve in log-log scale. Line $AB$ with slope $-1$ is tangent to the SE curve at point $(X_0, Y_0)$. Thus, the geometric meaning of $k$ is that $k = \exp(X_0)$, and we can prove

$$\frac{k}{n} = \exp\left(\frac{1}{a} - \frac{1}{c}\right).$$ \hspace{1cm} (2.13)
Table 2.2 shows the number of top-$k$ users for different kinds of UGC objects in OSNs. Column $cumsum$ is the cumulative contribution ratio of total content in the OSN for top-$k$ users. In general about 5% to 15% users can be considered as top users in OSNs. However, the $cumsum$ of top-$k$ users are quite close for all cases, ranging from 63% to 73%. The same method can be applied to estimate top-$k$ articles/videos in content sharing OSNs that follow stretched exponential distributions.

### 2.5.2 Discussions

The “80-20” rule of user contribution in OSNs indicates a small fraction of users contribute most content in the network. However, this metric is quite rough, and cannot reflect the inequality of users’ knowledge contributions in OSNs. The top-$k$ analysis based on stretched exponential model provides a quantitative method to characterize the concentration of UGC contents from different users.

To further understand the inequality of users’ contributions on UGC objects of different types, we conduct the stretched exponential fit on different durations of the three data sets. Our results show that, although different durations of data set have different numbers of users and UGC objects, parameter $c$ is almost a constant for the same social network system and content type across different durations, while parameter $a$ varies for different durations.

We have also studied different classes of objects in the Blog social network, by considering blog articles of different sizes and different numbers of tags attached by the authors. The stretched exponential parameters of different UGC objects are listed in Table 2.3. As shown in the Table, the parameter $c$ for best answer posts is smaller than that of all answer posts, and the parameter $c$ for blog photos is smaller than
Table 2.3: Stretched exponential parameters for different UGC contents

<table>
<thead>
<tr>
<th>UGC workload</th>
<th>c</th>
<th>a</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blog Article (all posts)</td>
<td>0.418</td>
<td>2.091</td>
</tr>
<tr>
<td>Blog Article (with tags)</td>
<td>0.3</td>
<td>0.761</td>
</tr>
<tr>
<td>Blog Article (&gt; 1 KB)</td>
<td>0.39</td>
<td>1.451</td>
</tr>
<tr>
<td>Blog Article (&gt; 2 KB)</td>
<td>0.31</td>
<td>0.702</td>
</tr>
<tr>
<td>Blog photo</td>
<td>0.32</td>
<td>1.730</td>
</tr>
<tr>
<td>Bookmark (imports)</td>
<td>0.33</td>
<td>1.311</td>
</tr>
<tr>
<td>Bookmark (all posts)</td>
<td>0.315</td>
<td>1.685</td>
</tr>
<tr>
<td>Answer (all posts)</td>
<td>0.245</td>
<td>0.980</td>
</tr>
<tr>
<td>Answer (best answers)</td>
<td>0.19</td>
<td>0.404</td>
</tr>
</tbody>
</table>

that of blog articles. In contrast, for Del.icio.us, the parameters $c$ for bookmark posts (bookmarks generated with Del.icio.us plug-in) and imported bookmarks (bookmarks generated with Web browser, which are imported to the Del.icio.us when a user joins the system) are almost the same. We conjecture that parameter $c$ reflects the quality of a UGC object or the effort of creating a UGC object, which may characterize some intrinsic property of UGC objects in social networks: the more effort a user needs to make to create a UGC object, the smaller $c$ is. In the Answer system, the “best answer”s are selected by the user who asked the question, thus the content quality has been judged by the asker herself. If we assume high quality answers needs more effort to create than low quality answers, then the distribution of user contribution for best answer posts, which are effort-consuming, would have smaller $c$ than that for normal answer posts. For Blog system, it is easy to understand that longer articles need more effort to compose than shorter articles, and selecting tags for an article needs extra effort. It is also understandable that composing a short blog article in the Web browser needs less effort than taking a photo, transferring it to a computer,
making some edit work such as rescaling, and then uploading it to one's blogspace with some descriptions. On the other hand, it is straightforward that there is no significant difference between the effort of bookmarking a Web page using a Web browser and that using a Del.icio.us plug-in.

For Wikipedia, the effort of composing a Wikipedia article can be much greater than that of composing a good answer for a user asked question. If we assume that the user contribution in Wikipedia also follows the stretched exponential distribution, then the parameter \( c \) would be much smaller than that of best answer in our study (0.19). When \( c \) is small, we have \( \log y \sim y^c \), and Equation 2.2 becomes power law. Thus, the reported power law distribution of Wikipedia author contribution [72] can be explained.

Our study suggests that the SE distribution can more accurately reflect individual user contributions in these OSNs (and potentially all knowledge-sharing OSNs), which is significantly different from the power law distributions in social networks, such as user’s online connections in IM and email networks and blog in-bound degree. These phenomena are caused by the aggregation effect of multiple users, which can be explained by the “rich-get-richer” or preferential attachment model [52]. However, for the activity of individual users, the lack of aggregated “rich-get-richer” effect implies power law cannot hold, neither can the power law with exponential cutoff model.

The distribution of individual user contribution is the building block to model more complex social network phenomena. Although models have been proposed to describe how user links are created in social networks and how the user networks evolve with time, the process of user link initialization during the evolution is often
oversimplified, as we mentioned in Section 2.3.3. The stretched exponential model can provide in-depth understanding on these social network phenomena.

2.6 Conclusion

Technology advancements have brought up many OSNs on the Internet. Previous studies have mainly focused on the traditional social network problems, such as network formation, connectivity, evolution, and information propagation on these networks. For knowledge-sharing-oriented OSNs, the user activities and contributions are critical. In this work, we have extensively analyzed user activities and contributions in three large OSNs and have revealed several new findings that are different from or contradicting to common assumptions. In particular, the user lifetime in these OSNs does not follow exponential distributions, and the user contribution does not follow power law distributions, but stretched exponential. Our analysis has further shown that even under the stretched exponential distribution, different types of content have different characteristics. Our empirical findings can be leveraged to further understand various OSN properties.

In the next chapter, we use the analysis we have performed as a solid foundation to help mitigate the increasingly severe UGC spam attack problems every OSN faces. We use the user content generation metrics we have found to characterize spamming behavior, and aim to propose a spamming behavior based spam detection solution.
CHAPTER 3

Spammer Behavior Analysis and Detection in User Generated Content

3.1 Introduction

In the last chapter, we have discussed extensively on user content generation in OSNs with knowledge sharing. However, spam in UGC has become a serious challenge to social network daily operation, as it degrades the value of such network, and also leads users to abandon such channel that is full of noise instead of useful information. For example, it has been estimated that 67% of social network users have been spammed in a survey [14] conducted by Sophos.

Spam in UGC happens at every level where the user is able to contribute content. For example, in a social network supporting online group discussions, spammers can create a group whose name advertising products sold by their customers, with discussions only related to that topic too. Any occurrence of such spam group may be indexed by web search engines or the group search engine of the social network, therefore spamming the search results of any keyword spammers put in the group name or discussions. If the social network operators fail to take down the spam group
in time, the social network ranking in search engines is going to be degraded, and the user experience also degrades significantly.

Different from email spamming, spamming in UGC sites is easier to conduct but harder to be detected. First, it is much easier to collect spamming targets for UGC spammers than collecting email addresses on the Web, since UGC sites are easier to be identified with search engines and the number of UGC sites is much smaller than that of email accounts. Second, it is also easier to post a spam article than to send a spam email. Although CAPTCHA [12] is often used for account registration in many UGC sites, posting articles in UGC sites often does not require any CAPTCHA verification. Third, a large number of small UGC sites such as blogs and forums may not have technical teams for anti-spamming. These sites are often the target of spamming attacks. Although most of these sites are not so popular and do not have large user populations, the total number of users and the corresponding audiences of these sites are huge. This is also one reason why UGC spam has increased rapidly in recent years.

Although a content-based spam detection can be effective to some extent, in practice, it has inherent limitations when being applied to UGC sites. First, a content-based classification needs new training data constantly due to the constant change of spam contents. This can be addressed for emails since email recipients often label unrecognized new spam. However, labeling UGC spam by readers is not so effective and accurate due to the open environment of UGC. The large volume of UGC spam makes the human aided labeling very costly. Second, as shown by recent measurements [66], a number of spam blogs are now created by professional spammers who often copy content from recent Web articles or Web sources with specific keywords.
that can help boost spam blog ranking. Thus, it is more difficult for a content-based spam classification method to distinguish spam posts from normal posts as they contain very similar content. Understanding the inherent patterns of UGC spamming behavior can help us to identify more stable, non-volatile features for spam detections.

In this work, first we analyze the trace of a UGC site of a large commercial search engine, which has over 6 million posts involving nearly 400 thousand users in 325 days. Our trace analysis shows that UGC spammers often exhibit uniquely different behavior patterns from those of normal users, including posting patterns, advertised spam links patterns, and link host related patterns. Furthermore, based on our study of spamming behavior patterns, we show that we can achieve low false positive and high true positive rates on detecting spammers with commonly used classification methods offline.

Motivated by these results, we design a runtime spam detection scheme, BARS (Blacklist-assisted Runtime Spam Detection), by leveraging these behavior patterns and a spam URL blacklist. In BARS, a spam classification model is trained with an initial set of labeled spammers and spam URLs. A blacklist of spammers and spam URLs is also initialized with the training set. By feeding only high confident spam URLs from classification to the blacklist when enough posting history information is collected, BARS ensures the high quality of the auto-expanding blacklist. The high quality blacklist is essential to a low false positive rate in runtime detection, while its auto-expanding feature helps improve the true positive rate. Meanwhile, any misclassified users and URLs can be reversed with the help of a high-priority whitelist, which further improves the detection performance. The evaluation results show the promising runtime performance of our scheme.
This chapter is organized as follows. In Section 3.2, we analyze the UGC dataset. We propose the spammer classification features we use and evaluate the dataset with multiple classifiers in Section 3.3. In Section 3.4, BARS scheme is designed and evaluated. Section 3.6 discusses related work and Section 3.7 concludes this chapter.

3.2 Measurement-based Analysis

3.2.1 Dataset Overview

Table 3.1: Dataset summary

<table>
<thead>
<tr>
<th>Type</th>
<th>Number</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>posts</td>
<td>6,595,917</td>
<td>100%</td>
</tr>
<tr>
<td>posts w/ links</td>
<td>1,832,112</td>
<td>27.8%</td>
</tr>
<tr>
<td>XYZ spam posts</td>
<td>1,130,718</td>
<td>17.1%</td>
</tr>
<tr>
<td>user IDs</td>
<td>382,090</td>
<td>100%</td>
</tr>
<tr>
<td>user IDs w/ links</td>
<td>157,305</td>
<td>41.2%</td>
</tr>
<tr>
<td>XYZ spammer IDs</td>
<td>12,116</td>
<td>3.2%</td>
</tr>
</tbody>
</table>

We collected user posts for 325 days in a large commercial blog site till August 2009. Table 3.1 shows a summary of the dataset. The total number of posts is more than 6 millions, of which more than 27% posts include outgoing hyperlinks (or links). The number of user IDs is more than 382 thousands (XYZ is the largest spam campaign).

