COMBATING THREATS TO THE QUALITY OF INFORMATION IN SOCIAL SYSTEMS

A Dissertation

by

KYUMIN LEE

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	James Caverlee
Committee Members,	Frank Shipman
	Yoonsuck Choe
	Takashi Yamauchi
Head of Department,	Hank Walker

August 2013

Major Subject: Computer Science

ABSTRACT

Many large-scale social systems such as Web-based social networks, online social media sites and Web-scale crowdsourcing systems have been growing rapidly, enabling millions of human participants to generate, share and consume content on a massive scale. This reliance on users can lead to many positive effects, including large-scale growth in the size and content in the community, bottom-up discovery of "citizen-experts", serendipitous discovery of new resources beyond the scope of the system designers, and new social-based information search and retrieval algorithms. But the relative openness and reliance on users coupled with the widespread interest and growth of these social systems carries risks and raises growing concerns over the quality of information in these systems.

In this dissertation research, we focus on countering threats to the quality of information in self-managing social systems. Concretely, we identify three classes of threats to these systems: (i) content pollution by social spammers, (ii) coordinated campaigns for strategic manipulation, and (iii) threats to collective attention. To combat these threats, we propose three inter-related methods for detecting evidence of these threats, mitigating their impact, and improving the quality of information in social systems. We augment this three-fold defense with an exploration of their origins in "crowdturfing" – a sinister counterpart to the enormous positive opportunities of crowdsourcing. In particular, this dissertation research makes four unique contributions:

- The first contribution of this dissertation research is a framework for detecting and filtering social spammers and content polluters in social systems. To detect and filter individual social spammers and content polluters, we propose and

evaluate a novel social honeypot-based approach.

- Second, we present a set of methods and algorithms for detecting coordinated campaigns in large-scale social systems. We propose and evaluate a content-driven framework for effectively linking free text posts with common "talking points" and extracting campaigns from large-scale social systems.

- Third, we present a dual study of the robustness of social systems to *collective attention* threats through both a data-driven modeling approach and deployment over a real system trace. We evaluate the effectiveness of countermeasures deployed based on the first moments of a bursting phenomenon in a real system.

- Finally, we study the underlying ecosystem of crowdturfing for engaging in each of the three threat types. We present a framework for "pulling back the curtain" on crowdturfers to reveal their underlying ecosystem on both crowdsourcing sites and social media.

DEDICATION

For my father, mother and Kyungran

ACKNOWLEDGEMENTS

To complete Ph.D. process successfully, I believe that the support from three groups of people – family, mentors and colleagues – is necessary and important. Fortunately, I have received warm encouragement, advice and support from the three groups during my Ph.D. process. I would like to express my thanks and gratitude to all of them.

My wife Kyungran Park is my best friend and life partner. My first Ph.D. year was the hardest time and I had a lot of concerns and doubts. With her encouragement and support, I have been able to be optimistic and courageous, and overcome obstacles. I am so lucky to have the best parents – Jungyoul Lee and Eonkwang Shin – who have been dedicated to me during their entire life. Even though they have lived in another country, I have felt that they have been living just next to me because they have often called me and chanted for my health and success. No matter what happened to me, they have stayed in my side. Seeing the growth of my children – Yejin and Yongjin – is always joyful. I love all of my family members including my sister Kyuok and brother Kyusun.

The biggest benefit during my Ph.D. process was meeting James Caverlee who has been and will forever be my advisor, teacher, and sometimes warm brother having a big heart. I still clearly remember the first moment when I met him in his IR class in Spring 2009. He had not only intellectual knowledge of the subject, but also humor and good personality. I hoped that he would be my advisor and fortunately he allowed me to work with him. Because of his teaching, encouragement and support, I was able to complete my PhD successfully. I was also admired for his unlimited support and advices regarding my personal issues. His good behaviors and attitudes

as an educator has inspired me to become an educator teaching the future generation and conducting research for a better society. Again, I would like to express thanks to him.

I am grateful to the members of the Infolab. Thanks to Zhiyuan Cheng as an office-mate for discussing several research topics, studying them together and being my close friend. Thanks to Krishna Kamath, Brian Eoff, Prithivi Tamilarasan for collaborating on research work with me. I will remember my senior lab-mates Elham Khabiri, Said Kashoob and Jeremy Kelley and other lab-mates Jeff McGee, Chiao-Fang Hsu, Anupam Aggarwal, Yuan Liang, Amir Fayazi, Vandana Bachini, Haokai Lu, Wei Niu, Hancheng Ge and Cheng Cao.

I would like to express thanks to Nish Parikh and Neel Sundaresan for giving me a research opportunity at eBay Research Labs, and to Jalal Mahmud, Jilin Chen and Jeffrey Nichols for giving me a research opportunity at IBM Almaden Research Center. I am indebted to Caverlee, Frank Shipman, Yoonsuck Choe and Takashi Yamauchi for reading and commenting on my dissertation.

TABLE OF CONTENTS

Page

ABSTRACT ii

DEDICATION iv

ACKNOWLEDGEMENTS v

TABLE OF CONTENTS vii

LIST OF FIGURES x

LIST OF TABLES xiv

1. INTRODUCTION 1

1.1 Motivation 1
1.2 Research Challenges 2
1.3 Overview of this Dissertation 3

2. RELATED WORK 9

2.1 Email and Web 9
2.2 Social Networks 10
2.3 Crowdsourcing Sites 12

3. DETECTING SOCIAL SPAMMERS AND CONTENT POLLUTERS . . 13

3.1 Introduction 13
3.2 Uncovering Social Spammers 15
3.2.1 Overall Framework 15
3.2.2 RC1: Study of Harvested Spam Users 19
3.2.3 RC2: Empirical Evaluation of Social Spam Signals 22
3.2.4 RC3: Large-Scale Social Spam Classification In the Wild . . . 32
3.3 A Long-Term Study of Content Polluters on Twitter 38
3.3.1 Introduction 38
3.3.2 Tempting Content Polluters 39
3.3.3 Who are the Harvested Twitter Users? 39
3.3.4 Profiling Content Polluters 45
3.3.5 Features 48
3.4 Summary 57

4. EXTRACTING COORDINATED CAMPAIGNS 58

4.1 Introduction 58
4.2 Content-Driven Campaign Detection 60
4.2.1 Problem Statement 60
4.2.2 Data 61
4.2.3 Metrics 62
4.3 Campaign Detection: Framework and Methods 64
4.3.1 Overall Approach 64
4.3.2 Message Level Campaign Detection 66
4.3.3 User Level Campaign Detection 74
4.3.4 MapReduce-Based Implementation 76
4.4 Experimental Study 77
4.4.1 Message Level 78
4.4.2 User Level 91
4.4.3 Temporal Analysis 96
4.4.4 Summary 98
4.5 Summary 99

5. COMBATING THREATS TO COLLECTIVE ATTENTION 100

5.1 Introduction 100
5.1.1 Our Approach 105
5.1.2 Summary of Key Findings 106
5.2 A Data-Driven Model for Studying Collective Attention Threats . . . 107
5.2.1 System Model 107
5.2.2 Posting Model 107
5.2.3 Collective Attention Access Models 109
5.2.4 Measuring Spam Impact 111
5.2.5 Seeding and Validating the Model 111
5.2.6 Threats from Individual Spammers 116
5.2.7 Threats from Coordinated Spammers 117
5.2.8 Countermeasures 118
5.3 Countermeasure Deployment on Twitter 121
5.3.1 Metrics 122
5.3.2 Countermeasure 1: Rule-Based Filtering 123
5.3.3 Countermeasure 2: Supervised Classification 125
5.3.4 Combining Countermeasures 130
5.3.5 Summary and Discussion 133
5.4 Summary 135

6. TRACKING AND REVEALING CROWDSOURCED MANIPULATION . 137

6.1 Introduction 137
6.2 Analysis of Crowdturfing Tasks and Participants 139

6.2.1 Types of Crowdturfing Campaigns . . . 139
6.2.2 Requesters and Workers . . . 141
6.3 Down the Rabbit Hole: Linking Crowdsourcing Workers to Social Media . . . 144
6.3.1 Following Crowd Workers onto Twitter . . . 145
6.3.2 Analysis of Twitter Workers: By Profile, Activity, and Linguistic Characteristics . . . 146
6.3.3 Network Structure of Twitter Workers . . . 151
6.3.4 Digging Deeper: Middlemen . . . 157
6.4 Summary . . . 160

7. CONCLUSIONS AND FUTURE WORK . . . 161

7.1 Conclusion . . . 161
7.2 Future Research Opportunities . . . 163

REFERENCES . . . 165

LIST OF FIGURES

FIGURE Page

1.1 Overall Dissertation Organization 4

3.1 Overall Framework of Social Honeypot-based Approach 16

3.2 MySpace – Feature Comparison 25

3.3 Twitter – Feature Comparison 29

3.4 Cumulative Distribution of Features Extracted from Users 30

3.5 MySpace – Spam Precision 34

3.6 Twitter – Spam Precision 36

3.7 An example of newly found spammer on Twitter 38

3.8 A chart of the number of content polluters tempted per day. On the fourth day of the study the honeypots were able to tempt a total of 391 content polluters, the most in a single day. The third highest single-day temptation was 339, which occurred on day 191. 40

3.9 The Twitter homepage of gmhomebiz, a user tempted by our social honeypots 43

3.10 The graph shows the changing number of users following the *gmhomebiz* account and the number of users followed. 44

3.11 The top two graphs are of content polluter accounts. The bottom two are legitimate users. The accounts in the top two graphs are engaging in the act of "follower churn" [139]. 45

3.12 "@spam" search results 54

4.1 Overall approach showing how to identify campaigns given a list of messages 65

4.2 In a messages graph, a node represents a message and there exists an edge between correlated messages. This figure shows an example of a message graph before extracting campaigns. 69

4.3 Four matches of correlated messages between user u_1 and u_2 74

4.4 Logical data flow of the three MapReduce jobs for identifying correlated messages 77

4.5 Performance of Shingling, I-Match and SpotSigs with different parameter value 80

4.6 This figure depicts the distribution of the size of candidate campaigns on a log-log scale. It follows a power law. 86

4.7 This figure depicts a candidate with 61,691 vertices. A blue dot and a black line represent a vertex and an edge, respectively. The area in the center is dark because most vertices in the center are very densely connected. 87

4.8 An example dense subgraph campaign: the center area is dark because vertices in the area are very densely connected; this subgraph is almost fully connected except a few vertices. While strict campaign detection identifies 5 different maximal cliques, cohesive campaign detection identifies a single coherent campaign including all vertices. . . 88

4.9 This figure depicts the distribution of the size of cohesive campaigns on a log-log scale. It also follows a power law. 89

4.10 303 candidate campaigns in the user-aggregated message graph 92

4.11 Campaign type distribution (threshold = 3.8) 93

4.12 Campaign type distribution (threshold = 9.5) 95

4.13 Average temporal graphs of four campaign categories 97

5.1 Example YouTube video designed to capitalize on collective interest during and immediately after the London Olympics Opening Ceremony 101

5.2 Spam messages targeting the Twitter trending topic “Glen Rice” . . . 102

5.3 Three spam approaches. Collective attention spam relies on the users themselves to seek out the content where the spam will be encountered. 103

5.4 All messages and spam messages associated with a popular topic “#LookAtMeNow”. Spammers quickly started posting spam messages as the topic became popular. 104

5.5 Spamness variation by a number of spam items in top 10 search result . . . 110

5.6 The left figure depicts a topic distribution generated by the model. Each color denotes a topic. The right figure depicts a log-log graph showing the frequency of number of content items associated with each topic in the simulation data. The heavy-tailed distribution is similar to bursty social media. . . . 114

5.7 Evaluating the impact of increasing the fraction of spammers in the system from 0 to 5% (i.e., $0 \sim 0.05$ in the x-axis) . . . 115

5.8 Evaluating the impact of increasing the fraction of spammers in the system from 0 to 100% (i.e., $0 \sim 1$ in the x-axis) . . . 116

5.9 Coordinated Spam: By focusing their efforts, groups can achieve even higher impact. . . . 118

5.10 With as few as 20% adopting the group strategy, spamness passes the 0.20 threshold. . . . 119

5.11 Applying a simple rule-based countermeasure greatly reduces spamness, but is not effective against strategic behavior. . . . 120

5.12 Evaluating Countermeasure 2: Supervised Classification. Average accuracy and false positive rate reported across 101 topics. . . . 127

5.13 Evaluating Countermeasure 2: Supervised Classification. Figures illustrate the total spam detected across all 101 topics for increasing deployment times. . . . 131

5.14 (Color) Spamness reduction change rate after applying the combined countermeasure. Spamness falls by 59% for the recency-based approach and by 73% for the relevance-based one. . . . 133

6.1 An example social media manipulation campaign . . . 140

6.2 Top 10 countries of workers and requesters . . . 142

6.3 Linking crowdsourcing workers to social media . . . 144

6.4 Three activity-based characteristics of workers (red line with stars) and non-workers (blue line with circles). Workers tend to mention few other users, but retweet more often, and include links more often than non-workers. . . . 147

6.5 Three linguistic characteristics of workers (red line with stars) and non-workers (blue line with circles). Workers tend to swear less, use anger less, and use the 1st-person singular less than non-workers. . . 150