Table 3.2 lists the top-20 domains of the outgoing links in blog posts, ranked by the number of links. According to the rank of link domains, we find the largest spam campaign, XYZ, in which all posts have links to domains in the form of $xyz*.*$, such as $xyz566.net$, advertising pirated software. XYZ spam accounts for 17.1% of total
Table 3.2: Top-20 outgoing link domains

<table>
<thead>
<tr>
<th>Domains</th>
<th>#Links</th>
<th>IP address</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xyz566.net</td>
<td>20,837,684</td>
<td>111.92.237.40</td>
<td>pirated software</td>
</tr>
<tr>
<td>xyz66.com</td>
<td>5,554,866</td>
<td>74.86.178.68</td>
<td>pirated software</td>
</tr>
<tr>
<td>tv1ccc.com</td>
<td>634,074</td>
<td>218.32.213.235</td>
<td>adult chatroom</td>
</tr>
<tr>
<td>xyz889.com</td>
<td>618,503</td>
<td>74.86.178.68</td>
<td>pirated software</td>
</tr>
<tr>
<td>258ww.com</td>
<td>228,985</td>
<td>220.229.238.55</td>
<td>adult chatroom</td>
</tr>
<tr>
<td>h2.idv.tw</td>
<td>216,058</td>
<td>220.228.6.5</td>
<td>adult chatroom</td>
</tr>
<tr>
<td>h4.idv.tw</td>
<td>213,333</td>
<td>220.228.6.5</td>
<td>adult chatroom</td>
</tr>
<tr>
<td>h5.idv.tw</td>
<td>213,194</td>
<td>220.228.6.5</td>
<td>adult chatroom</td>
</tr>
<tr>
<td>h3.idv.tw</td>
<td>200,001</td>
<td>220.228.6.5</td>
<td>adult chatroom</td>
</tr>
<tr>
<td>ut678.com</td>
<td>160,607</td>
<td>domain expired</td>
<td>adult chatroom</td>
</tr>
<tr>
<td>a5463.com</td>
<td>141,594</td>
<td>220.228.6.140</td>
<td>adult chatroom</td>
</tr>
<tr>
<td>xyz2007.com</td>
<td>140,279</td>
<td>74.86.178.68</td>
<td>pirated software</td>
</tr>
<tr>
<td>s5463.com</td>
<td>127,541</td>
<td>218.32.213.235</td>
<td>adult chatroom</td>
</tr>
<tr>
<td>kk0401.com</td>
<td>123,720</td>
<td>220.228.6.140</td>
<td>adult chatroom</td>
</tr>
<tr>
<td>kk1976.com</td>
<td>114,744</td>
<td>220.228.6.140</td>
<td>adult chatroom</td>
</tr>
<tr>
<td>t1.idv.tw</td>
<td>112,795</td>
<td>220.229.238.3</td>
<td>adult chatroom</td>
</tr>
<tr>
<td>ut789.com</td>
<td>108,787</td>
<td>domain expired</td>
<td>adult chatroom</td>
</tr>
<tr>
<td>youtube.com</td>
<td>105,181</td>
<td>74.125.115.113</td>
<td>video sharing</td>
</tr>
<tr>
<td>photobucket.com</td>
<td>98,156</td>
<td>209.17.65.42</td>
<td>image hosting</td>
</tr>
<tr>
<td>you.cc</td>
<td>93,700</td>
<td>208.109.181.70</td>
<td>domain services</td>
</tr>
</tbody>
</table>
posts in the blog site, but only 3.2% of total user IDs. The domains listed in Table 3.2 are mostly spam domains (note youtube.com is ranked 18th as the most popular non-spam domain). This indicates that most active links posted are dominated by spam sites. Spam in this dataset is mostly involved with pirated software, adult chatroom, etc.

![Figure 3.1: A spam blog example](image)

Figure 3.1 shows an example of a crafty spam blog. This spam blog is trying to embed spam links in the post copied from a news Web site. The spam links have anchors with text keywords like xyz, Reductil, and lv bags, which aim to...
promote the spam sites selling pirated software, counterfeit medicines, and fake luxury products. When the major content of the spam blog is copied from other places, it is difficult to detect spam blogs with only traditional textual features.

![Graph showing daily number of posts and upload traffic](image)

**Figure 3.2: Blog post stats over time (stack graph)**

Figure 3.2 shows the daily number of new posts and upload traffic volume of XYZ spam and other content in a stack graph. As we can observe in Figure 3.2(a), the daily number of XYZ spam posts is non-trivial in the system. Figure 3.2(b) shows that up to 83.7% of the total posting upload traffic is from XYZ spam. The upload traffic of XYZ spam and other posts is calculated based on the content length of posts. These results show the significant resource consumption by spam content in UGC systems.

### 3.2.2 Weekly and Daily Spamming Patterns

We first study the weekly and daily patterns of XYZ spam and compare them with other posts. Figure 3.3 shows the (normalized) number of posts in the time unit of week/day by binning all posts according to their posting hour in a week/day. All
Figure 3.3: Weekly and daily patterns of XYZ spam and other blog posts
the timestamps are extracted from the blog post in the local time zone. Figure 3.3(a) and 3.3(b) show that XYZ spam is posted everyday regardless of whether it is weekend or not, which is not different from other posts. However, Figure 3.3(c) and 3.3(d) show XYZ spam is posted constantly every hour except for the morning (5am to 11am), while others have a daily peak. According to [45], spammers do not have peak hour posting patterns. Our analysis confirms this finding and further reveals that some spammers have an off-hour pattern. We conjecture that these are professional spammers who are paid for posting, and they have their own work pattern.

3.2.3 Spammer Posting Patterns

![User active duration](image1)

![Posting time of a XYZ spammer ID](image2)

(a) User active duration  (b) Posting time of a XYZ spammer ID

**Figure 3.4: User activities: for users joined in the same week**

Because ID revoking is a straightforward method to thwart spam, spammers commonly have many different IDs. In order to examine the impact of spammer IDs, we study the active duration (the duration between the user’s first and last posts) of user IDs in the dataset. We check the set of users who started posting in the same week since the active durations are smaller for newly joined users. Figure 3.4(a)
shows the active duration of users joined in the second week of our trace collection. In this case, XYZ spammer IDs have shorter active durations than those of other users. Figure 3.4(b) shows the posting time of a XYZ spammer ID, which has a clear posting pattern with off-hours and ends after a few weeks. We have also studied spamming behavior for spammer IDs born in a different week. Figure 3.4(a) shows that for XYZ spammer IDs joined in the 7th week, the life span can be as large as our trace duration. After checking these IDs, we find spammers reuse their IDs after a long inactive duration if these IDs are not disabled in time.

![Figure 3.5: User posting interval](image)

Since spammers may advertise spam sites in an aggressive or machine-like manner [56], we thus investigate the posting intervals of XYZ spammers and other users. Figure 3.5(a) shows that a number of XYZ spammers post more frequently than other users, based on the median of user posting intervals. The vertical pattern around 1 minute in Figure 3.5(a) indicates that some spammers post with an almost constant frequency of one post every minute. These spammers exhibit bot-like behaviors, as
most email spammers. On the other hand, these bot-like spammers still have off-hours as shown in Figure 3.4(b). Figure 3.5(b) further shows the MAD (median absolute deviation) distribution of posting intervals. This figure shows that spammers’ posting intervals have smaller variances than those of normal users. In contrast to previous findings [56], there exist non-negligible spammer IDs posting with intervals indistinguishable from those of normal IDs.

3.2.4 Distribution of Posting Contributions

![Figure 3.6: Distribution of posting contributions](image)

In UGC networks, we define the contribution of a user as the number of posts posted by the user. With spam, this distribution could be vastly distorted as shown in Figure 3.6. All Users in Figure 3.6 shows the log-log scale rank distribution of all users’ contributions, and it has an abnormal flat step around the first hundreds of users.
As we have discussed in the last Chapter, user contribution distribution in multiple online social networks can be well characterized as stretched exponential distribution. However, in this social blog dataset which is largely polluted by spam content, the distribution is largely deviated from stretched exponential distribution as the head part is more like a straight line in log-log scale. Some may fit the head as power law distribution, but in fact it is a stretched exponential distribution that is adversely changed by content from very active spammers. Therefore when we observe a power law distribution, it actually indicates that serious UGC spam attacks are going on in this social network.

**Other Users** in Figure 3.6 shows that after removing XYZ spammers, the rank distribution of users’ post numbers is much smoother than before. **XYZ Spammers** in Figure 3.6 shows that the top-200 XYZ spammer IDs post a substantial number of posts, causing the abnormal flattening in the curve for all users.

![Figure 3.7: Ratio of #link-posts per user](image)

A spam article may have some outgoing links embedded in the content, in order to attract readers to click. For each user with links (a user who has at least one
outgoing link in his/her posts), we calculate the number of posts having link(s) inserted (#link-posts) and the total number of posts, then plot the ratio of these two numbers in Figure 3.7. The figure shows that almost all XYZ spammer IDs constantly post articles with links inserted. UGC spammers are not willing to post text-only posts that cannot directly get any clicks to their customers’ Web sites, since a user may not want to copy and paste a text-only URL in a Web browser to access the URL, and it is hard for the customer of spammers to evaluate the effectiveness of spamming without any link.

3.2.5 Link Patterns

As links in spam post are the entrance to the sites advertised by spammers, the presentation of links is usually optimized for their advertising purposes. Figure 3.8(a) shows that XYZ spam posts typically have either 1 or 2 links, which advertise a specific site, or more than 70 links, which advertise a number of items such as different video discs.

Looking into the links, we can see in Figure 3.8(b) that the median URL lengths of XYZ spam links are much shorter. According to our observation, a spam link usually points to a spam site without any path, or to a html file hosted in the root directory, with the intent of redirecting users to click as much content as possible. On the other hand, the link in normal posts is often composed by a query with multiple parameters, or has a long page depth, i.e., to a specific resource on the Web. Figure 3.8(c) shows that the anchor part of a XYZ spam link (the displayed text of a link) also has a shorter length. The analysis shows that normal users often post a link with the same anchor as the URL, while XYZ spammers tend to use shorter
Figure 3.8: Link patterns
keywords to attract user’s attention. Due to the intrinsic link advertising purposes of spammers, there exist substantial differences between spammers and non-spammers on these link metrics, which could be used for spam detection.