6.6 Network structure of all workers 151

6.7 Network structure of professional workers 155

LIST OF TABLES

TABLE Page

3.1 Confusion matrix example 23

3.2 MySpace – Performance results of top 10 classifiers 27

3.3 Twitter – Performance results of top 10 classifiers 32

3.4 Statistics of MySpace dataset 33

3.5 Statistics of Twitter dataset 33

3.6 Example of "About Me" content in new deceptive spam profiles . . . 36

3.7 Content polluter examples 41

3.8 Top five URLs posted by Content Polluters 46

3.9 Dataset 47

3.10 Confusion matrix 48

3.11 Top 10 features with χ^2 value and average feature values of polluters and legitimate users 51

3.12 The performance result of Random Forest 52

3.13 Boosting and bagging of the Random Forest classifier 52

3.14 The performance results of various feature group combinations 53

4.1 Identifying correlated messages 81

4.2 Refinements to shingling 83

4.3 Effectiveness comparison of campaign detection approaches 84

4.4 Top-10 largest campaigns 90

4.5 Top-10 significant terms for each campaign category 94

4.6 Campaign categories for low confidence threshold and high confidence threshold 94

4.7 Categories of Top-50 cohesive campaigns 96

5.1 A sample of 101 popular topics. In total, there are ~13m messages, of which 3.7% are spam 112

5.2 Evaluating the potential effectiveness of a low-accuracy (40%) and a high-accuracy (90%) collective spam detector 122

5.3 Confusion matrix 123

5.4 Top 10 features 126

5.5 On average, the supervised classifier countermeasure achieves 98% accuracy, detecting 50% of all spam messages 128

5.6 Combining countermeasures. We see a slight improvement in total spam detected compared to strictly applying the supervised classifier (from 50% to 55%) 132

6.1 Characteristics of crowdturfing workers 142

6.2 Characteristics of crowdturfing requesters 144

6.3 Twitter dataset 146

6.4 Properties of workers 148

6.5 Properties of non-workers 148

6.6 Top-10 hubs of the workers 153

6.7 Top-10 authorities of the workers 154

6.8 Top-10 followings of the professional workers 156

6.9 Top-10 followers of the professional workers 156

6.10 Top-10 middlemen with the number of professional workers 158

6.11 10 most retweeted messages by professional workers 159

1. INTRODUCTION

1.1 Motivation

The past few years have seen the rapid rise of many large-scale social systems – from Web-based social networks (e.g., Facebook, LinkedIn) to online social media sites (e.g., YouTube, Flickr) to large-scale information sharing communities (e.g., reddit, Yahoo! Answers) to crowd-based funding services (e.g., Kickstarter, IndieGoGo) to Web-scale crowdsourcing systems (e.g., Amazon Mechanical Turk, Crowdflower).

One of the key features of these systems is their leveraging of massive numbers of human participants – as primary contributors of content, as annotators and raters of other's content, as sources of expertise, and so on. We can observe the rapid growth of human participants in various social systems. For example, Facebook – launched in 2004 – now has more than a billion monthly active users. Among them, approximately 82% of monthly active users are outside the U.S and Canada, and 680 million monthly active users access Facebook via Facebook mobile products [35]. Twitter has over 500 million users, who generate over 340 million tweets daily; the Twitter service handles over 1.6 billion search queries per day [85, 140, 138]. Similarly online video site such as YouTube have become very popular. YouTube now has over 800 million unique users visit YouTube each month, and over 4 billion hours of video are watched each month on YouTube [160]. Freelancer.com and Elance.com – popular crowdsourcing sites – has 6.5 million users and 2 million users, respectively [86]. This reliance on users can lead to many positive effects, including large-scale growth in the size and content in the community, bottom-up discovery of "citizen-experts", serendipitous discovery of new resources beyond the scope of the system designers, and new social-based information search and retrieval algorithms. response to real-

world events or online phenomena.

But the relative openness and reliance on users coupled with the widespread interest and growth of these social systems carry risks and raise growing concerns over the quality of information in these systems. For example, 8.7% user accounts on Facebook in 2012 (83 million accounts) are fake [110, 121]. Akismet and Mollom reported that up to 90% of all comments on the Web are spam [151, 122]. Similarly, 1% messages and 5% accounts on Twitter are spam and spam accounts, respectively [137, 117]. VideoSurf estimated that 20% of online videos are spam [42]. Other new threats have been observed – including collective attention spam targeting a popular topic or an item to where user attention goes [75], malicious requesters recruiting many workers from a crowdsoucing site to spread manipulated contents to other websites for astroturfing [143], misinformation [17], fake reviews [96, 97], and astroturfing political campaigns [115].

1.2 Research Challenges

In the previous section, we described positive aspects of large-scale social systems, and several threats to information quality in these systems. We now identify some research challenges associated with these threats as follows:

- *Openness:* Social systems are inherently open to users who generate, share and consume information. Maintaining this openness is important for their continued growth and impact. But the openness and reliance of users allowed malicious participants to threaten information quality in these systems. Can we maintain the openness of social systems with assuring the quality of information?

- *Collaboration:* Many users organically participate in social systems to engage in collaborative activities (e.g., organizing for political change and sharing

disaster-related information). But social systems are a prime target for strategic influence. Some users inorganically engage with social systems and degrade quality of information by forming "astroturfing" campaigns, promoting a product aggressively and sending unwanted messages.

- *Collective Dynamics:* Collective attention – exemplified by breaking news, viral videos, and popular memes that captivate the attention of huge numbers of users – is one of the cornerstones of large-scale social systems. But this self-organization, leading to user attention quickly coalescing and then collectively focusing around a phenomenon, opens these systems to new threats like collective attention spam.

1.3 Overview of this Dissertation

In this dissertation research, we first identify three classes of threats to these systems (as shown in Figure 1.1): (i) content pollution by social spammers, (ii) coordinated campaigns for strategic manipulation, and (iii) threats to collective attention. To combat these threats, we propose four inter-related methods for detecting evidence of these threats, mitigating their impact, and improving the quality of information in social systems. We augment this three-fold defense with an exploration of their origins in "crowdturfing" – a sinister counterpart to the enormous positive opportunities of crowdsourcing. In particular, this dissertation research makes four unique contributions toward this direction:

- The first contribution of this dissertation research is a framework for detecting and filtering social spammers and content polluters in social systems. These spammers and content polluters are increasingly targeting these systems as part of phishing attacks, to disseminate malware and commercial spam messages, and to promote affiliate websites. Successfully defending against these

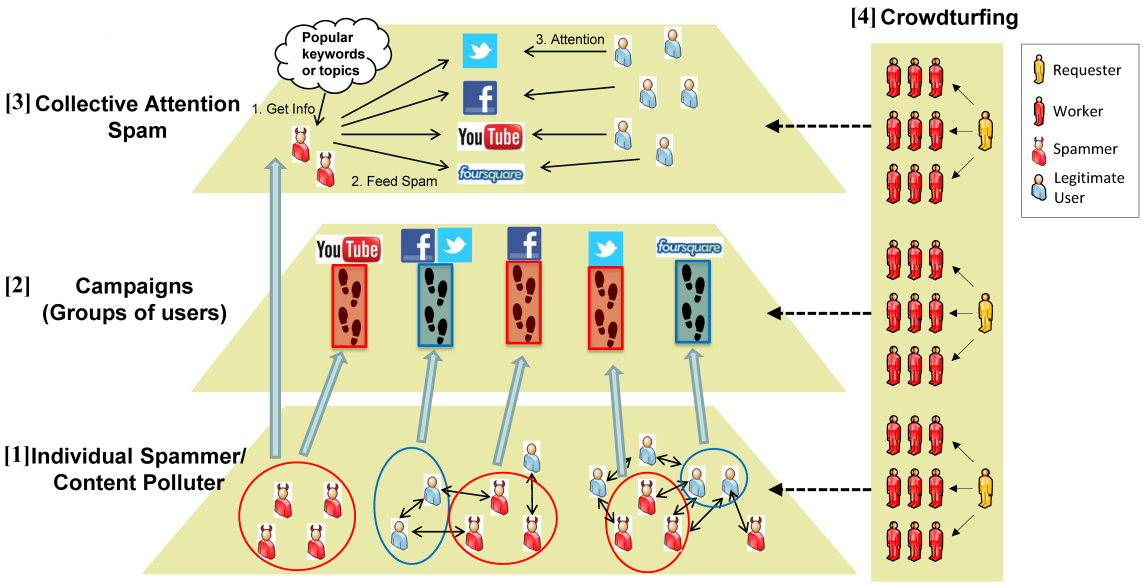


Figure 1.1: Overall Dissertation Organization

social spammers and content polluters is important to improve the quality of experience for community members, to lessen the system load of dealing with unwanted and sometimes dangerous content, and to positively impact the overall value of the social system going forward.

To detect and filter social spammers and content polluters, we propose a novel social honeypot-based approach. Two of the key components of the proposed approach are: (i) The deployment of social honeypots for harvesting deceptive spam profiles from social networking communities; and (ii) Statistical analysis of the properties of these spam profiles for creating spam classifiers to actively filter out existing and new spammers. Our empirical evaluation over both MySpace and Twitter demonstrates the effectiveness and adaptability of the honeypot-based approach, leading to the development of robust classifiers for ongoing protection.

- Second, we present a set of methods and algorithms for detecting coordinated campaigns in large-scale social systems. These campaigns – ranging from coordinated spam messages to promotional and advertising campaigns to political astro-turfing – are growing in significance and reach with the commensurate rise in massive-scale social systems. However, there has been little research in detecting these campaigns of strategic manipulation "in the wild".

  Hence, we propose and evaluate a content-driven framework for effectively linking free text posts with common "talking points" and extracting campaigns from large-scale social systems. Three of the salient features of the campaign extraction framework are: (i) an investigation of graph mining techniques for isolating coherent campaigns from large message-based graphs; (ii) a comprehensive comparative study of text-based message correlation in message and user levels; and (iii) an analysis of the temporal behaviors of various campaign types. Through an experimental study over millions of Twitter messages we identify five major types of campaigns – Spam, Promotion, Template, News, and Celebrity campaigns – and we show how these campaigns may be extracted with high precision and recall.

- Third, we examine threats to *collective attention* – exemplified by breaking news, viral videos, and popular memes that captivate the attention of huge numbers of users. But this self-organization opens these systems to new threats. In one direction, we have observed what we refer to as *collective attention spam* which relies on the users themselves to seek out the content – like breaking news, viral videos, and popular memes – where the spam will be encountered, potentially increasing its effectiveness and reach.

  To identify threats to collective attention, we take a two fold approach. First,

we develop *data-driven models* to simulate large-scale social systems based on parameters derived from a real system. In this way, we can vary parameters – like the fraction of malicious users in the system, their strategies, and the countermeasures available to system operators – to explore the resilience of these systems to threats to collective attention. Second, we pair the data-driven model with a *comprehensive evaluation over a Twitter system trace*, in which we evaluate the effectiveness of countermeasures deployed based on the first moments of a bursting phenomenon in a real system. Our experimental study shows the promise of these countermeasures to identifying threats to collective attention early in the lifecycle, providing a shield for unsuspecting social media users.

- Finally, we investigate the origins of threats to social systems. We study the ecosystem of crowdturfing which has recently been identified as a sinister counterpart to the enormous positive opportunities of crowdsourcing. Crowdturfers leverage human-powered crowdsourcing platforms to spread malicious URLs in social media, form "astroturf" campaigns, and manipulate search engines, ultimately degrading the quality of online information and threatening the usefulness of these systems.

  We present a framework for "pulling back the curtain" on crowdturfers to reveal their underlying ecosystem. Concretely, we analyze the types of malicious tasks and the properties of requesters and workers in crowdsourcing sites such as Microworkers.com, ShortTask.com and Rapidworkers.com, and link these tasks (and their associated workers) on crowdsourcing sites to social media, by monitoring the activities of social media participants. Based on this linkage, we identify the relationship structure connecting these workers in social media,

which can reveal the implicit power structure of crowdturfers identified on crowdsourcing sites. We identify three classes of crowdturfers – professional workers, casual workers, and middlemen.

The rest of this dissertation is organized as follows:

- *Chapter 2:* In this chapter, we discuss related work, specifically threats to information quality and methods to tackle the threats in email, search engines, social networks and crowdsourcing sites.

- *Chapter 3:* In this chapter, we present the design and real-world evaluation of a novel social honeypot-based approach to detect social spammers and content polluters. We investigate techniques and develop effective tools for automatically detecting and filtering social spammers. We also design and evaluate a system for automatically detecting and profiling content polluters on Twitter.

- *Chapter 4:* In this chapter, we investigate the problem of campaign detection in social media. We propose and evaluate an efficient content-driven graph-based framework for identifying and extracting campaigns from the massive scale of real-time social systems. We find six campaign types (spam, promotion, template, celebrity, news and babble), and analyze temporal behaviors of various campaign types.

- *Chapter 5:* In this chapter, we present a dual study of the robustness of social systems to collective attention threats through both a *data-driven modeling* approach and *deployment over a real system trace*. We explore the resilience of large-scale social systems to threats to collective attention, and identify countermeasures and demonstrate their effectiveness at filtering spam during the early development of a bursting phenomenon in a real system.

- *Chapter 6:* We present a framework for "pulling back the curtain" on crowdturfers to reveal their underlying ecosystem. By linking malicious tasks and their workers on crowdsourcing site to social media, we identify three classes of crowdturfers – professional workers, casual workers, and middlemen.

- *Chapter 7:* We conclude with a summary of the contributions of this work and provide a discussion of future research extensions to the results presented here.

2. RELATED WORK

In this chapter, we summarize threats to information quality and methods to tackle the threats in email, the Web, social networks and crowdsourcing sites.

2.1 Email and Web

To prevent and detect email spam, many approaches have been developed, including content-based filtering like whitelisting, blacklisting, keyword-based [26], statistical classification [3, 120], heuristic-based filtering [28, 129], collaborative filtering [108], network-level clustering approach [112], spambot identification [106, 127], data compression algorithms [12], and behavioral blacklisting [114]. Other classes of filtering approaches include challenge-response [62], MTA/Gateway filtering (Tarproxy [74]), greylisting [52], and micropayments [69]. Researchers have also analyzed the network-level characteristics of spammers [113], the underlying business operations of spam-advertised enterprises [65] and common spam in tweets and email [84], have quantified the effect of email spam on behavior and engagement of email users [30], and have studied the spam value chain [80]. In addition, Qian et al. [111] revealed triangular spamming that exploits routing irregularities of spoofed IP packets, and proposed practical detection and prevention methods. Stringhini et al. [126] proposed a system for filtering spam that takes into account how messages are sent from spammers, analyzing the communication at the SMTP protocol level. Improving spam blacklisting in terms of the false positive and false negative rates by dynamic thresholding and speculative aggregation was proposed [123]. Isacenkova and Balzarotti [61] conducted a measurement study of the behavior of a real world deployment of a challenge-response anti-spam system. Pitsillidis et al. [105] conducted comparative analysis of spam datasets.

Similarly, in the Web domain, researchers have considered content-based features of Web pages [37, 99], link-based features to distinguish between spam and legitimate hyperlinks [2, 31, 34], anomalous linking patterns [93, 6, 51, 152], and trust-aware ranking for favoring whitelist Web pages [50, 153]. Cheng et al. [23] have detected spam websites using a semi-supervised learning framework based on posts in SEO forums. Caverlee and Liu [18] measure the "credibility" of each web page to find good and malicious pages. Other researchers have studied to detect link farms [5, 7].

2.2 Social Networks

Several research efforts have found a high degree of reciprocity in social networks (e.g., [72]), meaning that many users may elect to make themselves susceptible to a spammer (e.g., by becoming "friends" and subsequently the target of spam messages). Jagatic et al. [63] have shown that adding "social" contextual clues (like sending a spam message from a known "friend" account) can increase the effectiveness of such attacks. Similarly, Brown et al. [16] showed that context-aware attacks in social systems are very effective. Heymann et al. [53] have summarized three main anti-spam strategies in social systems: (i) detection strategy; (ii) demotion strategy; and (iii) prevention strategy in social networking communities. Detection strategy is that users or moderators detect spam manually or the system detects it automatically. Demotion strategy requires using ranking algorithms so that relevant pages or objects can be retrieved on top k results. Prevention strategy is that system designers limit automated interaction or make some details of the system secret. Markines et al. [90] assumed that the purpose of social spammers is to earn money by attracting users to advertisements. They proposed six features based on the post-level (tags), the resource-level (annotated web pages), and the user-level. They found that post-level information was most effective for spam classification.

Friend-in-the-middle attacks (hijacking approach) on social networking sites can be used to harvest social data, and the collected data is used for spam and phishing emails targeted at the victim's friends [56]. Wondracek et al. [150] revealed a novel de-anonymization attack that exploits group membership information on social networking sites. They warned that spammers can predict a user's identify by misusing the group membership information and target the user for future attack like sending spam or disseminating malware software. [132] analyzed the tweets sent by suspended users on Twitter.

Other types of social spam have been described and solutions proposed. Examples include Twitter-based threats like spam URLs posted in Twitter [49], the detection of malicious URLs [87], real-time URL spam filtering service [131], trend-stuffing [59], the existence of malicious Twitter spam accounts and methods for detecting them [76], and the adoption of link farms in Twitter [45]. Other researchers have proposed domain-specific spam detection solutions for video sites, social tagging sites, and others [8, 60, 68, 82, 101]. In addition, researchers have begun studying group spammers and their tactics. Gao et al. [44] studied spam behavior on Facebook; their approach finds coordinated spam messages that use the same malicious URL. Mukherjee et al. [96, 97] proposed an approach to detect group spammers in product reviews. Their approach consists of frequent itemset mining techniques and several behavior features derived from collusion among fake reviewers. The Truthy system [115] detects astroturf political campaigns on Twitter. Gonçalves et al. described the abuse of social media and pollical manipulation [48]. Gao et al. [43] proposed campaign identification approach and validated it on Facebook and Twitter data. Caverlee et al. [19] have proposed the *SocialTrust* framework for tamper-resilient trust establishment in online communities.

2.3 Crowdsourcing Sites

With the rise in popularity of commercial crowdsourcing services, there have been many efforts to analyze the nature of jobs available and their characteristics. For example Kittur et al. [66] studied Amazon Mechanical Turk and found that a large number of workers can be hired within a short time and for low cost. Similar studies – e.g., [11] – have shown the potential of crowdsourcing. And researchers have begun developing new crowd-based platforms – e.g., [1] [40] – for augmenting traditional information retrieval and database systems, embedding crowds into workflows (like document authoring) [9], and so forth.

A key question for these crowd-based systems is how to control the quality of workers and outputs due to the openness of these sites. For example, Venetis and Garcia-Molina [141] described two quality control mechanisms. The first mechanism repeats each task multiple times and combines the results from multiple users. The second mechanism defines a score for each worker and eliminates the work from users with low scores. Xia et al. [157] provided a real-time quality control strategy for workers who evaluate the relevance of search engine results based on the combination of a qualification test of the workers and the time spent on the actual task. The results are promising and these strategies facilitate reducing the number of bad workers. Recently, Wang et al. [143] coined the term "crowdturfing" (crowdsourcing + astroturfing) to refer to crowdsourcing systems where malicious campaigns are hosted by employers. They have studied crowdsourcing sites based in China and the impact of these sites on one social networking site – Weibo.

3. DETECTING SOCIAL SPAMMERS AND CONTENT POLLUTERS*

3.1 Introduction

In this chapter, we begin to investigate our first threat to social systems. This threat has been caused by individual social spammers and content polluters.

Social spammers are increasingly targeting these systems as part of phishing attacks [63], to disseminate malware [10] and commercial spam messages [16, 162], and to promote affiliate websites [83]. In the past year alone, more than 80% of social networking users have "received unwanted friend requests, messages, or postings on their social or professional network account" (Source: Harris Interactive, June 2008). Unlike traditional email-based spam, *social spam* often contains contextual information that can increase the impact of the spam (e.g., by eliciting a user to click on a phishing link sent from a "friend") [16, 36, 63].

Successfully defending against these social spammers is important to improve the quality of experience for community members, to lessen the system load of dealing with unwanted and sometimes dangerous content, and to positively impact the overall value of the social system going forward. However, little is known about these social spammers, their level of sophistication, or their strategies and tactics. Filling this need is challenging, especially in social networks consisting of 100s of millions of user profiles (like Facebook, MySpace, Twitter, YouTube, etc.). Traditional techniques for discovering evidence of spam users often rely on costly human-in-the-loop

*Reprinted with permission from "Uncovering social spammers: social honeypots + machine learning" by Kyumin Lee, James Caverlee, and Steve Webb, 2010. *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, 435-442, Copyright 2010 by ACM. Reprinted with permission from "Seven Months with the Devils: A Long-Term Study of Content Polluters on Twitter" by Kyumin Lee, Brian David Eoff, and James Caverlee, 2011. *Proceedings of the 5th International AAAI Conference on Weblogs and Social Media*, Copyright 2011 by AAAI.

inspection of training data for building spam classifiers; since spammers constantly adapt their strategies and tactics, the learned spam signatures can go stale quickly. An alternative spam discovery technique relies on community-contributed spam referrals (e.g., Users A, B, and C report that User X is a spam user); of course, these kinds of referral systems can be manipulated themselves to yield spam labels on legitimate users, thereby obscuring the labeling effectiveness. And neither spam discovery approach can effectively handle *zero-day* social spam attacks for which there is no existing signature or wide evidence.

With these challenges in mind, we propose and evaluate a novel honeypot-based approach for detecting social spammers and content polluters in social systems. In Section 3.2, the proposed approach is designed to (i) automatically harvest spam profiles from social networking communities, avoiding the drawbacks of burdensome human inspection; (ii) develop robust statistical user models for distinguishing between social spammers and legitimate users; and (iii) actively filter out unknown (including zero-day) spammers based on these user models. Drawing inspiration from security researchers who have used honeypots to observe and analyze malicious activity (e.g., for characterizing malicious hacker activity [124], generating intrusion detection signatures [70], and observing email address harvesters [109]), we deploy and maintain *social honeypots* for trapping evidence of spam profile behavior, so that users who are detected by the honeypot have a high likelihood of being a spammer (i.e., low false positive rate). Over two distinct communities (MySpace and Twitter), we find that the proposed approach provides generalizable and effective social spam detection.

Section 3.3 presents the first long-term study of social honeypots via a seven-month deployment of 60 honeypots on Twitter that resulted in the harvesting of

36,000 candidate content polluters[†]. We provide a detailed examination of the harvested Twitter users, including an analysis of link payloads, user behavior over time, and followers/following network dynamics. We experimentally evaluate a wide range of features – including user demographics, properties of the Twitter follower/following social graph, Tweet content, and temporal aspects of user behavior – to investigate the effectiveness of automatic content polluter identification, even in the presence of strategic polluter obfuscation. Finally, we empirically validate the social honeypot-derived classification framework on an alternative Twitter spam dataset, which shows the flexibility and effectiveness of the proposed approach.

3.2 Uncovering Social Spammers

3.2.1 Overall Framework

In this subsection, we present the conceptual framework of the proposed honeypot-based approach and outline the research questions motivating our examination of this framework.

3.2.1.1 Problem Statement

In social networking communities like MySpace and Facebook, there are a set of k users $U = \{u_1, u_2, \ldots, u_k\}$. Each user u_i has a profile p_i. Profiles are self-describing pages that are created and controlled by a given user. For example, users typically include information such as their name, gender, age, and so on in their profiles. Each community has its own profile format, but most fields in the formats are the same.

The **social spam detection problem** is to predict whether u_i is a spammer through a classifier c when p_i is given. A classifier

$$c : u_i \rightarrow \{spammer, legitimate\, user\}$$

[†]We use the term "social spammer" and "content polluter" interchangeably.

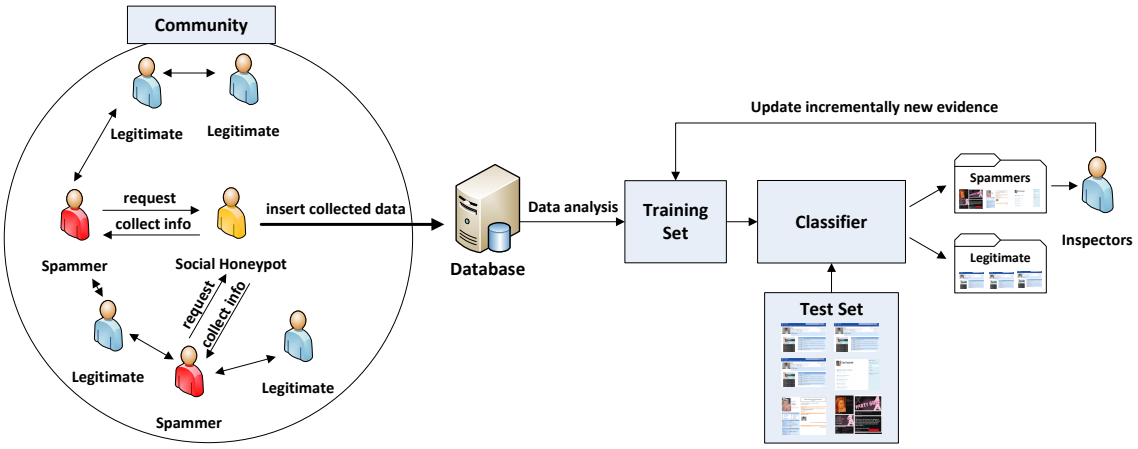


Figure 3.1: Overall Framework of Social Honeypot-based Approach

approximates whether u_i is a spammer. To build c, we need to extract a set of m features $F = \{f_1, f_2, \ldots, f_m\}$ from U. For example, we can extract F_{u_i} from p_i of u_i.

Whereas traditional email spam detection has focused on identifying *spam messages* which are of relatively low individual value to the spammer (and whose identification typically doesn't threaten the ongoing ability of a spammer to send new messages), social spam detection is focused on identifying and eliminating *spam accounts* themselves. This detection is potentially more disruptive to spammers, since these accounts typically represent a more expensive investment by the spammer (through email and social media account registrations).

3.2.1.2 Solution Approach

We propose to monitor spammer activity through the creation of social honeypots. We define social honeypots as information system resources that monitor spammers' behaviors and log their information (e.g., their profiles and other content created by them in social networking communities). Social honeypots and traditional honeypots (e.g., in domains such as network systems and emails [70, 109, 124]) share a similar

purpose in that they both monitor and log the behaviors of spammers or attackers. However, traditional honeypots typically target network or systems-level behavior, whereas social honeypots specifically target community-based online activities.