3.2.6 Content Characterizations

Figure 3.9: Cumulative distribution of content metadata

Figure 3.9(a) shows the median of post length distribution of users who posted at least one link. XYZ spammers have a slightly larger post length, as some spam posts are copied from Web articles with links or spam content inserted. We also calculate the entropy of the post content by treating it as an ordinary file. Figure 3.9(b) shows that the difference between XYZ spammers and other users is small on this entropy metric.

We further study the content of spam posts by comparing the word frequencies of the most active spam posts. The words are extracted from the sample of manually labeled spam posts (see Section 3.3.2 for details). As shown in Table 3.3, the top-20 most frequently used words of spam type 1 are related to pirated software or movies,
while those of spam type 2 are related to adult chatroom or pornography. There is almost no overlapping between them except film and movie. This indicates that if we only rely on content textual features to detect spam, we have to discover new textual features for every new type of spam, which requires repetitive labeling and training effort.

<table>
<thead>
<tr>
<th>Type</th>
<th>Top-20 Most Frequently Used Words (in English)</th>
</tr>
</thead>
<tbody>
<tr>
<td>spam type 1</td>
<td>official, english, software, DVD, chinese, traditional, disc, film, ULE, xyz, tools, subtitles, price, compilations, sound, DOD, movie, XYZ, CD, TND.</td>
</tr>
<tr>
<td>spam type 2</td>
<td>chatroom, video, adult, dating, beauty, erotic, av, chat, spicegirls, film, picture, movie, free, porn, japanese, game, photo, passion, girls, lover.</td>
</tr>
</tbody>
</table>

3.2.7 Hosting Behaviors

A spam host refers to the Web host in the spam link. In order to defeat blacklist-based spam detection, the host owners often register many host names or even different domain names [27]. Moreover, due to the extra cost to obtain individual IP addresses in Web hosting, lots of spam hosts share IP addresses (Table 3.2 also shows this phenomenon). Thus we expect that for spam hosts, the ratio of unique IP addresses to unique hosts should be small. Figure 3.10(a) shows that more than half of XYZ spammers’ posts point to hosts with less IPs. Figure 3.10(b) plots the ratio of the number of link domains to the number of link hosts. In this case, most XYZ
domains have only one host name. This indicates that the cost-effectiveness consideration of hosting services could in fact serve as an important metric to detect spam advertised hosts.

3.2.8 Link Spam in UGC Sites

Because spam is prevalent on the Web among blogs, forums, or other UGC sites, we quantify the extent of this problem by querying a sample of manually confirmed spam blog links in Yahoo Site Explorer [15], which can return the inlinks of a URL, i.e., the list of Web pages having that URL. Figure 3.11 shows that the queried spam links have a median of 6 to 7 inlink domains (some links have no query results). This implies that a spam link may often be posted to several UGC sites to promote the same spam site. This also suggests that an anti-spam collaboration network among UGC sites can more effectively prevent new spam from being posted in multiple UGC sites.
3.3 Offline Spammer Detection

With the spamming characteristics identified in the previous section, we aim to evaluate their effectiveness with offline classification in this section.

3.3.1 Features and Classifiers

To study the effectiveness of UGC spammer classification by using non-textual features of spamming behavior, we only use the features listed in Table 3.4 in our evaluations. The features we choose are unique in that we collect all posts of a user and use the median or deviation to capture the user’s patterns. Five sets of non-textual features are shown in Table 3.4, including user activities, post contributions, link patterns, hosting behaviors, and content metadata. All these features are selected based on the measurement analysis shown in Section 3.2.

Our classifiers are built based on Orange [13], a python-based data mining library. We have conducted experiments of blog spammer classification with several machine learning classifiers: Naive Bayes (NB), Logistic Regression (LR), and Decision Tree

Figure 3.11: Inlink #domains of sample spam links
Figure 3.12: Classification results using each feature only
Table 3.4: Complete list of features used in evaluations. \textbf{link-posts} represents those posts with link(s) inserted.

<table>
<thead>
<tr>
<th>Feature sets</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>User activities</td>
<td>median (MAD) of posting interval active duration</td>
</tr>
<tr>
<td>Post contributions</td>
<td>#link-posts</td>
</tr>
<tr>
<td></td>
<td>#text-posts</td>
</tr>
<tr>
<td></td>
<td>#link-posts/#posts</td>
</tr>
<tr>
<td>Link patterns</td>
<td>median (MAD) of #URLs</td>
</tr>
<tr>
<td></td>
<td>median of URL length</td>
</tr>
<tr>
<td></td>
<td>median of anchor length</td>
</tr>
<tr>
<td>Hosting behaviors</td>
<td>median (MAD) of #hosts</td>
</tr>
<tr>
<td></td>
<td>median (MAD) of #IPs/#hosts</td>
</tr>
<tr>
<td></td>
<td>median (MAD) of #domains</td>
</tr>
<tr>
<td></td>
<td>median (MAD) of #domains/#hosts</td>
</tr>
<tr>
<td>Content metadata</td>
<td>median of content length</td>
</tr>
<tr>
<td></td>
<td>median of content entropy</td>
</tr>
</tbody>
</table>

(DT). The Naive Bayes method is conducted as the baseline for performance comparison because of its simplicity. Both Naive Bayes and Logistic Regression return the probability of a user being a spammer. The Decision Tree learning method is also conducted in evaluation, with the standard C4.5 algorithm.

We use false positive rate (FPR) and true positive rate (TPR) to evaluate the classification performance. False positive rate is defined as the ratio of false positive items to the sum of true negative items and false positive items (ratio of non-spam classified as spam), and true positive rate is defined as the ratio of true positive items to the sum of true positive items and false negative ones (ratio of spam detected). Another metric false negative rate is defined as $1 - TPR$ (ratio of spam not detected).

$$False\ Positive\ Rate\ (FPR) = \frac{FP}{TN + FP}$$
True Positive Rate (TPR) = \frac{TP}{TP + FN}

3.3.2 Spammer Classification

Table 3.5: Classification results of different methods

<table>
<thead>
<tr>
<th>Methods</th>
<th>FPR</th>
<th>TPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naive Bayes (NB)</td>
<td>6.2%</td>
<td>99.4%</td>
</tr>
<tr>
<td>Logistic Regression (LR)</td>
<td>5.0%</td>
<td>99.5%</td>
</tr>
<tr>
<td>Decision Tree (DT)</td>
<td>1.6%</td>
<td>98.6%</td>
</tr>
</tbody>
</table>

We randomly sampled 1.37% (2,167) users from the total 157,305 users who posted at least one article containing link(s). Then we labeled each sample user as spammer or non-spammer based on the links from their posts. The labeling work was done by two persons without any knowledge of the features used in blog spammer detection. Among these 2,167 users, 1,087 are spammers, which accounts for 50.2% of the sample users. The labeled 2,167 users created a total number of 65,456 posts.

In the first set of experiments, we classify the labeled dataset with different classifiers based on all features shown in Table 3.4. All experiments are performed by using 10-fold cross-validation [43] to avoid biased selections of training and testing sets. Table 3.5 shows the classification results. As shown in the table, Naive Bayes, Logistic Regression, and Decision Tree have comparable performance. Decision Tree has the lowest false positive rate of 1.6% (with 98.6% true positive rate), while Logistic Regression has the highest true positive rate of 99.5% (with 5% false positive rate).
Table 3.6: Classification results on entire dataset

<table>
<thead>
<tr>
<th>Methods</th>
<th>spammer XYZ</th>
<th>non-spammer XYZ</th>
<th>spammer non-XYZ</th>
<th>non-spammer non-XYZ</th>
</tr>
</thead>
<tbody>
<tr>
<td>NB</td>
<td>12,087</td>
<td>70,108</td>
<td>29</td>
<td>75,081</td>
</tr>
<tr>
<td>LR</td>
<td>12,100</td>
<td>69,855</td>
<td>16</td>
<td>75,334</td>
</tr>
<tr>
<td>DT</td>
<td>12,072</td>
<td>67,228</td>
<td>44</td>
<td>77,961</td>
</tr>
</tbody>
</table>

Figure 3.12(a) shows the classification results when each feature is evaluated separately with Decision Tree. Figure 3.12(a) shows that a few features from different feature sets have very good true positive rate with low false positive rate, including the number of domains or hosts (hosting behavior), active duration (user activity), URL length (link pattern), and the number of text-only posts (post contribution). On the other hand, some features only provide limited performance, such as the content length/entropy, as their values are largely overlapping between spammers and non-spammers. Figure 3.12(b) shows the results of Logistic Regression (the results of Naive Bayes are similar and are omitted due to the page limit). Because these features are mostly presented as continuous values, Decision Tree is better at splitting the value space and thus gets better results utilizing each feature.

3.3.3 Spammer Detection on Entire Dataset

Because the entire dataset we collected is too large (over 1.8 million posts with links), it is infeasible to label every user in the dataset. We then use the labeled dataset as the training set, and run the three classifiers on the entire dataset. Because the classifiers do not include the feature of whether the user is a XYZ spammer or not, they could possibly classify XYZ spammers as non-spammers when being applied
to the entire dataset. As shown in Table 3.6, all three classifiers detect almost all XYZ spammers: only up to 44 out of 12,116 XYZ spammers are incorrectly classified as non-spammers. There are about 50% of IDs classified as spammers among the user IDs with link-post(s), and XYZ spammer IDs only account for 15% of them. We randomly sample 1,000 users out of the classified spammers (by Decision Tree classifier), and find only 7 of them are non-spammers, which indicates an estimated 0.7% false positive rate. We also randomly sample 1,000 users out of the classified non-spammers and find 9 of them are spammers, indicating an estimated 0.9% false negative rate. The performance results are consistent with the results on the labeled dataset, showing the effectiveness of spamming behavior features based classifications.

3.4 BARS: Blacklist-assisted Runtime Spam Detection

The result in the previous section shows that spammers can be well detected offline based on non-textual features of all posts of a user. However, for a runtime system, a new spam post needs to be detected right away to minimize its adverse impact. This implies that a runtime spam detection scheme is demanded to classify a new post as soon as it is posted, based on the features of that post and past posts by the same user. In this case, it is challenging to classify the first post of a new user as no history information is available.

In this section, we propose a runtime spam detection scheme BARS (blacklist-assisted runtime spam detection) utilizing non-textual features, with the help of an auto-expanding spam blacklist, and a high priority non-spam whitelist. Our proposed scheme BARS is built on user behavior machine learning (ML) model. The non-textual behavior features are generated at runtime based on the new post and past
posts of the same user. A spam URL blacklist is also maintained to help identify new spam posts. For a new post, if it has a URL in the blacklist, it will be classified as spam. With a highly accurate URL blacklist, spam posts containing these links can be promptly detected without the user’s posting history. By providing high confident spam URLs from the ML model classified results to the blacklist, we can detect more spam than before. Meanwhile, the mis-classified users and URLs in the blacklist are reversed with the help of a high priority whitelist, which is essential for maintaining low false positives. Algorithm 1 shows how BARS works.