While social honeypots alone are a potentially valuable tool for gathering evidence of social spam attacks and supporting a greater understanding of spam strategies, it is the goal of this research project to support ongoing and active *automatic* detection of new and emerging spammers (See Figure 3.1). In practice, we deploy a social honeypot consisting of a legitimate profile and an associated bot to detect social spam behavior. If the social honeypot detects suspicious user activity (e.g., the honeypot's profile receiving an unsolicited friend request) then the social honeypot's bot collects evidence of the spam candidate (e.g., by crawling the profile of the user sending the unsolicited friend request plus hyperlinks from the profile). What entails *suspicious user behavior* can be optimized for the particular community and updated based on new observations of spammer activity.

As the social honeypots collect spam evidence, we extract observable features from the collected candidate spam profiles (e.g., number of friends, text on the profile, age, etc.). Coupled with a set of known legitimate (non-spam) profiles which are more populous and easy to extract from social networking communities, these spam and legitimate profiles become part of the initial training set of a spam classifier. Through iterative refinement of the features selected and the classifier used (e.g., SVM), the spam classifier can be optimized over the known spam and legitimate profiles.

Based on these developed classifiers, we can then explore the wider space of unknown profiles. On MySpace alone there are 100s of millions of profiles, of which some unknown fraction are spam. Using the classifiers based on the harvested social honeypot data, we iteratively explore these profiles "in-the-wild" to detect new spammers that have yet to be identified by a social honeypot directly. In our design

of the overall architecture, we include human inspectors in-the-loop for validating the quality of these extracted spam candidates. Instead of inspecting the entirety of all profiles, these inspectors are guided to validate just the few spam candidates recommended by the learned classifiers. Based on their feedback, the spam classifiers are updated with the new evidence and the process continues. Given the overall architecture, we address three important research challenges in turn in the rest of this chapter:

- *Research Challenge #1 [RC1]:* Do social honeypots collect evidence of spam with low false positives? In other words, do honeypots really work in practice? A poorly performing social honeypot will negatively impact the spam classification approach, leading to poor performance.

- *Research Challenge #2 [RC2]:* Can we build classifiers from the harvested social honeypot profiles and known legitimate profiles to effectively identify spam profiles? Since social honeypots are triggered by suspicious user *activity*, we must explore if the harvested spam data contains signals that are strongly correlated with observable profile features (e.g., content, friend information, posting patterns, etc.). It is our hypothesis that spammers engage in behavior that is correlated with observable features that distinguish them from legitimate users.

- *Research Challenge #3 [RC3]:* Finally, can the developed classifiers be effectively deployed over large collections of unknown profiles (for which we have no assurances of the degree of spam or legitimate users)? This last question is important for understanding the promise of social honeypots in defending against new and emerging spam as it arises "in-the-wild."

3.2.2 RC1: Study of Harvested Spam Users

Based on the overall social honeypot framework, we selected two social networking communities – Myspace and Twitter – to evaluate the effectiveness of the proposed spam defense mechanism. Both MySpace and Twitter are large and growing communities and both also support public access to their profiles, so all data collection can rely on purely public data capture.

- *MySpace Social Honeypot Deployment:* In previous research [146], we created 51 generic honeypot profiles within the MySpace community for attracting spammer activity so that we can identify and analyze the characteristics of social spam profiles. To observe any geographic artifacts of spamming behavior, each profile was assigned a specific geographic location (i.e., one honeypot was assigned to each of the U.S. states and Washington, D.C.). Each honeypot profile tracks all unsolicited friend requests. Upon receiving a friend request, we store a local copy of the profile issuing the friend request, extract all hyperlinks in the "About Me" sections (we find that these typically point to an affiliate spam website), and crawl the pages pointed to by these hyperlinks. Based on a four month evaluation period (October 2007 to January 2008), we collected 1,570 profiles whose users sent unsolicited friend requests to these social honeypots.

- *Twitter Social Honeypot Deployment:* Similarly, we created and deployed a mix of honeypots within the Twitter community – some of them had personal information such as biography, location and so on, while others did not have this personal information. Some social honeypots posted tweets, while others did not post them. We omit some of the concrete details of the Twitter honeypot deployment due to the space constraint. From August 2009 to September 2009,

these social honeypots collected 500 users' data.

3.2.2.1 MySpace Observations

After analyzing the harvested spam profiles from MySpace, we find some interesting observations (more fully detailed in [146]): (1) The spamming behaviors of spam profiles follow distinct temporal patterns. (2) The most popular spamming targets are Midwestern states, and the most popular location for spam profiles is California. (3) 57.2% of the spam profiles copy their "About Me" content from another profile. (4) Many of the spam profiles exhibit distinct demographic characteristics (e.g., age, relationship status, etc.). (5) Spam profiles use thousands of URLs and various redirection techniques to funnel users to a handful of destination Web pages. Through manual inspection, we grouped the harvested spam profiles into five categories:

- *Click Traps:* Each profile contains a background image that is also a link to another Web page. If users click anywhere on the profile, they are directed to the link's corresponding Web site.

- *Friend Infiltrators:* These nominally legitimate profiles befriend as many users as possible so that they can infiltrate the users' circles of friends and bypass any communication restrictions imposed on non-friends. Once a user accepts a friend request from one of these profiles, the profile begins spamming the user through existing communication systems (e.g., message spam, comment spam, etc.).

- *Pornographic Storytellers:* Each of these profiles has an "About Me" section that consists of randomized pornographic stories, which are book-ended by links that lead to pornographic Web pages. The anchor text used in these

profiles is extremely similar, even though the rest of the "About Me" text is almost completely randomized.

- *Japanese Pill Pushers:* These profiles contain a sales pitch for male enhancement pills in their "About Me" sections. According to the pitch, the attractive woman pictured in the profile has a boyfriend who purchased these pills at an incredible discount.

- *Winnies:* All of these profiles have the same headline: "Hey its winnie." However, despite this headline, none of the profiles are actually named "Winnie." In addition to a shared headline, each of the profiles also includes a link to a Web page where users can see the pictured female's pornographic pictures.

3.2.2.2 Twitter Observations

Similarly, we discovered various types of spam users in the harvested data from Twitter. In many cases, spammers inserted malicious or spam links into their tweets. Since most Twitter links use a form of URL-shortening, users clicking on these links have no assurances of the actual destination.

- *Duplicate Spammers:* These users post a series of nearly identical tweets. In many cases, the only different content from tweet to tweet is the inclusion of different @usernames (or @replies). The insertion of these @usernames essentially delivers the tweet to the *username*'s account even if the spammer has no relationship with the intended target.

- *Pornographic Spammers:* Their data such as user image, profile URL, and text and links in tweets contains adult content.

- *Promoters:* These users post tweets about several things such as online business, marketing and so on. Their posting approach is more sophisticated than

duplicate spammers since spam tweets are randomly interspersed with seemingly innocuous legitimate tweets.

- *Phishers:* Similar to promoters, these users use a mix of strategies to deliver phishing URLs to targets on Twitter.

- *Friend Infiltrators:* Much like their counterparts on MySpace, these users have profiles and tweets that are seemingly legitimate. They follow many people and intend to accumulate many followers; then they begin engaging in spam activities like posting tweets containing pornographic or commercial content.

These observations indicate that social honeypots can successfully attract spammers across fundamentally different communities (MySpace vs. Twitter), and the results suggest that building automatic classifiers may be useful for identifying social spam.

3.2.3 RC2: Empirical Evaluation of Social Spam Signals

We next explore whether there are discernible spam signals in the harvested spam profiles that can be used to automatically distinguish spam profiles from legitimate profiles. Since social honeypots are triggered by spam *behaviors* only, it is unclear if the corresponding profiles engaging in the spam behavior also exhibit clearly observable spam signals. If there are clear patterns (as our observations in the previous subsection would seem to indicate), then by training a classifier on the observable signals, we may be able to predict new spam even in the absence of triggering spam behaviors.

3.2.3.1 Classification Approach and Metrics

As part of this empirical evaluation of social spam signals, we consider four broad classes of user attributes that are typically observable (unlike, say, private messaging

between two users) in the social network: (i) user demographics: including age, gender, location, and other descriptive information about the user; (ii) user-contributed content: including "About Me" text, blog posts, comments posted on other user's profiles, tweets, etc.; (iii) user activity features: including posting rate, tweet frequency; (iv) user connections: including number of friends in the social network, followers, following. For MySpace and Twitter we select a subset of these features to train the classifier.

The classification experiments were performed using 10-fold cross-validation to improve the reliability of classifier evaluations. When a dataset is not large, it is common to use 10-fold cross-validation to achieve statistically precise results. In 10-fold cross-validation, the original sample is randomly divided into 10 equally-sized sub-samples. 9 sub-samples are used as a training set and the remaining one is used as a testing set; the classifier is evaluated, then the process is repeated for a total of 10 times. Each sub-sample is used as a testing set once in each evaluation. The final evaluation result is generated by averaging the results of the 10 evaluations. In practice, we evaluated over 60 different classifiers in the Weka [149] machine learning toolkit using 10-fold cross-validation with default values for all parameters. Classification results are presented in the form of a confusion matrix as in Table 3.1.

Table 3.1: Confusion matrix example

		Predicted	
		Spammer	Legitimate
Actual	Spammer	a	b
	Legitimate	c	d

To measure the effectiveness of classifiers based on our proposed features, we used

the standard metrics such as precision, recall, accuracy, the F_1 measure, false positive and true positive. Precision is the ratio of correctly predicted users as a class to the total predicted users as the class. For example, the precision (P) of the spammer class in Table 3.1 is $a/(a+c)$. Recall (R) is the ratio of correctly predicted users as a class to the actual users in the class. The recall of the spammer class in the table is $a/(a+b)$. Accuracy is the proportion of the total number of predictions that were correct. The accuracy in the table is $(a+d)/(a+b+c+d)$. F_1 is a measure that trades off precision versus recall. F_1 measure of the spammer class is $2PR/(P+R)$. A false positive is when the actual Y class users are incorrectly predicted as X class users. The false positive of the spammer class is c. A true positive is when actual X class users are correctly predicted as X class users. The true positive of the spammer class is a.

To measure the discrimination power between spammers and legitimate users of each of the proposed features, we generate a Receiver Operating Characteristics (ROC) curve. ROC curves plot false positive rate on the X axis and true positive rate on the Y axis. The closer the ROC curve is to the upper left corner, the higher the overall accuracy is. The ideal ROC curve includes the coordinate (0, 1), indicating no false positives and a 100% true positive rate.

3.2.3.2 MySpace Spam Classification

We randomly sampled 388 legitimate profiles from MySpace (which were labeled by us) and 627 deceptive spam profiles from the 1,570 deceptive spam profiles collected by our social honeypots. When we sampled the profiles, we considered several conditions. Profiles have to be public, and marital status, gender, age, and "About Me" content in the profiles have to be valid (i.e., a non-empty value). In addition, we removed duplicated profiles among the 1,570 deceptive spam profiles in the case

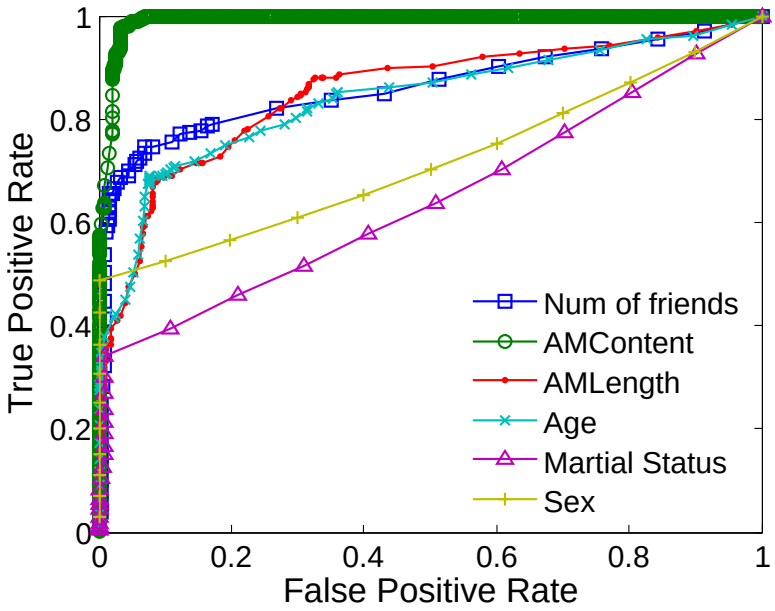


Figure 3.2: MySpace – Feature Comparison

that a spammer sent a friend request to several social honeypots. The goal of spam classification over the MySpace data is to predict whether a profile is either spammer or legitimate.

We considered several representative user features: number of friends, age, marital status, gender, as well as some text-based features modeling user-contributed content in the "About Me" section. Specifically, we consider a bag-of-words model in which we remove punctuation, make all letters lowercase, tokenize each word, remove stopwords, and do stemming for each word using the Porter stemmer [107]. We assigned weights to each word based on tf-idf weighting: $\text{tf-idf}_{t,d} = \log(1 + tf_{t,d}) \times \log(\frac{N}{df_t})$, where $\text{tf}_{t,d}$ means term frequency of term t in a profile's "About Me", N is the number of profiles, and df_t is the number of profiles which includes term t. We also measured the length in bytes of the "About Me" content.

Before evaluating the effectiveness of our classifiers, we investigated the discrim-

ination power of our individual classification features. Recognizing that social spam classification is an example of an adversarial classification problem [29], we wanted to evaluate the robustness of our features against an adversarial attack. To show the discrimination power and robustness of each feature, we generated ROC curves for each feature using an AdaboostM1 classifier. The results are shown in Figure 3.2. Marital status and sex are the least discriminative features, which is unsurprising because they are represented by a small set of predefined values (e.g., "Married", "Single", "Male", etc.) that will inevitably appear in both legitimate and spam profiles. On the other hand, the bag of words features extracted from "About Me" content (AMContent) are the most discriminative. This is a very encouraging result because it means our classifier was able to distinguish between legitimate and spam "About Me" content with a high degree of accuracy. Therefore, if spammers begin varying the other features of their profiles (to appear more legitimate), our classifiers will still be effective. Additionally, the "About Me" content is the most difficult feature for a spammer to vary because it contains the actual sales pitch or deceptive content that is meant to target legitimate users.

In Table 3.2, the performance results for the top 10 classifiers are shown. The table clearly shows that all of the classifiers were successful. Each classifier generated an accuracy greater than 98.4%, an F_1 measure over 0.98, and a false positive rate below 1.6%. Overall, meta-classifiers (Decorate, LogitBoost, etc.) performed better than tree classifiers (FT and J48) and a function-based classifier (SimpleLogistic). The best classifier is Decorate, which is a meta-learner for building diverse ensembles of classifiers. It obtained an accuracy of 99.21%, an F_1 measure of 0.992, and a 0.7% false positive rate. We additionally considered different training mixtures of spam and legitimate training data (from 10% spam / 90% legitimate to 90% spam / 10% legitimate); we find that the metrics are robust across these changes in training data.

Table 3.2: MySpace – Performance results of top 10 classifiers

Weka Classifier	Accuracy	F_1	FP
Decorate	**99.21%**	**0.992**	**0.7%**
SimpleLogistic	99.01%	0.99	0.9%
FT	99.01%	0.99	0.9%
LogitBoost	99.01%	0.99	1.3%
RandomSubSpace	98.72%	0.987	1.1%
Bagging	98.62%	0.986	1.2%
J48	98.42%	0.984	1.5%
OrdinalClassClassifier	98.42%	0.984	1.5%
ClassBalancedND	98.42%	0.984	1.5%
DataNearBalancedND	98.42%	0.984	1.5%

3.2.3.3 Twitter Spam Classification

To evaluate the quality of spam classification over Twitter, we randomly selected 104 legitimate users (labeled by us) from a previously collected Twitter dataset of 210,000 users. We additionally considered two classes of spam users: the 61 spammers and the 107 promoters sampled from 500 users' data collected by the social honeypots. For each user, we collected the user profile, tweets (status update messages), following (friend) information and followers' information. The goal of spam classification over the Twitter data is to predict whether a profile is either spammer, a promoter, or legitimate. When we sampled users' data, we considered two conditions: the profiles did not have a *verified account badge* and the number of tweets had to be over 0. The *verified account badge* is one way Twitter ensures that profiles belong to known people (e.g, Shaquille O'Neal and not an impersonator).

Unlike MySpace profiles which emphasize on longer-form personal information sharing (e.g., "About Me" text) and usually have self-reported user demographics (e.g., age, gender), Twitter accounts are noted for their short posts, activity-related features, and limited self-reported user demographics. For user features, we consider

the longevity of the account on Twitter, the average tweets per day, the ratio of the number of following and number of followers, the percentage of bidirectional friends ($\frac{|following \cap followers|}{|following|}$), as well as some features of the tweets sent, including:

- The ratio of the number of URLs in the 20 most recently posted tweets to the number of tweets ($|URLs|/|tweets|$).
- The ratio of the number of *unique* URLs in the 20 most recently posted tweets to the number of tweets ($|unique\ URLs|/|tweets|$).
- The ratio of the number of @usernames in the 20 most recently posted tweets to the number of tweets ($|@username|/|tweets|$).
- The ratio of the number of *unique* @usernames in the 20 most recently posted tweets to the number of tweets ($|unique\ @username|/|tweets|$).

Additionally, we measure the average content similarity over all pairs of tweets posted by a user:

$$\sum_{a,b \in \text{set of pairs in tweets}} \frac{similarity(a,b)}{|\text{set of pairs in tweets}|}$$

where the content similarity is computed using the standard cosine similarity over the bag-of-words vector representation $\overrightarrow{V}(a)$ of the tweet content:

$$similarity(a,b) = \frac{\overrightarrow{V}(a) \cdot \overrightarrow{V}(b)}{|\overrightarrow{V}(a)||\overrightarrow{V}(b)|}$$

We finally considered some text-based features to model the content in the tweets. Since tweets are extremely short (140 characters or less), we consider a bag-of-words model and a sparse bigrams model [27]. We remove punctuation, make all letters

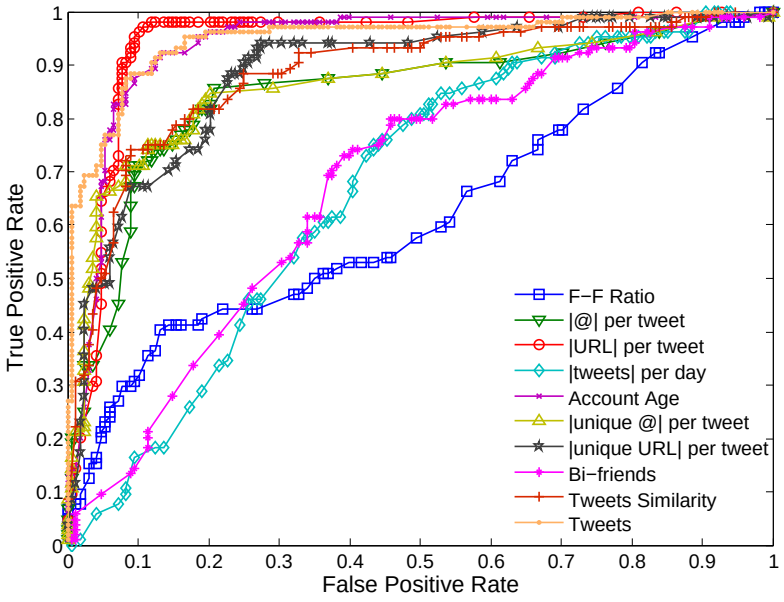
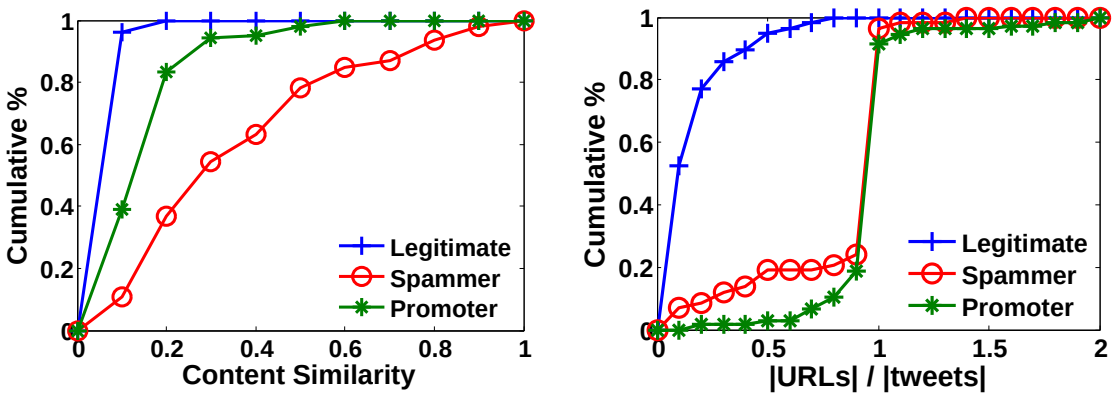


Figure 3.3: Twitter – Feature Comparison

lowercase, tokenize each word in the bag-of-words model and tokenize a pair of words in the sparse bigrams model. The sparse bigrams model generates a pair of words separated by no more than k words. We assigned k = 3 in our system, while k = 0 yields ordinary bigrams. If a tweet is "check adult page view models", the sparse bigrams will generate the features "check adult", "check page", "check view", "adult page", "adult view", "adult models", "page view", "page models", "view models". We weighted terms and bigrams using tf-idf weighting as in the previous MySpace classification.

In order to know how much discrimination power each feature has for spammers, promoters and legitimate users, we generated ROC curves of the proposed features using Decorate in Figure 3.3. Average posted tweets per day (*|tweets| per day*), percentage of bidirectional friends (*bi-friends*), and ratio of number of following and number of followers (*F-F Ratio*) have low discrimination powers relatively, while



(a) Content Similarity

(b) Avg URLs per Tweet

Figure 3.4: Cumulative Distribution of Features Extracted from Users

ratio of number of unique URLs in recently posted top 20 tweets and number of the tweets ($|unique\ URL|\ per\ tweet$), ratio of number of @username in recently posted top 20 tweets and number of the tweets ($|@|\ per\ tweet$), ratio of number of unique @username in recently posted top 20 tweets and number of the tweets ($|unique\ @|\ per\ tweet$), and average content similarity between a user's tweets (*tweets similarity*) have good discrimination powers. Account age, text-based features extracted from tweets, and ratio of number of URLs in recently posted top 20 tweets and number of the tweets ($|URL|\ per\ tweet$) have the best discrimination power. Overall, the proposed all features have positive discrimination power.

To further illustrate, Figure 3.4(a) presents the cumulative distributions of content similarity in tweets posted by each user class. It shows clear distinction among legitimate users, spammers and promoters. The content similarity in tweets of each spammer is the largest compared to the other classes because some of them post almost the same content or even duplicate tweets. Promoters have a goal of promoting something like online business, marketing and so on; naturally their tweets include common terms like the name of a product. Therefore, the content similar-

ity in their tweets is larger than legitimate users' one because legitimate users post tweets about their news such as what they are doing, where they are and so on. The content similarity in tweets of legitimate users is the smallest. Figure 3.4(b) shows the cumulative distributions of the average number of URLs in the tweets of each user. Tweets posted by legitimate users include the smallest number of URLs; not surprisingly, the majority of spammers and promoters post tweets with URLs. The curves of spammers and promoters are overlapped near 1 in the X axis, meaning that promotor and spammer behavior is closely coupled in our dataset.

Table 3.3 shows the performance results for the top 10 classifiers. Each of the top 10 classifiers achieved an accuracy greater than 82.7%, an F_1 measure over 0.82, and a false positive rate less than 10.3%. As in the case with MySpace, the meta classifiers (Decorate, LogitBoost, etc.) produced better performance than tree classifiers (BFTree and FT) and function-based classifiers (SimpleLogistic and LibSVM). The best classifier was Decorate, which obtained an accuracy of 88.98% accuracy, an F_1 measure of 0.888, and a 5.7% false positive rate. As in the MySpace analysis, we additionally considered different training mixtures of spam and legitimate training data (from 10% spam / 90% legitimate to 90% spam / 10% legitimate); we find that the classification metrics are robust across these changes in training data.

3.2.3.4 Summary

Based on our empirical study over both MySpace and Twitter, we find strong evidence that social honeypots can attract spam behaviors that are strongly correlated with observable features of the spammer's profiles and their activity in the network (e.g., tweet frequency). These results hold across two fundamentally different communities and confirm the hypothesis that spammers engage in behavior that is correlated with observable features that distinguish them from legitimate users. In

Table 3.3: Twitter – Performance results of top 10 classifiers

Weka Classifier	Accuracy	F_1	FP
Decorate	**88.98%**	**0.888**	**5.7%**
LogitBoost	87.86%	0.877	6.2%
HyperPipes	85.29%	0.846	8.1%
Bagging	84.56%	0.844	7.5%
RandomSubSpace	84.19%	0.837	8.3%
BFTree	83.82%	0.84	7.2%
FT	83.46%	0.832	8.3%
SimpleLogistic	83.46%	0.832	8.5%
LibSVM (SVM)	83.09%	0.825	10.2%
ClassificationViaRegression	82.72%	0.823	9.1%

addition, we find that some of these signals may be difficult for spammers to obscure (e.g., content containing a sales pitch or deceptive content), so that the results are encouraging for ongoing effective spam detection.

3.2.4 RC3: Large-Scale Social Spam Classification In the Wild

So far, we have seen that the deployed social honeypots can collect evidence of spam behavior, and that these behaviors are correlated with spam signals which can support automatic spam classification. In this final study, we explore whether these classifiers can be effectively deployed over large collections of unknown profiles (for which we have no assurances of the degree of spam or legitimate users). Concretely, we apply the developed classifiers for both MySpace and Twitter over datasets "in-the-wild" to better understand the promise of social honeypots in defending against new and emerging spam and zero-day spam attacks.

3.2.4.1 Data and Setup

For this final study, we considered two large datasets:

- *MySpace Dataset:* The first dataset is a crawl over MySpace, including about 1.5 million of public profiles collected in 2006 and 2007. A full description of

Table 3.4: Statistics of MySpace dataset

Public Profiles	Private Profiles	Total Profiles	Size
1,576,684	274,988	1,851,672	150GB

Table 3.5: Statistics of Twitter dataset

User Profiles	Tweets	Following	Followers	Size
215,345	4,040,415	51,650,754	65,904,253	11.3GB

this dataset and its characteristics is available in [20]. Table 3.4 summarizes statistics of this dataset.

- *Twitter Dataset:* We also collected a large dataset from Twitter for the period September 2 to September 9, 2009. We sampled the public timeline of Twitter (which publishes a random selection of new tweets every minute), collected usernames, and then used the Twitter API to collect each user's recently posted top 20 tweets, plus the user's following (friends) and followers' information. Table 3.5 presents statistics of this Twitter dataset. It consists of 215,345 user profiles, 4,040,415 tweets.