3.4.1 ML & Blacklist Interaction

A blacklist is initialized by the training set, and can automatically expand by adding new URLs from a new spam post if the post has any URL in the blacklist. If the post has no URL in the blacklist but can be classified as spam based on spamming behaviors, new URLs from the post are spam URL candidates. In light of how the classification model works, it is intuitive that the classification is likely to be more trustworthy if the user have produced several posts in the system. Therefore, we set a threshold ($T_{history}$) and only provide new URLs of the classified spam post to the blacklist when the number of existing posts by the same user is larger than the threshold. In this way, the blacklist expands with high confidence.

Algorithm 1 details how the blacklist expands with more spam URLs. The classifier can possibly classify a spammer’s first several posts as non-spam due to the lack of user history. After this boosting period, the classifier is finally able to detect this user as a spammer with enough history. Added as a new spammer to the blacklist,
Algorithm 1 BARS: blacklist-assisted runtime spam detection

1: Learner ← ML model training with user behavior features
2: blacklist ← spam URLs in training set (or other sources)
3: blackusr ← spammer IDs in training set
4: whitelist ← non-spam URLs in training set (or other sources)
5: seedwhiteusr ← non-spammer IDs in training set
6: remove URLs in both blacklist and whitelist from the blacklist
7: for a new post do
8: pid ← this new post; uid ← pid’s user ID
9: if pid has URL in blacklist then
10: blSpam ← True
11: else
12: blSpam ← False
13: mlSpam ← Learner(features generated from pid and uid’s past posts)
14: n ← 1 + # of uid’s past posts
15: if blSpam AND not mlSpam AND n > T_{history} then
16: detectSpam ← False
17: else if blSpam OR mlSpam then
18: detectSpam ← True
19: else
20: detectSpam ← False
21: if detectSpam AND (blSpam OR (mlSpam AND n > T_{history})) then
22: newURLs ← pid’s new URLs, i.e. not in whitelist/blacklist
23: if uid not in blackusr then
24: blackusr.add(uid)
25: newURLs.append(new URLs from uid’s past posts)
26: blacklist: include newURLs
27: if not detectSpam AND (uid in seedwhiteusr OR (not mlSpam AND n > T_{history})) then
28: whitelist: include pid’s new URLs
29: remove URLs in both blacklist and whitelist from the blacklist
URLs in past posts of the spammer can now be added to the blacklist to improve the
detection performance of future spam.

3.4.2 Anti-Detection Prevention

In our trace, spam posts rarely contain any non-spam URL. However, to escape
from our spam detection scheme, future spammers may add non-spam URLs to spam
posts, so that the blacklist expanding may incorrectly include non-spam URLs. To
defeat this kind of anti-detection, a high priority steady-growing whitelist containing
non-spam URLs is used in BARS. The whitelist is initialized with non-spam URLs
from the training set, and is updated when a new post is confidently identified as non-
spam (URLs of the user’s past posts are not included to minimize false positives).
The whitelist is set to have a higher priority than the blacklist. When a URL is firstly
included in the blacklist, it can still be removed from the blacklist and inserted to the
whitelist, if the URL repeatedly triggers conflicts between the high confident classifier
and the blacklist detection results. The whitelist is used solely to maintain the low
false positive rate of the blacklist.

3.4.3 User Clustering

In addition, the accuracy of the ML model can also be improved by overcoming
its limitation of not having enough history information for a new user ID. For this
purpose, we can cluster user IDs based on shared URLs. As we know, spammers
typically have multiple user IDs to promote the same spam site. Thus, for the first
post by a new user ID, although it has no posting history, we can incorporate recent
posts of other users sharing the same URLs to generate features for classification.
Besides the improvement to the ML classifier, the blacklist can also get more and accurate input from the ML classification results with user clustering.

3.5 BARS Evaluation

We evaluate our runtime spam detection scheme on the labeled blog posts (refer to Section 3.3.2) by splitting the dataset into training sets and a testing set according to the posting time. The training sets are selected from the latest posts of the first 160 days, with increasing durations. For all training sets, the testing set is selected from the 161st to 325th days, excluding posts by those user IDs existing in the first 160 days. As a result, the testing set is completely independent of the training set, which enables us to assess the efficiency of runtime spam detection.

3.5.1 Performance Evaluation

We compare three schemes in our evaluation experiments. ML uses machine learning classifier only. For a new post, all the features are generated based on the post and past posts of the same user. We use the spamming features listed in Table 3.4 only. The machine learning classifier uses C4.5 Decision Tree. And we also cluster users who share URLs (clustering users who share domains or hosts has a higher false positive rate). BARS uses the algorithm shown in Algorithm 1 based on ML. We set the $T_{\text{history}}$ threshold to 10. OPT provides optimal feedback from the classifier to the blacklist (or whitelist) by filtering non-spam URLs (or spam URLs), which is a cheating algorithm as it requires the labels of the testing set as input. In our evaluation, ML is used as a baseline algorithm, and OPT shows the best result we can get by combining machine learning classifier and the blacklist.
Figure 3.13: Runtime spam detection performance
Figure 3.13(a) and 3.13(b) show the runtime spam detection results of posts in the testing set. Figure 3.13(c) and 3.13(d) show the results of users, which are similar to the results of posts. The false positive rate of ML generally decreases with the increase of the training set duration, while the true positive rate increases accordingly. Comparing ML to BARS, the true positive rate is increased from 91% to 95% while the false positive rate is decreased from 15% to 13%, for the 1 day training set case. That is, the runtime spam detection performance is improved by BARS mainly on the increase of the true positive rate. OPT only outperforms BARS within less than 1% in our evaluation, which further indicates the effectiveness of BARS in detecting spam.

![Figure 3.14: Runtime spam detection FP/FN of ML model: posts with history length less or greater than $T_{\text{history}}=10$](image)

Figure 3.14 shows the FP (false positives) and FN (false negatives) for posts with a history length greater than $T_{\text{history}}$ (GT), and those otherwise (LE) in ML. When the training set duration is larger than 1 day, the false positives for posts with a long history are significantly decreased. This indicates that we can be more confident...
with the classification results for posts with enough history. As a result, the feedback from the classifier to the blacklist is trustworthy, and that is the reason why the performance of BARS is close to OPT. Figure 3.15 shows the spam post detection results of BARS by tuning $T_{history}$. As we can see, the false positive rate is not sensitive to the change of the threshold, and the true positive rate varies within 1% when the training set duration is larger than 5 days. We have similar results of users and are omitted here.

### 3.5.2 Anti-Detection Attacks Evaluation

In order to validate the effectiveness of whitelist, we also evaluate the performance of our schemes by artificially inserting non-spam URLs to spam posts. First, we introduce non-spam URLs (in our labeled dataset) to spam posts after their appearance, as spammers can achieve this by copying links from non-spam posts. The spam detection results of posts in Figure 3.16(a) and 3.16(b) show that BARS performs as well as before due to the steady growth of whitelist (the results of users are similar
Figure 3.16: Runtime spam detection results: manually inserting non-spam URLs to spam posts
and thus omitted). We also evaluate the case of inserting popular non-spam URLs to spam posts before their first appearance in non-spam posts, considering that some popular links can be firstly copied from the Web by spammers. The results in Figure 3.16(c) and 3.16(d) indicate that BARS still performs better than ML with only minor increase of the false positive rate. The reason is that a popular non-spam URL is firstly included in the blacklist, causing some non-spam posts to be detected as spam. With the growth of user history length, BARS detects repetitive conflicts between classifier and blacklist detection results, and then removes the non-spam URL from the blacklist.

In summary, BARS can effectively improve the spam detection performance than using the ML model alone, especially in the small training set case. The blacklist and whitelist well maintain the low false positive rate and increase the true positive rate, even under several anti-detection attacks.

### 3.5.3 Discussions

BARS is sensitive to the history length of a spammer ID. If a spammer resorts to use a new ID each time when posting a new spam, it would be difficult for our scheme. However, the cost of spam posting is also increased for such spammers as the new ID registration requires extra effort. On the other hand, we can cluster user IDs based on shared URLs, which has already shown its effectiveness in our evaluation.

URL shortening has also been increasingly used recently, particularly in microblogs like Twitter. URL shortening, however, does not compromise the URL-based features used in classifications, as well as the blacklist and whitelist. The reason is
that the original URL can be retrieved by visiting the shortened URL, and then be
used for spam detection. The increased cost of retrieving the original URL is trivial.

3.6 Related Work

It is estimated that 78% of emails on the Internet are spam [10]. Plenty of research
has been conducted on email spam. For example, researchers have characterized spam
traffic [32] and network-level spammers’ behavior [63]. Many schemes, such as Naive
Bayes based classifications [18], DNS-based blacklist [39], and domain-based email
authentication methods [26], have been studied or deployed to fight against email
spam. Ramachandran et al. [64] proposed behavior-based blacklist by grouping email
spammers with similar spamming domains. Xie et al. [78] proposed to automatically
extract spam URL patterns from distributed and bursty botnet-based spam cam-
paigns. Hao et al. [38] proposed to detect spammers with network-level features that
are hard to change.

Web spam, especially link spam, that contains a large number of links to boost
the page rank of linked sites in search engines, has drawn significant attention in
recent years. Web spam hosts are discovered based on a small seed set by using the
link structure of the Web to propagate trust [37]. Becchetti et al. were able to detect
80.4% Web spam based only on link properties with 1.1% false positives [20]. Castillo
et al. combined link-based and content-based features for a decision tree classifier and
were able to detect up to 88.4% of spam hosts with 6.3% false positives [22]. Wang et
al. studied Web spam traffic and found that Blogspot.com was responsible for 25%
of Web spam, which served as the doorway domains for Web spammers [75].
The volume of spam increases quickly in UGC sites. For example, Niu et al. [60] found that more than half of blog posts on two blog sites were spam, and forum spam often showed up in major search engines. Kolari et al. [45] have characterized spam blog properties such as non-power-law degree distributions and no-peak daily patterns. Goetz et al. [31] proposed a generative model to produce temporal and topological properties of blog networks, such as the inter-posting time. Sato et al. [66] found most of their studied spam blogs were created by a very small number of professional spammers. These spammers copied spam blogs from recent Web content or Web sources with specific keywords in order to avoid spam detection and promote spam links. Grier et al. [33] studied spam in Twitter and found the click-through rate of Twitter spam is much higher than email spam.