In both cases, the collected profiles are unseen to our system, meaning that we do not know ground truth as to whether a profile is spam or legitimate. As a result, the traditional classification metrics presented in the previous subsection would be infeasible to apply in this case. Rather than hand label millions of profiles, we adopted the spam precision metric to evaluate the quality of spam predictions. For spam precision, we evaluate only the predicted spammers (i.e., the profiles that the classifier labels as spam). Spam precision is defined as:

$$SpamPrecision = \frac{true\,positives}{true\,positives + false\,positives}$$

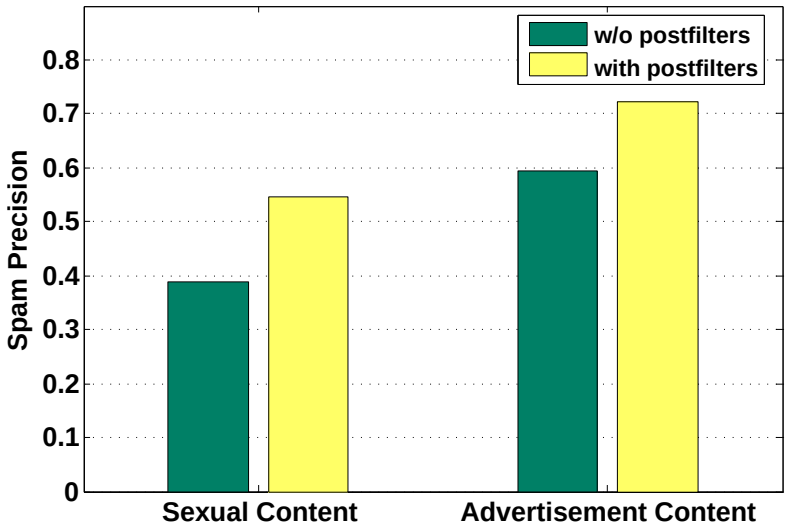


Figure 3.5: MySpace – Spam Precision

3.2.4.2 Finding Unknown Spam on MySpace

As in the previous subsection, we trained a classifier over a training set consisting of 388 legitimate profiles (labeled by us) and 627 deceptive spam profiles collected from social honeypots. In the interests of efficiency, we used the LibSVM classifier – an implementation of support vector machines – which is a widely popular classifier and classifies a large dataset quickly with high accuracy. Its classification time is faster than meta classifiers that proved successful in the previous experiments. We sampled from the 1.5 million public profiles a smaller test set of 43,000 profiles. We repeated this sampling procedure four times so we had four different test sets.

As we classified each of four test sets, a human inspector verified whether the newly found predicted spam was actually spam, added the new instances to the training set, and the process continued. In each test set, LibSVM classifier predicted about 30 users as spammers. In each subsequent iteration, we found that the spam precision increased.

Figure 3.5 shows the evaluation results of the fourth test set. The left two bars of the figure present spam precision based on sexual content. If an unseen profile is classified to a deceptive spam profile by LibSVM, and its "About Me" content includes sexual content, it will be considered as a real deceptive spam profile. The right two bars of the figure present spam precision based on advertisement content. If predicted spam profile's "About Me" content includes advertisement content, it will be considered as a real deceptive spam profile. Note that there are two results: with postfilter and without postfilter. We found that LibSVM incorrectly predicted spam labels for profiles containing special profile layout links, e.g., "click here to get a theme for your myspace" or "click here for myspace backgrounds", which are similar to spammer techniques for inserting links into spam profiles. These types of profile layout links are common on MySpace, which allows users to adjust their profile layouts. To correct these errors, we inserted a "postfilter" to check for these standard links and remove their spam labels.

As we can see, using postfilters improved about 40% spam precision in sexual content and about 21% in advertisement content. Detecting spammers whose profiles include advertisement content is easier than detecting spammers whose profile include sexual content. Even with the fairly good results (70% spam precision), the results are significantly worse than what we observed in the previous subsection over the controlled dataset. We attribute this decline in performance to the time-based mismatch between the harvested social honeypots and the large MySpace dataset. The honeypots were deployed in 2007, but the large MySpace data was crawled in 2006. As a result, the spam signatures developed from the honeypots have difficulty identifying spam in an earlier dataset when those spam signature may have not been in use at all. Even with these challenges, the results are fairly good.

As an example, Table 3.6 illustrates a legitimate-appearing profile in the first

Table 3.6: Example of "About Me" content in new deceptive spam profiles

"About Me" content
I moved to san diego 3 months ago with my boyfriend, well, ex-boyfriend now . . . one thing i did find is this webcam site. it pays pretty decent and the best part is that its really fun, too . . . , click here to visit me.

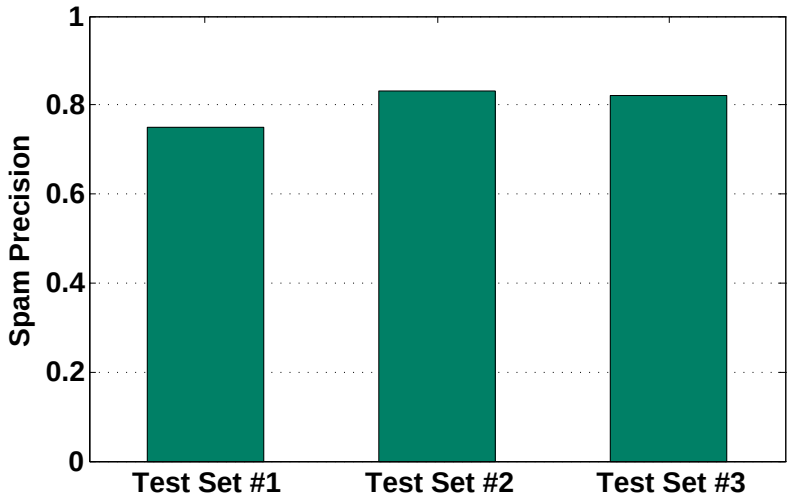


Figure 3.6: Twitter – Spam Precision

part of the "About Me" content, but then inserts a URL which links to an external site (usually porn or sexual sites) in the middle part or the last part of the "About Me" content.

3.2.4.3 Finding Unknown Spam on Twitter

Unlike the MySpace data mismatch, the social honeypots deployed on Twitter pre-date the large Twitter dataset collection. Hence, we are interested in this final experiment to discover if these honeypots can effectively detect new and emerging spam. For Twitter classification, we again relied on the training set consisting of 104 legitimate users' data, 61 spammers' data and 107 promoters' data which were used in Section 3.2.3. For prediction, we considered two cases: legitimate or spam+promoter.

We randomly selected a test set of 1,000 users' data from the total dataset of about 210,000 users. We repeated this process three times, resulting in three test sets. The feature generator produces the same features used in the previous subsection. We selected Decorate as a classifier because it showed the best performance in the previous Twitter study. Human inspectors view spam data predicted by the classifier, and then decide whether or not they are real spam data. They will add newly found spam data with labels (spam or non-spam) to the training set in order to iteratively improve the classifier's accuracy.

Figure 3.6 presents spam precision results obtained from the three test sets. In each test set, the Decorate classifier predicted about 20 users as spammers. We assessed whether the predicted spammers were real spammers. In the first iteration, spam precision was 0.75, nearly matching the performance of the controlled classifier reported in the previous subsection. By the third iteration, the spam precision was 0.82. We see in this experiment how the social honeypots provide strong ability to discover unknown spam; and as these newly discovered spammers are added to the training set, the classifier becomes more robust (resulting in the improvement from the first to the third iteration).

As an example, Figure 3.7 shows an example of newly found spammer. The spammer's tweets include URLs which link to sex search tool pages. It is interesting that the spammer has 205 followers, meaning that this spammer has successfully inserted himself into the social network without detection. We additionally found that about 20% of the users predicted to be spammers were *bots* that post tweets automatically using the Twitter API.

Based on this large-scale evaluation of spam "in-the-wild", we can see how social honeypots can enable effective social spam detection of previously unknown spam instances. Since spammers are constantly adapting, these initial results provide

Figure 3.7: An example of newly found spammer on Twitter

positive evidence of the robustness of the proposed approach.

3.3 A Long-Term Study of Content Polluters on Twitter

3.3.1 Introduction

In the previous section, we have presented the design and real-world evaluation of a novel social honeypot-based approach to detect individual social spammers. We have investigated techniques and developed effective tools for automatically detecting and filtering spammers who target social systems. Our empirical evaluation over both MySpace and Twitter has demonstrated the effectiveness and adaptability of the honeypot-based approach to social spam detection.

In contrast to the study in the previous section, in this section we present the first long-term study of social honeypots via a seven-month deployment of 60 honeypots on Twitter that resulted in the harvesting of 36,000 candidate content polluters. We provide a detailed examination of the harvested Twitter users, including an analysis of link payloads, user behavior over time, and followers/following network dynamics. We experimentally evaluate a wide range of features – including user demographics,

properties of the Twitter follower/following social graph, Tweet content, and temporal aspects of user behavior – to investigate the effectiveness of automatic content polluter identification, even in the presence of strategic polluter obfuscation. Finally, we empirically validate the social honeypot-derived classification framework on an alternative Twitter spam dataset, which shows the flexibility and effectiveness of the proposed approach.

3.3.2 Tempting Content Polluters

As we created social honeypot accounts in the previous section, we created and deployed 60 new social honeypot accounts on Twitter whose purpose is to pose as Twitter users, and report back what accounts follow or otherwise interact with them.

The system ran from December 30, 2009 to August 2, 2010. During that time the social honeypots tempted 36,043 Twitter users, 5,773 (24%) of which followed more than one honeypot. One user was tempted by twenty-seven different honeypots. After removing users who followed more than one honeypot, 23,869 users remained. Figure 3.8 shows the number of polluters tempted per day.

3.3.3 Who are the Harvested Twitter Users?

Our overall goal is to automatically attract content polluters via our social honeypots so that we can provide ongoing and dependable policing of the online community. Of course, a user identified by the social honeypot system is not necessarily a content polluter. Our intuition, however, is that given the behavior of the social honeypots there is no reason for a user who is not in violation of Twitter's rules to be tempted to message or follow them. Since social honeypot accounts post random messages and engage in none of the activities of legitimate users, it seems reasonable that the likelihood of a legitimate user being tempted to be similar, if not less, than the likelihood an error would be made in hand-labeling the type of users.

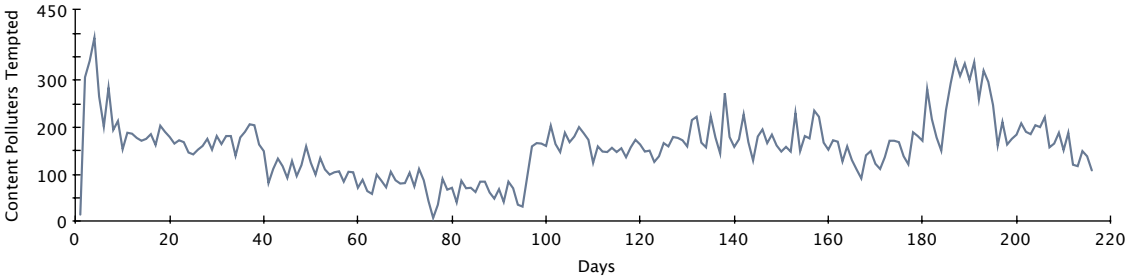


Figure 3.8: A chart of the number of content polluters tempted per day. On the fourth day of the study the honeypots were able to tempt a total of 391 content polluters, the most in a single day. The third highest single-day temptation was 339, which occurred on day 191.

3.3.3.1 Users Detected via Social Honeypots vs. Official Twitter Spammers

To support this intuition, we first investigated the 23,869 polluters the honeypots lured to see if any were considered as official violators of Twitter's terms of service [139]. We found that Twitter eventually suspended the accounts of 5,562 (or 23% of the total polluters identified by the social honeypots). We observe that of the 5,562, the average time between the honeypot tempting the polluter and the account being suspended was 18 days. In one case, the honeypot snared a polluter 204 days before Twitter terminated the account. In other words, the social honeypots identified polluters much earlier than through traditional Twitter spam detection methods (again, on average by 18 days). But what of the remaining 77% (18,307) of the polluters that were caught but not suspended by Twitter? Are these merely legitimate accounts that have been erroneously labeled as polluters?

3.3.3.2 Cluster Analysis of Harvested Users

To better understand who these harvested Twitter users are, we manually investigated them via cluster analysis. We used the Expectation-Maximization (EM) algorithm [33] and a set of features for representing each harvested Twitter user (de-

Table 3.7: Content polluter examples

Content Polluters	Tweets
Duplicate Spammer	T1: OFFICIAL PRESS RELEASE Limited To 10,000 "Platinum Founders" Reseller Licenses http://tinyurl.com/yd75xyy T2: OFFICIAL PRESS RELEASE Limited To 10,000 "Platinum Founders" Reseller Licenses http://tinyurl.com/yd75xyy
Duplicate @ Spammer	T1: #Follow @_anhran @PinkySparky @RestaurantsATL @combi31 @BBoomsma @TexMexAtl @DanielStoicaTax T2: #Follow @DeniseLescano @IsabelTrent @kxtramoney @PhoinixROTC44 @ATL_Events @HoldemTalkRadio
Malicious Promoter	T1: The Secret To Getting Lots Of Followers On Twitter http://bit.ly/6BiLk3 T2: Have Fun With Twitter - Twitter Marketing Software http://bit.ly/6ns0sc
Friend Infiltrator	T1: Thank you for the follows, from a newbie T2: @EstherK Yes I do and and thatnks for the follow

scribed more fully in the following subsection) to find groups of harvested users with similar appearances/behaviors. EM is a well-known clustering algorithm, and finds the best number of clusters, assigning a probability distribution about the clusters to each instance (each harvested user account). EM discovered nine clusters. We investigated each of the clusters, focusing on major clusters which included a large number of harvested users. Based on our observations, we grouped these users into four categories of content polluters (illustrated in Table 3.7):

- *Duplicate Spammers:* These content polluters post nearly identical tweets with or without links.