Detecting spam in UGC sites has different challenges from detecting traditional Web spam. Since spam content locates in a single site, it is hard to use the link structure of the Web to help detect such spam. Kolari et al. [44] used SVM to evaluate spam blog detection based on local (content) features such as bag-of-words, bag-of-anchors, and bag-of-urls. In [46], global features, such as incoming or outgoing links to a node, were shown to be less effective than local features on spam blog detection. Lin et al. [56] proposed a spam blog detection method based on temporal, content, and link self-similarity properties. Their results show up to 95% accuracy by combining all the features including traditional content features. Ma et al. demonstrated the effectiveness of host-based features to classify malicious URLs, including the TLD, URL’s path tokens (e.g., ebayisapi, banking), WHOIS dates, and DNS record [57]. Katayama et al. [40] evaluated the impact of sampling confidence to SVM learning for spam blog detection. A recent work [29] on Facebook proposed an unsupervised
algorithm to detect the wall posts of malicious attackers by clustering posts based on shared links. Lee et al. [50] deployed honeypots on MySpace and Twitter and evaluated the effectiveness of social spam signatures generation.

Different from existing schemes, we characterize the spamming behavior patterns at the user level in this work. Because non-textual spamming features do not change as fast as the spam content, it provides a unique opportunity for us to detect UGC spammers. We can utilize these features to improve the runtime spam detection accuracy and robustness.

3.7 Conclusion

The massive volume of user generated content in social media has witnessed the surge of spam in UGC sites, such as spam blogs. Due to the volatility of spam content, we seek to explore the spamming behavior patterns for such spam detection. In this work, we have conducted a thorough analysis of a large blog trace to study the user activities in about one year. Our analysis provides several new findings on the spamming behavior in blog-like UGC sites. Based on these non-textual features, we have applied several classifiers to classify UGC spammers. The experimental results not only show effectiveness of our proposed scheme, but also confirm the features we have identified through analysis. We further design and evaluate a runtime spam detection scheme, BARS, which shows promising detection performance.
CHAPTER 4

Social Graph based Unsupervised Spam Detection in User Generated Content

4.1 Introduction

Spam in online social networks increases quickly because of the viral distribution of information provided by massive social connections on the Internet. The effectiveness of email spam detection also contributes to such a trend. A study [11] shows that email spam has dropped by half in 2010 and spammers are more aggressively targeting social networks and search engines. It is estimated that 67% of social network users have been spammed in a survey [14] conducted by Sophos.

To detect spam in online social networks, many supervised machine learning based methods have been proposed. For example, Lee et al. [50] proposed to deploy honey-pots in social networks, and apply machine learning to detect spam using captured spam as the training set. Benevenuto et al. [21] suggested to detect promoters and spammers in a video social network with user-based, video-based, and social network based features. Markines et al. [58] proposed to use six features at post-, resource-, or user-level to capture spam in a social bookmarking system. However, supervised machine learning methods have some inherent limitations when being applied to a large
social network site. Specifically, labeling the training set is required for supervised learning, which incurs a high human labor cost. Moreover, the labeling work has to be done repetitively to maintain effectiveness for spam detection given the volatility of the spam content and some spam posting patterns. Lastly, the supervised model always lags behind spam attacks with new patterns of spam content.

Different from supervised ones, unsupervised schemes that do not have the training cost have been proposed to detect spam in emails and social networks by directly leveraging the spamming patterns. For example, Xie et al. [78] proposed AutoRE to automatically extract spam URL patterns based on the distributed and bursty patterns of botnet-based email spam campaigns. Applying this approach in detecting social spam, however, may suffer from a high false negative rate since a number of spam posts in social networks are continuously sent over months instead of following the bursty pattern [61]. Gao et al. [29] identified spam by clustering posts based on text and URL similarities and then expecting spam posts to form large clusters with bursty posting patterns. This approach assumes that spam clusters are not connected to non-spam ones. However, spam posts may include non-spam URLs to increase their legitimacy as shown in our data and discussed in [78], which effectively connects spam clusters to non-spam clusters, making it highly difficult to distinguish spam from non-spam. Thereby, it is desirable and imperative to design an unsupervised scheme that can address the limitations of existing schemes.

In this work, we first propose a sybil defense based spam detection scheme SD2. In SD2, a user graph is constructed by combining the social graph and the user-link graph. The former represents the social relationship between active non-spammers,
while the latter characterizes the spam link sharing activity of spammers. Observing that spammers and non-spammers usually form different communities in the user graph, SD2 applies community detection based sybil defense algorithm to the user graph and achieves better spam detection performance than existing schemes. However, because the effectiveness of sybil defense is subject to the spam attack intensity, SD2 does not perform well when the level of attacks increases.

To improve the spam detection performance under an increased level of spam attacks, we further design a new UNsupervised social networK spam detection scheme, called UNIK. Instead of picking out spam directly, UNIK works by capturing the properties of non-spammers in the network first, and then clustering suspicious spammers based on the landing pages they are advertising, leveraging both the social graph and the user-link graph. The underpinning of UNIK is that while spammers constantly change their patterns to evade detection, non-spammers do not have to do so and thus have a relatively non-volatile pattern. UNIK first constructs a user-link graph connecting users who share URL links. Given that a spammer often uses different accounts to post spam URLs, the user-link graph constructed would include almost all spammers in the system, although non-spammers who share URLs are also included. UNIK then constructs the social graph according to the mutual social connections between users, and identifies non-spammers with the help of the social graph. The URLs mainly posted by these identified non-spammers are collected as a URL whitelist, which captures the patterns of non-spam URLs. By trimming non-spam URL edges matching the whitelist in the user-link graph, UNIK isolates a large portion of non-spammers in the user-link graph. Finally, UNIK differentiates spammers
from non-spammers with respect to the node degree in the trimmed user-link graph and detects the majority of spammers.

UNIK is expected to overcome the limitations of two existing unsupervised spam detection schemes [78], [29] and SD2, as UNIK exploits non-spam patterns to detect spam. First, the AutoRE scheme [78] works by detecting spam that is sent with two patterns: distributed and bursty. Correspondingly, spam that is typically posted in the same long duration as normal posts will not be detected. UNIK works by removing non-spammers from the user-link graph which covers most spammers, so it is able to detect most of the spam. Second, the spam clustering scheme [29] relies on the assumption that spam and non-spam posts can be clustered into different groups utilizing the sharing URLs between them. However, as shown in [78], spam content often includes non-spam URLs to increase the legitimacy, which effectively breaks the assumption even if only a handful legitimate URLs are included. UNIK overcomes this limitation by using the non-spam pattern to remove non-spam URLs from the user-link graph, therefore it is robust to the spam attacks with legitimate URLs. Third, SD2 uses a sybil defense algorithm to cluster non-spammers and spammers, whose performance is subject to the spam attack intensity. UNIK identifies non-spam URL signatures based on the social graph and the URL sharing pattern, and then removes non-spam URLs from the user-link graph, thus its effectiveness is maintained in spite of the spam attack intensity since non-spam patterns are largely not affected.

We evaluate the performance of SD2 and UNIK with a 10-month dataset from a commercial social blog site. SD2 shows its superior performance compared to the AutoRE scheme [78] and the spam clustering scheme [29] by reducing both the false positive rate and the false negative rate in detecting spam of this dataset. UNIK also
shows comparable performance to SD2 when being applied to the dataset. Furthermore, UNIK maintains its performance when the spam attack increases, while the performance of SD2 degrades accordingly.

Based on the spam detection result of UNIK, we have identified a number of large spam campaigns in this social blog dataset. Our analysis shows that different spam campaigns demonstrate distinct characteristics. On one hand, this indicates the ineffectiveness of some detection schemes relying on these spamming patterns. On the other hand, this also means that UNIK is able to effectively group spammers into spam campaigns, which provides an opportunity to extract spam signatures from these campaigns and use them to detect spam in other systems.

The rest of the chapter is organized as follows. Section 4.2 evaluates the limitations of existing work and motivates our work. Section 4.3 presents the design, evaluation and analysis of SD2. Section 4.4 illustrates the design of UNIK, and Section 4.5 evaluates its performance. Section 4.6 analyzes the spammer clusters detected by UNIK. Section 4.7 discusses other related work, and Section 4.8 concludes this chapter.

4.2 Motivation

In this section we present the limitations of existing unsupervised spam detection schemes based on a dataset from a large social network site. The results motivated our new designs in this study.

4.2.1 Dataset

We have collected posts and social connections of users for over 10 months from a large commercial social blog site [69]. The number of user IDs who have at least one URL in their posts is more than 176 thousands. These IDs have more than 2 million
posts with URL(s) in the trace collection duration. We developed a supervised machine learning algorithm to detect spammers in this dataset. The spammers detected by the supervised algorithm have both the false positive rate and the false negative rate around 1% to 2%. Because of the accuracy of the supervised algorithm and the infeasibility to label every user in the dataset, we use the results of the supervised algorithm as a ground truth in evaluating the performance of unsupervised spam detection schemes. As we have aforementioned, the good performance of a supervised algorithm is achieved with a price, thus it is still desirable to discover an unsupervised algorithm with similar performance.

Figure 4.1(a) shows the number of new user accounts created every day, stacked with non-spammers on top of spammers. Although the number of spammers is relatively small, some of the spammers posted a number of spam articles to the site as shown in Figure 4.1(b).

![Figure 4.1: Statistics of a social blog dataset](image)

(a) New users created every day  
(b) Cumulative distribution of post number by spammers/non-spammers
4.2.2 Evaluation of AutoRE

We first discuss the performance of AutoRE [78] that detects email spam with distributed and bursty patterns. We applied AutoRE to our social blog dataset by replacing the AS number with the user ID as hinted by [29]. Figure 4.2(a) shows that the suggested 5-day threshold of spam URL active duration has a false negative rate of 26.8%. We further changed the threshold from 5 days to 320 days. As a result, most spam is detected, but the false positive rate is increased to 44.3%. Figure 4.2(a) shows the performance results of AutoRE by tuning the threshold between 5 days and 320 days. Clearly, tuning the threshold cannot help to improve the overall performance.

AutoRE suggests that most spam URLs have a bursty active duration. However, Figure 4.2(b) shows that in our dataset, more than 50% of spam URLs have an active duration of more than 5 days. This finding indicates that if we rely on the bursty pattern of spamming to detect spam, we risk to miss a significant portion of spam that is as active as non-spam [70].

Figure 4.2: Evaluating AutoRE

We first discuss the performance of AutoRE [78] that detects email spam with distributed and bursty patterns. We applied AutoRE to our social blog dataset by replacing the AS number with the user ID as hinted by [29]. Figure 4.2(a) shows that the suggested 5-day threshold of spam URL active duration has a false negative rate of 26.8%. We further changed the threshold from 5 days to 320 days. As a result, most spam is detected, but the false positive rate is increased to 44.3%. Figure 4.2(a) shows the performance results of AutoRE by tuning the threshold between 5 days and 320 days. Clearly, tuning the threshold cannot help to improve the overall performance.

AutoRE suggests that most spam URLs have a bursty active duration. However, Figure 4.2(b) shows that in our dataset, more than 50% of spam URLs have an active duration of more than 5 days. This finding indicates that if we rely on the bursty pattern of spamming to detect spam, we risk to miss a significant portion of spam that is as active as non-spam [70].
Our study shows that the main reason for the performance degradation of AutoRE is due to the change of the spam pattern: lots of spam has been posted with a non-bursty pattern instead of a bursty one.

4.2.3 Evaluation of FBCluster

Gao et al. [29] proposed a scheme, referred to as FBCluster, to detect spam clusters in a social network site. FBCluster constructs a similarity graph of posts on which posts sharing the same URL or similar text are connected. Spam clusters in the graph are detected if a cluster is distributed and bursty, i.e., the cluster is posted by more than 5 user IDs and the median interval of posts is less than 1.5 hours as suggested by FBCluster.

According to the FBCluster scheme, we first construct a URL sharing graph connecting posts sharing the same URL in our dataset. Then we apply the suggested thresholds (5, 1.5 hrs) to detect spam, and we get a false positive rate of 39.3% with a false negative rate of 0.2%. The low false negative rate suggests that FBCluster is able to detect most spam in our dataset as spam URLs are often shared by spam posts. However, the high false positive rate indicates that FBCluster has mistakenly included a number of non-spam posts in the detected spam clusters. We examine the largest cluster in our dataset, and find it has a non-trivial percentage (14.4%) of non-spam posts. Even if we only select the largest cluster as the detected spam cluster, the false positive rate is still as high as 30.1% while the false negative rate increases to 7.0%.

To understand why FBCluster has such a high false positive rate, we further check the URLs that are posted by both spam and non-spam posts. We find there
are about 0.7% of non-spam URLs that are also presented in spam posts. We call this phenomenon the legitimate URL inclusion attack, as spammers include legitimate URLs in their posts to avoid being detected [78]. Although the scale of this attack is small in the dataset, any occurrence of such attacks will effectively connect a spam cluster and a non-spam cluster. Therefore, the reason for the high false positives is that FBCluster is unable to separate spam clusters from non-spam clusters when there exist legitimate URL inclusion attacks. This limitation of FBCluster is rooted from the fact that spammers can intentionally add legitimate URLs to their posts, increasing the difficulty of distinguishing spam from non-spam.

4.3 Applying Sybil Defense

4.3.1 SD2: Sybil Defense based Spam Detection

In the previous section, we have shown the limitations of existing unsupervised schemes in detecting spam. We notice that FBCluster [29] is able to detect the majority of spam in the trace, however, it fails to separate spam clusters from non-spam clusters, and leads to a high false positive rate. This phenomenon is very similar to the sybil attack, in which sybil nodes are connected with non-sybil nodes. This motivates us to investigate sybil defense based schemes [79, 25, 71] to detect spam, given such studies have not been conducted before.

However, directly applying sybil defense schemes for spam detection has several challenges. First, existing sybil defense schemes use the social network graph to identify non-sybil/sybil nodes. As a result, a non-trivial portion of low degree nodes need to be removed in the pre-processing [79, 25] of the social graph, in order to shorten the mixing time [59]. This prevents sybil defense schemes from detecting all
the spam as a number of spammer IDs will be removed in the pre-processing. Second, spammer IDs are not necessarily participating in the social network at all because it is hard for spammers to convince non-spammers to be their friends.

Motivated by our findings, we propose SD2, a Sybil Defense based Spam Detection scheme by using the social graph and a user-link graph that connects users sharing the same URL. SD2 overcomes the problem of FBCluster by effectively separating non-spammers from spammers with the sybil defense scheme. SD2 also includes most spammers for detection by capturing the intensive URL sharing activities among spammers with the user-link graph to make the best use of the sybil defense scheme. Ideally, SD2 only removes inactive/new non-spammers and few spammers in the pre-processing, and then detects non-sybil nodes as the non-spammers and sybil nodes as the spammers, resulting few false positives and few false negatives.

In general, SD2 works as follows. First, a social graph connecting users with mutual social connections is constructed. Second, a user-link graph connecting users sharing URLs is added to the social graph, generating a user graph including almost all users we are interested in. Third, a pre-processing procedure is applied to remove nodes with less than 3 degrees [25]. Fourth, community detection based sybil defense [71] is applied to the user graph to rank nodes with the expectation that non-sybil nodes have higher ranks and sybil nodes have lower ranks. Finally, a cutoff is applied to ranked nodes to identify sybil nodes as spammers.

In SD2, a critical step is to find out the right cutoff point, for which we propose a method based on conductance ratio. For a set of nodes $A$ in a graph, conductance $\gamma$ reflects the community structure of $A$. Define $B = \bar{A}$, or the complement of $A$, then conductance is the number of edges between $A$ and $B$, divided by the sum of node
degrees in $A$ (or $B$ if the number is smaller). If we define $e_{AB}$ as the number of edges between $A$ and $B$, $e_{AA}$ ($e_{BB}$) as the number of edges within $A$ ($B$). Then conductance of $A$ is defined as:

$$\text{conductance}(A) = \frac{e_{AB}}{e_{AB} + 2 \times \min(e_{AA}, e_{BB})}$$

Clearly, conductance indicates the intensity of connections between $A$ and the rest of the graph $B$. Conductance of 0 means strongest community structure (no outside connections), while 1 means weakest community structure (all connections are external).

The community detection based sybil defense algorithm [71] outperforms existing sybil defense approaches by utilizing the conductance metric. The algorithm starts from a trust node or trust node set $s$. New neighbor nodes of $s$ is repeatedly added to the $s$ with the preference of minimizing conductance, until all connected nodes are added. The community detection algorithm ranks nodes by the order of adding to the trust node set $s$, as the nodes ranked higher are more likely to be non-sybil nodes.

In applying the community detection based sybil defense algorithm, SD2 needs to find an optimal cutoff point to separate non-sybil nodes from sybil-nodes. However, there is no suggestion on the cutoff point selection from the algorithm. Therefore, we propose a cutoff method for SD2 based on our observations from the dataset. Figure 4.3 shows the conductance and conductance-ratio values of ranked nodes in the user graph constructed with the dataset. The conductance value is computed when the node is added to the trust node set. The conductance-ratio is computed as the ratio of new conductance to previous conductance upon adding a new node. As shown in Figure 4.3, the conductance-ratio is very close to 1 until it hits the middle
part, where the value starts to vibrate sharply. SD2 thus selects the cutoff point upon the sudden increase of the conductance-ratio. The detail of SD2 is shown by Algorithm 2.

**Algorithm 2 SD2: sybil defense based spam detection**

- **social_graph:** $(V, E)$ where $V$ are users and $E$ are social connections
- **user_link_graph:** $(V, E)$ where $V$ are users and $E$ are shared URLs between $V$

**Require:** user_graph $\leftarrow$ social_graph + user_link_graph

**Ensure:** spammers Detected (spam detected are the posts by spammers Detected)

```plaintext
for v in user_graph.V do
    if v.degree < 3 then
        remove v from user_graph.V
    s ← the most connected node as the trust node
    rankedV ← community_detection(user_graph, s)
    isSybil ← False
    spammers_detected ← list()
for i = 1 to len(rankedV) - 1 do
    if isSybil or rankedV[i].conductance-ratio increases sharply then
        if not isSybil then
            isSybil ← True
        spammers_detected.append(rankedV[i].userID)
```

In terms of spam posts detection, SD2 outperforms existing schemes substantially as shown in Figure 4.8(d): the false positive rate is 2.8% and the false negative rate is 1.4%. If we consider the spammers detection performance, the false positive rate is 0.9% and the false negative rate is 3.0%.

The reason for SD2 to outperform existing schemes is twofold. First, it is able to effectively separate non-spammers from spammers by detecting the community of non-spammers with the social graph. Second, it is able to detect most spammers by including them with the user-link graph, even if most of them are not in the social graph.
Figure 4.3: Conductance of the user graph (social graph + user-link graph) of original dataset

Figure 4.4: Conductance of the user graph with simulated sybil attacks
4.3.2 Limitations of SD2

Although SD2 shows very good performance when it is evaluated with our real world dataset, it does have some limitations. Specifically, the performance of sybil defense in separating non-spammers from spammers degrades when the number of sybil attack increases, which results the degradation of SD2 performance.

Figure 4.4 shows the conductance and conductance-ratio of our dataset with simulated sybil attacks. We randomly select 10% of users in the social graph, and add the same number of spammers connecting to these users, forming a scale free social network among them. As a result, the cutoff point leads to higher false negative rate of 10.0% in terms of spammer detection performance.

4.4 Design of UNIK

Figure 4.5: UNIK workflow

Because the performance of sybil defense based unsupervised spam detection scheme SD2 degrades when there is an increasing level of attacks, in this section,
we further design a new scheme UNIK: UNsupervised socIal networK spam detection. UNIK aims to overcome the limitations of SD2 by exploring the social graph and user-link graph separately.

4.4.1 Overview

Constructing the social graph is straightforward based on mutual relationships between any two users. In addition to that, in an online social network, users post content and URLs. For spammers to promote some spam sites or advertisements, they keep posting the URLs pointing to the spam sites. And they often do this using different user accounts. Thus, UNIK also constructs a separate graph based on the posted URLs by each user in the network. On this graph, users are defined as nodes, and the shared URLs posted by them are defined as edges, and we call it a *user-link graph*.

At a high level, UNIK starts with a user-link graph based on the relationship between users and the URLs posted by them. Instead of directly detecting spammers,
UNIK takes careful steps to remove non-spammers from this user-link graph with high confidence by leveraging both the social graph of social network users and the user-link graph based on the content. Because non-spammers do not have to constantly change their patterns as spammers do to escape from detection, non-spammers have relatively stable patterns. By deliberately removing non-spammers based on their patterns, UNIK is minimally affected by the sybil or legitimate URL inclusion attack by spammers, and achieves a better performance.

The underpinnings of UNIK are twofold. First, non-spammers are the main body of the social network (we assume that if spammers become dominant, the social network is no longer valuable to perform any detection). Second, spammers usually keep posting the same spam URLs with the same or different user accounts. Thus, spam URLs have high occurrences in the content on the social network.