- *Duplicate @ Spammers:* These content polluters are similar to the Duplicate Spammers, in that they post tweets with a nearly identical content payload, but they also abuse Twitter's @username mechanism by randomly inserting a legitimate user's @username. In this way, a content polluter's tweet will be delivered to a legitimate user, even though the legitimate user does not follow

the content polluter.

- *Malicious Promoters*[‡]*:* These content polluters post tweets about online business, marketing, finance and so on. They have a lot of following and followers. Their posting approach is more sophisticated than other content polluters because they post legitimate tweets (e.g., greetings or expressing appreciation) between promoting tweets.

- *Friend Infiltrators:* Their profiles and tweets are seemingly legitimate, but they abuse the reciprocity in following relationships on Twitter. For example, if user A follows user B, then user B typically will follow user A as a courtesy. Previous literature [94, 147] has shown that reciprocity is prevalent in social networking web sites including Twitter. After they have a large number of followers, friend infiltrators begin engaging in spam activities (e.g., posting tweets containing commercial or pornographic content).

What we see is that although not suspended by Twitter, these accounts are engaging in aggressive promotion and negative behaviors, e.g., following a large number of users, and shortly dropping them, exclusively posting promotional material, posting pornographic material, and so on.

3.3.3.3 Followers and Following

We next investigated the properties of the collected content polluters, to explore what behaviors and properties these users displayed. First, we found that on average they followed 2,123 accounts, and the average number of followers they had was 2,163.

[‡]While product promotion is allowed by Twitter, accounts of this nature often are guilty of violating Twitter's definition of spam which includes if the account's updates consist mainly of links, and if the account repeatedly follow and unfollow other users or promotes third-party sites that claim to get you more followers.

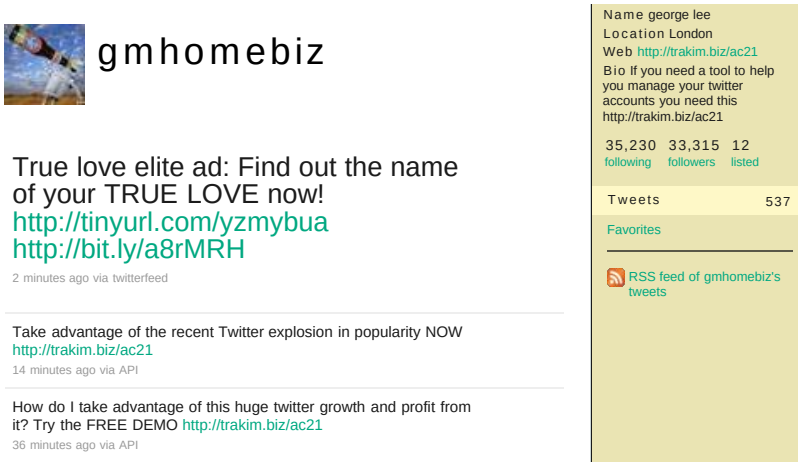


Figure 3.9: The Twitter homepage of gmhomebiz, a user tempted by our social honeypots

These numbers are higher than most legitimate users which only have between 100 and 1,000 followers and following counts [71, 147]. Figure 3.9 shows the account homepage of a content polluter the social honeypots tempted. It appears to be a legitimate user; the profile information has been fully completed, and the appearance of the page has been customized. However, this account is following 35,230 users, and has a following of 33,315. Those counts are drastically different from most legitimate users who typically follow fewer than 1,000 users.

3.3.3.4 Tweeting Activity

The discovered content polluters posted on average only four tweets per day. We assume the controllers of these accounts are aware that if they post a large number of tweets per day, they will be easily detected by Twitter and their accounts will be suspended. Instead, they post a few tweets per day attempting to mimic the pattern of a legitimate user. However, they cannot hide the large number of users they follow and the large number of users following them since their goal is to promote to a vast audience.

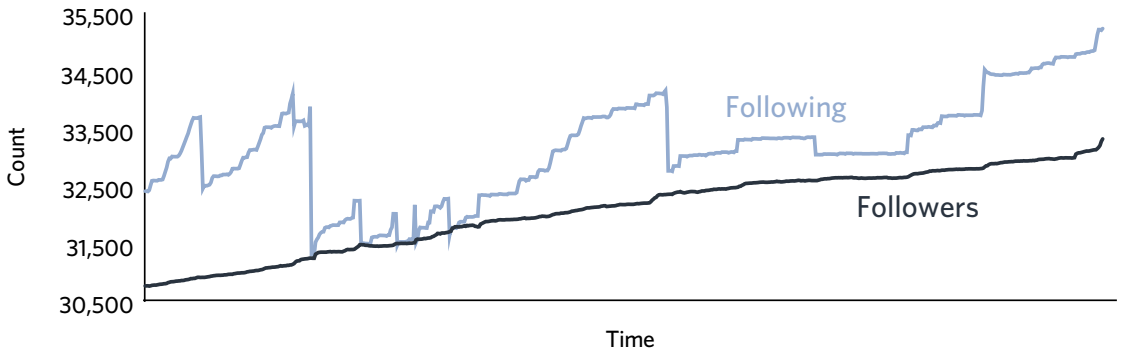


Figure 3.10: The graph shows the changing number of users following the *gmhomebiz* account and the number of users followed.

3.3.3.5 Behavior Over Time

This observation and intuition led us to investigate their temporal and historical profile information which includes the number of following and followers collected by our system once per hour, since they were tempted. The number of users the content polluters were following fluctuated significantly over time. Figure 3.10 presents a portion of the temporal information of the content polluter shown in Figure 3.9. This polluter manipulated the number of accounts it was following in order to achieve a balance between the number of following and followers, presumably to maintain a balance so that Twitter will not investigate and possibly suspend the account. To further illustrate, Figure 3.11 shows the change in the number of following and followers for two content polluters and two legitimate users.

3.3.3.6 Link Payloads

Twitter users often add an URL to the text of a tweet; thus allowing them to circumvent Twitter's 140 character limitation. Table 3.8 shows the five most frequently posted URLs, where we have converted shortened URLs (e.g., http://bit.ly/6BiLk3)

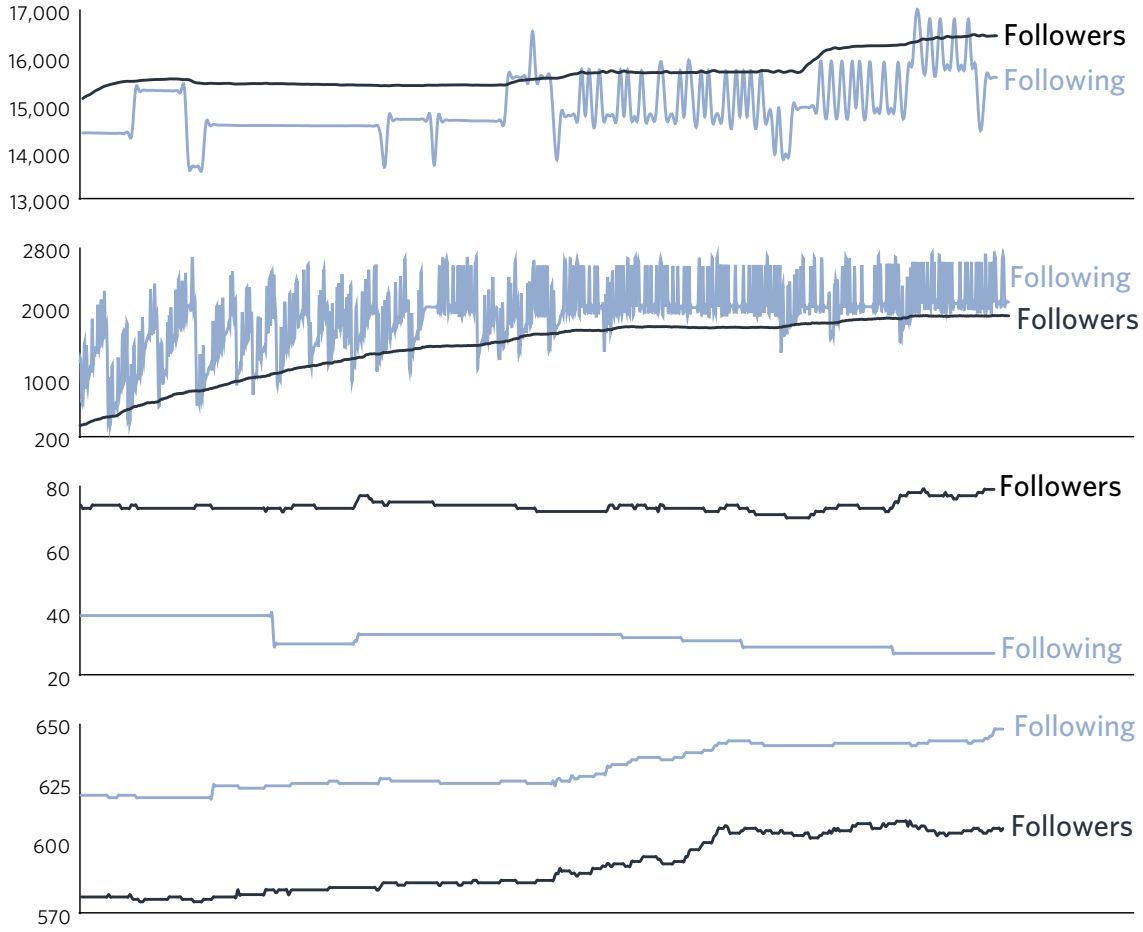


Figure 3.11: The top two graphs are of content polluter accounts. The bottom two are legitimate users. The accounts in the top two graphs are engaging in the act of "follower churn" [139].

to their original long form for easier understanding. Most linked to disreputable pages such as automatic promotion/bot software and phishing sites, with some links being inserted into hundreds of tweets in a clear attempt at link promotion.

3.3.4 Profiling Content Polluters

In this subsection, we aim to automatically "profile" Twitter-based content polluters by developing automatic classifiers for distinguishing between content polluters

Table 3.8: Top five URLs posted by Content Polluters

Freq.	URL	Linked Page
2,719	www.thetweettank.com	twitter bot software
2,348	shop.cooliohigh.com	sunglasses seller
2,227	friendfeed.com	social networking site
1,919	www.tweetsbot.com	twitter bot software
771	wefollow.com	twitter 3rd party site

and legitimate users. Do we find that content polluters engage in particular behaviors that make them clearly identifiable? Or do they strategically engage in behaviors (e.g., posting frequency, history of friends in the network) that make them "invisible" to automated detection methods? For example, we have seen that our harvested content polluters post ∼four tweets a day, which seems well within "normal" behavior (in contrast to email spammers who issue millions of spam emails).

3.3.4.1 Classification Approach and Metrics

To profile content polluters on Twitter, we follow a classification framework where the goal is to predict whether a candidate Twitter user u is a content polluter or a legitimate user. To build a classifier c

$$c : u \rightarrow \{polluter, legitimate\ user\}$$

we used the Weka machine learning toolkit [149] to test 30 classification algorithms, such as naive bayes, logistic regression, support vector machine (SVM) and tree-based algorithms, all with default values for all parameters using 10-fold cross-validation. 10-fold cross-validation involves dividing the original sample (data) into 10 equally-sized sub-samples, and performing 10 training and validation steps. In each step, 9 sub-samples are used as the training set and the remaining sub-sample is used as the

validation set. Each sub-sample is used as the validation set once.

Table 3.9: Dataset

Class	User Profiles	Tweets
Polluters	22,223	2,380,059
Legit Users	19,276	3,263,238

For training, we relied on a dataset[§] (summarized in Table 3.9) of content polluters extracted by the social honeypots and legitimate users sampled from Twitter.

- *Content Polluters:* We filtered the original 23,869 polluters collected by the social honeypots to exclude those that were (nearly) immediately identified and suspended by Twitter. The reason why we dropped these short-lived polluters is that Twitter already has their own solution for the short-lived polluters, and our target is content polluters that are alive for a long time (at least two hours, since our system tempted them). For the remaining 22,223 polluters, we collected their 200 most recent tweets, their following and follower graph, and their temporal and historical profile information including the number of following and followers collected by our system once per hour since they were tempted by a honeypot.

- *Legitimate Users:* To gather a set of legitimate users, we randomly sampled 19,297 Twitter users. Since we have no guarantees that these sampled users are indeed legitimate users (and not polluters) and hand labeling is both time consuming and error-prone, we monitored the accounts for three months to see if they were still active and not suspended by Twitter. After three months, we

[§]Available at http://infolab.tamu.edu/data

found that 19,276 users were still active and so we labeled them as legitimate users. Even though there is chance of a false positive in the legitimate user set, the results of our classifier study should give us at worst a lower bound on accuracy since the introduction of possible noise in the training set would only degrade our results.