Accordingly, UNIK works as follows. First, it constructs a user-link graph based on the posted URLs by the users, and a social graph based on the mutual relationship of the users. Second, it leverages the social graph to identify non-spammers. A URL whitelist is constructed by identifying the sharing activities of their posted URLs. Third, when user-link graph URL edges are filtered by the URL whitelist, UNIK makes most non-spammers become isolated or low-degree nodes on the user-link graph, which can be removed safely and left spammers node detectable. Figure 4.5 depicts the workflow of UNIK and Figure 4.6 shows an example on the edge trimming in separating non-spammers from spammers.
4.4.2 Generating Whitelist to Trim Edges

UNIK first tries to identify non-spammers by applying the SD2 algorithm to the social graph only. In contrast, the standalone SD2 algorithm detects spammers by using the combination of the social graph and the user-link graph, as the latter is required to connect spammers. Since non-spammers are mostly active in the social graph, we do not need to combine the two graphs in this step.

As shown in the last section, SD2 might have a non-trivial false negative rate in detecting spammers upon a sybil attack to the social graph. Therefore, the identified non-spammers are not 100% correct. Fortunately, UNIK manages to tolerate the error in the identified non-spammers list as shown in follows.

Based on the identified non-spammers list, UNIK generates a whitelist covering the URLs in identified non-spammers’ posts, so that more non-spammers can be identified with the whitelist. However, if a spam URL is incorrectly included in the whitelist, the spammers sharing this spam URL may be removed from the user-link graph, causing false negatives. This situation is even worse for those widely shared URLs. To deal with such situations, UNIK uses the identified non-spammers and the user-link graph to help detect such spam URLs. For a URL shared among users, we have more confidence to include it on the whitelist if more than half of the users sharing this URL are non-spammers. That is, UNIK requires shared URLs to meet this condition to avoid inclusion of spam URLs in the whitelist. Note that the whitelist can be built based on domains or hosts, other than the URL itself, because a wider coverage of the whitelist is able to decrease false positives while errors in the whitelist only lead to the increase of false negatives.
Based on the generated whitelist, UNIK examines the edges on the user-link graph. Shared URL edges in the user-link graph are trimmed if they match the whitelist. After these removals, non-spammers who only share whitelisted URLs become isolated on the user-link graph because all their edges are trimmed. Thus, they can be removed, and the remaining nodes on the user-link graph are mostly spammers, with a limited number of non-spammers whose edges are mostly trimmed.

4.4.3 Applying the Threshold of Node Degree

The trimmed user-link graph may include some non-spammers because the whitelist is not likely to cover every non-spam URL. To further improve the detection performance, UNIK aims to remove as many non-spammers as possible based on the URL sharing properties of spammers. Typically, the URL sharing activities of spammers are much more intensive than non-spammers in terms of the number of shared URLs or the number of sharing users. This means that a spammer node often has more edges than a non-spammer node in the user-link graph. UNIK thus examines the node degree, and detects users whose degree is beyond a threshold as spammers.

To compute the node degree on the trimmed user-link graph, intuitively, edge(s) should exist between every two users sharing URL(s), and the edge weight is set to 1. This however has a time complexity of $O(n^2)$ as all the users sharing a URL are fully connected. To reduce the processing time, instead, for each shared URL, UNIK organizes the users sharing the URL into a circular node list, and only adds two edges to a node, one connecting the node to its preceding node and the other connecting it to its succeeding node, with each edge weight set as half of the number of other
sharing users. By the increase of the edge weight, for each shared URL, the sum of edge weight of a node increases with the same amount as in a fully connected graph, which is the number of other sharing users. In this way, the user-link graph is quickly constructed with a linear number of edges connecting nodes, while the sum of edge weights of each node $\sum_{\text{shared-URLs}} \text{other-sharing-users}$ is exactly the same as the node degree in a fully connected graph $\sum_{\text{other-sharing-users}} \text{shared-URLs}$. With the edge weight, UNIK applies a heuristic threshold on the sum of edge weights, below which the connected nodes are deemed as non-spammers and get removed. In the end, only spammers exist on the user-link graph, possibly with few non-spammers who have intensive URL sharing activities as well.

The threshold of the node degree or the sum of edge weights needs to be determined beforehand for UNIK to work. In the next section, we will show that such a threshold can be independently determined (Figure 4.7(b)). For the best spam detection result, this threshold can be estimated by gauging a small sample of spammers comparing to non-spammers from time to time.

The UNIK spam detection algorithm is shown by Algorithm 3.

4.5 Performance Evaluation

We have evaluated the unsupervised spam detection scheme UNIK with our dataset shown in Section 4.2.1. Table 4.1 shows the statistics of the social graph and user-link graph (constructed with a linear number of edges) built with the dataset.

As UNIK applies whitelist and edge-weight-sum threshold to detect spammers, both of which can have different choices and both of which can be used independently,
Algorithm 3 UNIK spam detection

Require: user_link_graph: (V, E) where V are users and E are shared URLs between V (linearly added)

Require: social_graph: (V, E) where V are users and E are social connections

Require: edge_weight_sum threshold $w$

Ensure: spammers_detected (spam_detected are the posts by spammers_detected)

for v in social_graph.V do
    if v.degree < 3 then
        remove v from social_graph.V

WhiteUser ← social_graph.V - SD2(social_graph)
whitelist ← set()
for v in WhiteUser do
    for url in v.posts do
        if url not in whitelist and len(sharing_users(url) in WhiteUser) >= len(sharing_users(url)) * 0.5 then
            whitelist.add(url)
for e in user_link_graph.E do
    if e matches whitelist then
        remove e from user_link_graph.E
for v in user_link_graph.V do
    if v has no edges then
        remove v from user_link_graph.V
spammers_detected ← list()
for v in user_link_graph.V do
    if sum_edge_weights(v) > w then
        spammers_detected.append(v)

Table 4.1: Statistics of social blog dataset

<table>
<thead>
<tr>
<th>Type</th>
<th>Nodes</th>
<th>Edges</th>
<th>Avg. Degree</th>
<th>Spammers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social Graph</td>
<td>141,619</td>
<td>474,701</td>
<td>6.7</td>
<td>206</td>
</tr>
<tr>
<td>User-link Graph</td>
<td>176,692</td>
<td>578,862</td>
<td>6.6</td>
<td>79,344</td>
</tr>
</tbody>
</table>

93
we first evaluate their different choices individually in order to find the best suitable ones.

4.5.1 Evaluation of Whitelist and Edge-weight-sum Threshold

![Figure 4.7: Whitelist and edge-weight-sum threshold evaluation](image)

We first evaluate the spammer detection effectiveness by only applying different types of whitelist, including URL-based, host-based, and domain-based whitelist. We also evaluate Host+1Path based whitelist, where the whitelist matches the hostname and the first path in the URL. For example, if the whitelist pattern is http://a.com/service, then URLs like http://a.com/service/1.html or http://a.com/service?q=1 will be matched. Host+1Path whitelist is a hybrid between URL-based and host-based whitelist. Lastly, we evaluate the approach of not using whitelist to trim edges, but only removing identified non-spammer nodes, namely the WhiteUser approach.
Figure 4.7(a) shows the spammer detection results after applying the whitelist or WhiteUser only. Because WhiteUser only covers identified non-spammers and the URL-based whitelist has the narrowest coverage of non-spam URLs, the user-link graph still has a number of non-spammers remaining, which results the highest false positive rate, but the lowest false negative rate as most spammers are still remaining. The domain-based whitelist and the host-based whitelist further decrease the false positive rate, since the coverage of non-spammers increases. However, any errors in the whitelist may result in a higher false negative rate as the spammers will be removed from the graph. The \texttt{Host+1Path} based whitelist outperforms the URL-based whitelist in terms of the false positive rate, and its false negative rate is also smaller than that of the host-based whitelist. Figure 4.7(a) also shows that if we include all URL hosts of identified non-spammers in the whitelist (all-host-based whitelist), the false negative rate increases substantially. This indicates that it is necessary to check whether the URL is mainly shared among identified non-spammers to avoid the inclusion of spam link in the whitelist.

We then evaluate applying the edge-weight-sum threshold alone on the original user-link graph to detect spammers. The threshold increases exponentially (base 2) from 1 to 16,384, and nodes are detected as spammers with the sum of their edge weights larger than or equal to the threshold. Figure 4.7(b) shows that applying this threshold alone could detect spammers with a false positive rate of 10.7% and a false negative rate of 3.6% when the threshold is set to 256. Although the performance of applying the threshold alone is not satisfactory, it does show that using this threshold can help improve the detection performance.
4.5.2 UNIK Evaluation

We now evaluate the effectiveness of UNIK with the help of the whitelist and the edge-weight-sum threshold together. Figure 4.8(a) shows the spammer detection results. The Host+1Path based whitelist detects spammers with a false positive rate of 0.6% and a false negative rate of 3.7% when the edge-weight-sum threshold is 256. The host-based whitelist has a false positive rate of 0.5% and a false negative rate of 4.0% with the same threshold. In contrast, WhiteUser and the all-host-based whitelist
show worse performance. For WhiteUser, only non-spammers identified based on the social graph are removed from the user-link graph, and no edges are further removed for other non-spammers. As a result, applying the edge-weight-sum threshold can only help improve the performance to a limited extent.

Figure 4.8(b) shows the distribution of the sum of edge weights for spammers and non-spammers in the user-link graph using the host-based whitelist. Most spammers have a sum of more than 64, while more than 70% of non-spammers have a sum less than 64 before the edge trimming. After the edges are trimmed by the whitelist, more than 80% of non-spammers have a sum less than 64. This means the whitelist could help further differentiate spammers with non-spammers when applying the edge-weight-sum threshold. Note that although the threshold cutoff still marks 20% of non-spammers in the user-link graph as spammers, the small number of non-spammers remaining in the graph results in a small number of false positives in the spam detection. Figure 4.8(c) shows that if we apply the edge-weight-sum threshold earlier before applying the whitelist, the performance is worse than applying the threshold after applying the whitelist. This further validates that trimming edges by the whitelist in advance is helpful to applying the edge-weight-sum threshold in detecting spammers. On the other hand, WhiteUser does not trim any edge, so it is indifferent to applying the threshold earlier or later.

4.5.3 Comparisons with Other Schemes

Our UNIK scheme can detect spammers with a false positive rate of 0.6% and a false negative rate of 3.7%. In terms of the spam post being detected, the false positive rate is 3.7% and the false negative rate is 1.0% as shown in Figure 4.8(d).
This suggests that UNIK is able to achieve the same level of detection performance as SD2. Figure 4.8(d) also shows that FBCluster [29] has high false positives. When we apply the suggested (5, 1.5 hrs) thresholds, the false positive rate is 39.3% while the false negative rate is 0.2%. We also have evaluated the AutoRE algorithm used in [78] by replacing the AS with the user account, which however has a 26.8% false negative rate when applying the suggested 20-AS and 5-day thresholds. Section 4.2 shows that tuning the threshold for FBCluster and AutoRE still cannot improve their performance.

4.5.4 Social Network Sybil Attack

Although UNIK works well in the evaluations with our dataset, its effectiveness is still subject to other possible enhanced attacks launched by spammers. To investigate the robustness of UNIK under attacks, we first evaluate UNIK performance by launching sybil attacks to the social graph of our dataset. To do so, we randomly
select 10% of users in the social graph, and add the same number of spammers connecting to these users, forming a scale free social network among them. Then we double the number of spammers connected to the social network.

Figure 4.9(a) shows the impact of this sybil attack on UNIK spammer detection performance. We observe that the host-based whitelist is subject to sybil attacks: the false negative rate increases substantially when the sybil attack increases. This is because an increased level of sybil attacks increases errors in generating the host-based whitelist, trimming corresponding spam URL edges, and resulting in false negatives. However, if the whitelist is based on $\text{Host+1Path}$, then the errors are limited to URLs strictly matching the prefixes, which effectively limits the false negatives. In facing such attacks in the social graph, we should choose to limit the whitelist coverage for better performance. The false positive rate is 0.6% and the false negative rate is 4.3% for $\text{Host+1Path}$ based whitelist with the edge-weight-sum threshold of 256, when the sybil attack intensity is 20% of social network users. This indicates that UNIK overcomes the limitation of SD2 when sybil attacks increase in the social network.

4.5.5 Legitimate URL Inclusion Attack

Similar to the sybil attack occurred in the social graph, spammers can include legitimate URL in the spam so that they can escape from being detected. Such inclusion increases the connectivity of the spammers and non-spammers in the user-link graph. For this kind of attacks, spammers need to crawl legitimate URLs of non-spammers in the user-link graph, which is often limited in crawling speed or scope.
We simulate legitimate URL inclusion attack to our dataset by assuming the spammers have made the effort crawling a major portion of total legitimate URLs in the user-link graph, and have inserted the crawled legitimate URLs into every spam posts. Since the total number of legitimate URLs is much larger than that of spam URLs, each spam post in this simulated attack only contains a minor fraction of spam URLs while the majority of URLs are legitimate.

Figure 4.9(b) shows the evaluation results of UNIK under such attacks. For a legitimate URL being attacked, it will not be included in the whitelist, as the majority of sharing users of that URL are mainly spammers after the attack. However, the host or Host+1Path prefix of the URL may still be included in the whitelist, because it is unlikely that other URLs with the prefix are all attacked. Even if the URL prefix is not included in the whitelist, the edges of attacked non-spammers will still be trimmed by the whitelist if not all URLs are attacked. As shown in Figure 4.9(b), the host-based whitelist shows slightly better performance due to its wider coverage which is harder to be defeated by the attack. The Host+1Path based whitelist still has a false positive rate of 2.7% and a false negative rate of 1.9% (applying the edge-weight-sum threshold of 256) even if 50% of legitimate URLs are attacked by spammers. In summary, UNIK works well under legitimate URL inclusion attacks for different kinds of whitelist.

4.5.6 Limitations of UNIK

We have shown the promising performance of UNIK evaluated with our social blog dataset. And we also have shown the stable performance of UNIK facing difference scales of spam attacks. However, because UNIK is based on applying sybil defense
scheme to detect spammers, UNIK has limitations in facing some types of spam attacks. For example, we have seen reports from [29] and [33] that a high percentage of spammers are using compromised user accounts to post spam in private social networks. The reason is that in private social networks, users primarily interact with a small list of other users, so spammers have to compromise normal users’ accounts to widely promote their spam content. In this case, UNIK will have trouble to detect such compromised accounts as spammers since they are an integral part of the social graph. Therefore, UNIK is more suitable for fighting spam in an open social network such as groups, forums, or blogs, where the spamming activities are more intensive.

UNIK also needs to address the issue of URL shortening that is widely used in Twitter-like social networks. In facing shortened URLs, UNIK may need to fetch the final destination redirected by such a URL, so that the user-link graph can correctly represent the link sharing activities of different users. This increases the complexity of the system implementation of UNIK by introducing the cost of resolving URL redirects. In practice, we have seen systems incorporated such mechanisms in fighting social spam with reasonable cost [70].

4.6 Spam Cluster Analysis

Based on the UNIK scheme presented in the last section, we are able to group spammers into different clusters in the user-link graph in our dataset. Each of these spammer clusters represents a group of spammer accounts that are interconnected with shared URLs in their posts, corresponding to different spam campaigns. Studying the spammer clusters can enable us to understand the patterns of different spam campaigns, and can also help develop spam signatures in fighting against future spam.
We first plot the number of spammer cluster sizes in Figure 4.10(a). Interestingly, the cluster size distribution follows power law, indicating that spam campaigns also have a natural distribution on their sizes. We are interested in the largest clusters, so we pick the top-4 clusters to study their characteristics. The top-4 clusters have 63699, 6634, 3159, and 724 spammer IDs, respectively. Figure 4.10(b) shows the number of new accounts created over time for each of the top-4 clusters. We observe that in #1 cluster new accounts were continuously created over the trace duration, while new accounts were created at different periods in other clusters. This clearly indicates that different spammer clusters have different activities patterns. There exists some correlation between the #1 cluster and the #3 cluster as their ID creation activities are both bursty around day 50.

![Cluster size distribution](image1)

**Figure 4.10:** Spammer cluster size and activity over time

We also study the user activities of each spammer ID in the top-4 clusters. Figure 4.11(a) shows the median of posting interval for each user in the top-4 clusters. The #4 cluster demonstrates the shortest interval distribution: more than 90% IDs
have a median of interval within 1 minute. On the contrary, the #3 cluster shows the longest interval distribution: most IDs have a median of interval larger than 40 days. Figure 4.11(b) shows the active duration of each ID in the top-4 spammer clusters. The #4 cluster shows a very short active duration while the #2 cluster shows a median duration of 100-day per user ID. The active duration distribution of #3 cluster shows three discretely stages including 0, 50, 137 days. The significant differences between different clusters imply that the behavior of spammer ID varies substantially. Therefore it is highly difficult to capture the spammer behavior with only a handful patterns.

Figure 4.11: Spammer cluster posting activity

Figure 4.12 shows the sharing intensity of each host in the top-4 clusters. The sharing intensity of a host is defined as the number of users that ever posted link(s) pointing to the host, which captures the intensity of advertising activities of a spam host. The #2 cluster shows the strongest sharing intensity of the advertised spam hosts, while the other clusters show similar intensity distributions. This is correlated
with the longest active duration distribution of # 2 cluster as shown in Figure 4.11(b). This finding implies that the host sharing intensity is increased with the increase of spamming activity by spammers over time.

4.7 Other Related Work

Spam in online social networks has been actively studied through measurement-based analysis recently. For example, Thomas et al. [70] proposed a real-time system Monarch to detect spam based on URL-related features, and they found Twitter spam campaigns are long lasting than email spam campaigns. Grier et al. [33] analyzed spam in Twitter with public blacklists, and showed that 8% of all URLs are on popular blacklists, although the blacklists are too slow in preventing the damage of harmful URLs. Gao et al. [28] proposed social network features such as social degree for online spam filtering based on Facebook and Twitter datasets.

Several other approaches have been proposed to detect spam in emails with spam templates, network-level features, shared IP addresses, or email target domains. Qian et al. [62] proposed an unsupervised learning based email spam filter since spam
from the same campaign often contains unchanged textual snippets by using the same template. Hao et al. [38] studied using network-level features to distinguish email spammers from non-spammers as the first level defense since it is difficult to maintain IP-based blacklists. BotGraph [80] evaluated a graph-based approach with the MapReduce model to cluster millions of bot-users observing that bot-users share IP addresses to log in or send emails. SpamTracker [64] tried to catch email spammers earlier than traditional IP-based blacklists do by using the target domains that a IP address sends emails to as a behavioral feature to cluster IP addresses.

4.8 Conclusion

Albeit a number of spam detection schemes have been designed, spam in online social networks increases significantly in recent years. In this study, we first analyze the limitations of existing representative schemes, and then design a sybil defense based spam detection scheme, SD2. SD2 is an unsupervised scheme, which outperforms existing unsupervised schemes significantly with the help of the social network graph. But it suffers from escalating sybil attacks to the social network. Thus, we further propose UNIK that is highly robust to an increased level of spam attacks. UNIK differs from existing schemes in that it detects non-spam URL patterns from the social network instead of spam URLs directly, because the non-spammer patterns are relatively more stable than spammers. UNIK demonstrates its performance with a 10-month social network dataset. Based on UNIK detection, we have also analyzed spammer clusters in the dataset, and find distinct spamming patterns of different spam campaigns, indicating the volatility of spamming patterns.
CHAPTER 5

Concluding Remarks

Technology advancements have brought up many Online Social Networks (OSNs) on the Internet. With the massive increasing user generated content (UGC) in these networks, recent years have witnessed the surge of spam in social media. Today, spam detection of UGC in social networks remains to be a challenge problem due to the heterogeneous complexity in distributed network systems, the huge impact of UGC propagation, and the enormous volume of UGC in social networks. However, previous social network studies have mainly focused on the social graphs themselves of these networks, such as network formation, connectivity, evolution, and information propagation on these networks, while most anti-spamming solutions are based on information retrieval of the social content where social graphs are seldom used. In this dissertation, we studied the user contribution and spamming in online social networks, and addressed two important research problems: (1) discover user content generation patterns, and (2) apply discovered patterns to improve spam detection in user generated content.

We have analyzed user activities, especially user contributions, in three large OSNs. Our results have shown new findings that are different from or contradicting to common assumptions. We find user posting contributions follow stretched
exponential distribution, instead of previously assumed power law distribution. This baseline pattern enables us to determine whether a UGC site is seriously attacked by spam content. We also find the daily and weekly patterns of user activity and non-exponential user active time distribution. These patterns form the foundation of our spam detection study.

We have conducted an analysis of a large social blog trace to compare the normal user and spammer activities over about one year. Based on the comparison study, we first propose an offline classification scheme that detects spammers relying on their non-textual patterns, such as posting patterns, spam link patterns, and spam host patterns. We further propose a runtime spam detection scheme, BARS, to improve the spam detection performance of a runtime system. Our evaluations show the effectiveness and robustness of BARS with real world trace and enhanced spam attack trace.

To solve the inherent training cost problem in traditional and widely used supervised classification schemes, we also propose an unsupervised scheme called UNIK based on user clustering behaviors in social networks, and the advertising pattern of spam content. By utilizing the connection-based social graph and the content-based user-link graph at the same time, UNIK solves the high false positive or high false negative issues in existing unsupervised schemes. UNIK also withstands an increased level of spam attack on the social connection graph or the content link graph. We finally discuss the implications of applying UNIK to analyze spam campaigns. Our study shows user behavior based approaches on social spam detection in UGC systems are highly effective with little training effort.