Table 3.10: Confusion matrix

		Predicted	
		Polluter	Legitimate
Actual	Polluter	a	b
	Legit User	c	d

We compute precision, recall, F-measure, accuracy, area under the ROC curve (AUC), false negatives (FNs) and false positives (FPs) as metrics to evaluate our classifier. In the confusion matrix, Table 3.10, a represents the number of correctly classified polluters, b (called FNs) represents the number of polluters misclassified as legitimate users, c (called FPs) represents the number of legitimate users misclassified as polluters, and d represents the number of correctly classified legitimate users. The precision (P) of the polluter class is $a/(a+c)$ in the table. The recall (R) of the polluter class is $a/(a+b)$. F_1 measure of the polluter class is $2PR/(P+R)$. The accuracy means the fraction of correct classifications and is $(a+d)/(a+b+c+d)$. AUC is a measure showing classification performance. The higher AUC is, the better classification performance is. 1 AUC value means a perfect performance.

3.3.5 Features

The quality of a classifier is dependent on the discriminative power of the features. Based on our previous observations, we created a wide variety of features belonging

to one of four groups: User Demographics (**UD**): features extracted from descriptive information about a user and his account; User Friendship Networks (**UFN**): features extracted from friendship information such as the number of following and followers; User Content (**UC**): features extracted from posted tweets; and User History (**UH**): features extracted from a user's temporal and historical profile information.

The specific features for each feature group are:

UD the length of the screen name, and the length of description

UD the longevity of the account

UFN the number of following, and the number of followers

UFN the ratio of the number of following and followers

UFN the percentage of bidirectional friends:

$$\frac{|following \cap followers|}{|following|} \; and \; \frac{|following \cap followers|}{|followers|}$$

UFN the standard deviation of unique numerical IDs of following

UFN the standard deviation of unique numerical IDs of followers

UC the number of posted tweets

UC the number of posted tweets per day

UC $|links|$ *in tweets* / $|tweets|$

UC $|unique\ links|$ *in tweets* / $|tweets|$

UC $|@username|$ *in tweets* / $|tweets|$

UC $|unique\ @username|\ in\ tweets\ /\ |tweets|$

@username features can detect a content polluter posting tweets with various @usernames.

UC the average content similarity over all pairs of tweets posted by a user

$$\sum_{a,b \in \text{set of pairs in tweets}} \frac{similarity(a,b)}{|\text{set of pairs in tweets}|}$$

UC the ZIP compression ratio of posted tweets:

$$\frac{uncompressed\ size\ of\ tweets}{compressed\ size\ of\ tweets}$$

The compression ratio can detect a content polluter posting nearly identical tweets because when we compress its tweets, their compressed size is significantly decreased.

UH the change rate of number of following obtained by a user's temporal and historical information:

$$\sqrt{\frac{1}{n-1}\sum_{i=1}^{n-1}(f_{i+1}-f_i)}$$

where n is the total number of recorded temporal and historical information, and f_i means the number of following of the user extracted in ith temporal and historical information.

We computed the χ^2 value [159] of each of the features to determine their discriminative power. The larger the χ^2 value is, the higher discriminative power the corresponding feature has. The results showed all of our features had positive discrimination power, though with different relative strengths. Table 3.11 shows the top 10 features with χ^2 value and average feature values of polluters and legitimate users. The standard deviation of numerical IDs of following returned the highest χ^2

Table 3.11: Top 10 features with χ^2 value and average feature values of polluters and legitimate users

Feature	χ^2 value	Polluters	Legitimate Users
standard deviation of following	26,708	35,620,487	19,368,858
the change rate of $\|following\|$	23,299	29.6	1.5
standard deviation of followers	22,491	35,330,087	22,047,831
$\|following\|$	15,673	2,212	327
longevity of the account	15,467	279	506
ratio of the number of following and followers	12,115	11.1	1.5
$\|links\|$ per tweet	11,827	0.65	0.21
$\|@username\|$ *in tweets* / $\|tweets\|$	9,039	0.2	0.51
$\|unique\ @username\|$ *in tweets* / $\|tweets\|$	8,859	0.12	0.17
$\|unique\ links\|$ per tweet	7,685	0.48	0.18

value because polluters follow users randomly. Content polluters' following standard deviation is much higher than legitimate users' standard deviation. The change rate of number of following outperforms other features because polluters increased the number of following in order to contact a larger number of users and promote to them, and then decreased the number of following in order to maintain a balance between the number of following and followers to avoid being suspended by Twitter. The average change rate of polluters was 29.6, while the average change rate of legitimate users was 1.5. Like the standard deviation of following, polluters' follower standard deviation is much higher than legitimate users' follower standard deviation. Polluters had larger followings than legitimate users, and shorter longevity than legitimate users.

3.3.5.1 Classification Results

Using the classification setup described above and these feature groups, we tested 30 classification algorithms using the Weka machine learning toolkit [149]. Across most classifiers (25 of the 30 tested), we find that the results are consistent, with

accuracy ranging from 95% to 98%, indicating that the strength of classification lies mainly in the choice of features and is relatively stable across choice of particular classifier. For the other 5 of the 30 tested, accuracy ranges from 89% to under 95%. Tree-based classifiers showed the highest accuracy results. In particular, Random Forest produced the highest accuracy as shown in Table 3.12. Its accuracy was 98.42% and 0.984 F_1 measure.

Table 3.12: The performance result of Random Forest

Classifier	Accuracy	F_1	AUC	FNs	FPs
Random Forest	98.42%	0.984	0.998	301	354

We additionally considered different training mixtures of polluters and legitimate users, ranging from 1% polluter and 99% legitimate to 99% polluter and 1% legitimate. We find that the classification quality is robust across these training mixtures.

Table 3.13: Boosting and bagging of the Random Forest classifier

Classifier	Accuracy	F_1	AUC	FNs	FPs
Boosting	**98.62%**	**0.986**	0.995	287	**287**
Bagging	98.57%	**0.986**	**0.999**	**248**	345

In order to improve the Random Forest classifier, we additionally applied standard boosting [41] and bagging [13] techniques. Both create multiple classifiers and combine their results by voting to form a composite classifier. Table 3.13 shows the results. Both outperformed the original Random Forest. Boosting of Random Forest classifier produced the best result, 98.62% accuracy and 0.986 F_1 measure. These results provide strong evidence that social honeypots attract polluter behaviors that

are strongly correlated with observable features of their profiles and their activity in the network.

3.3.5.2 Handling Strategic Polluter Obfuscation

As time passes, the designers of content polluters may discover which features are signaling to our system that their polluters are not legitimate Twitter users, and so these features may lose their power to effectively profile and detect polluters. Thus, we tested the robustness of the polluter-based classifier by constraining the classifier to have access only to certain feature groups, mimicking the scenario in which content polluters strategically target our system (and, hence, entire feature groups lose their ability to distinguish polluters). We also considered scenarios in which one, two or even three entire feature groups lose their effectiveness.

Table 3.14: The performance results of various feature group combinations

Feature Set	Accuracy	F_1	AUC	FNs	FPs
UD	76.17%	0.762	0.839	6,007	3,882
UH	85.34%	0.854	0.899	4,130	1,950
UC	86.39%	0.864	0.932	2,811	2,837
UFN	96.46%	0.965	0.992	510	958
UD+UC	88.61%	0.886	0.953	2,469	2,256
UD+UH	92.45%	0.925	0.967	1,743	1,389
UC+UH	94.38%	0.944	0.979	1,111	1,221
UFN+UH	97.11%	0.971	0.994	496	702
UD+UFN	97.50%	0.975	0.995	437	597
UFN+UC	97.92%	0.979	0.996	413	448
UD+UC+UH	95.42%	0.954	0.985	878	1,022
UD+UFN+UH	97.78%	0.978	0.996	395	524
UFN+UC+UH	98.13%	0.981	**0.997**	361	413
UD+UFN+UC	**98.26%**	**0.983**	**0.997**	**333**	**388**

Following the classification setup described above, we trained the content polluter classifier based on different feature group combinations, resulting in the effectiveness

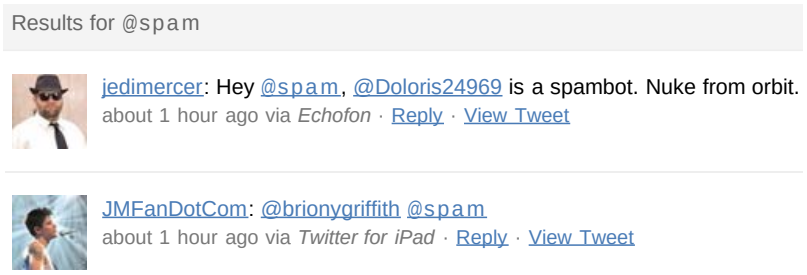


Figure 3.12: "@spam" search results

measures shown in Table 3.14. First, in the extreme case in which only a single feature group is available (either User Demographics, User Friendship Network, User Content, or User History), we see results ranging from 76.17% accuracy to 96.46% accuracy. Considering pairs of features, we can see that the classification accuracy ranges from 88.61% to 97.92%. Even in the case when the content polluters obfuscate the single most distinguishing signal (User Friendship Network), the UD+UC+UH case resulted in 95.42% accuracy and nearly equaled the performance across the other measures. Together, these results indicate that the signals distinguishing content polluters from legitimate users are not tied to a single "super-feature", but are a composite of multiple inter-related features. This gives us confidence going forward that content polluters cannot trivially change a single behavior (e.g., by manipulating their follower-following ratio) and become invisible. Rather they must become more like legitimate users, which necessarily decreases the effectiveness and impact of their pollution attempts (e.g., by reducing the number of links per tweet, reducing the number of *@username* per tweet, and so on).

3.3.5.3 Validation with Twitter @spam

Complicating the effective profiling of content polluters is the potential mismatch between models built on polluters harvested by social honeypots and for content pol-

luters in Twitter-at-large. We have seen that the framework presented in this chapter is effective at tempting large amounts of content polluters and at discriminating between polluters and legitimate users. However, it could be argued that there is inherent bias in the testing framework we have employed since the capacity of the classification framework to distinguish polluters (even with 10-fold cross validation) is linked to the collection method via social honeypots or that we are guilty of overfitting the models. Thus, we also evaluated the quality of our approach by applying the learned content polluter models to a test set that was collected *entirely separately* from our Twitter-based social honeypots.

To collect a content polluter dataset orthogonally from the honeypot framework, we monitored Twitter's spam reporting channel over a four month period using the Twitter search API. Twitter supports user-based spam reporting via a special @spam Twitter account which allows users to report suspicious or malicious users to @spam. Figure 3.12 illustrates a sample @spam search result. In the first result, *jedimercer* reported *Doloris24969* as a suspicious account to @spam account. Twitter investigates the reported user and if Twitter itself determines that the user has engaged in harmful activities, only then is the account suspended. If we find that our approach can effectively identify spam accounts from this alternative source, we will have confidence in the robustness and wide applicability of our results. Accounts suspended in this way may behave differently from the ones detected by our honeypots (e.g., they may never follow another account as our honeypots require).

Concretely, we constructed our orthogonal test set by searching for tweets containing "@spam" once per 20 minutes, and extracted the user account names (*@username*) listed in the tweets. When each user was reported to @spam, we collected their descriptive profile information, friendship information, tweets, and temporal and historical profile information. We continued to monitor these reported candi-

date spam accounts to identify only those that were actually suspended by Twitter (in effect, throwing away all of the false positives reported by users but not subsequently found to be spammers by Twitter itself). The four month observation period led to a total of 2,833 suspended accounts.

Following the classifier setup described in the previous subsection, we trained a Random Forest classifier using the content polluter data collected by our social honeypots and the set of legitimate users (recall Table 3.9). The trained classifier predicted class labels (content polluter or legitimate user) for the 2,833 suspended users in the separate @spam testing set.

We find that our approach leads to 96.75% accuracy and 0.983 F_1 measure over these @spam profiles. When we applied bagging to the Random Forest classifier, we achieved an even better result, 98.37% accuracy and 0.992 F_1 measure. We did not compute AUC because the test set does not include legitimate users. These results indicate that there is a strong capacity of our approach to detect harmful users on Twitter, even if they have not been directly discovered by our social honeypots.

To investigate the cases in which our classifier did not perform well (for the 46 spammers who were misclassified as legitimate users), we manually examined their profiles, friendship networks, and historical behavior. In all cases, the misclassified users have a low standard deviation of numerical IDs of following and followers (which was a strong discriminating feature in our content polluter study). Most of these users were quickly suspended by Twitter after they were first reported, meaning that the historical and temporal profile features were not available to our system. For those users for which we did have sufficient historical and temporal profile information, most engaged in widespread *@username* messages to contact many users rather than directly following users.

3.4 Summary

In this chapter, we have presented the design and real-world evaluation of a novel social honeypot-based approach to detect social spammers and content polluters. First, we have investigated techniques and developed effective tools for automatically detecting and filtering spammers who target social systems. By focusing on two different communities, we have seen how the general principles of (i) social honeypot deployment, (ii) robust spam profile generation, and (iii) adaptive and ongoing spam detection can effectively harvest spam profiles and support the automatic generation of spam signatures for detecting new and unknown spam. Our empirical evaluation over both MySpace and Twitter has demonstrated the effectiveness and adaptability of the honeypot-based approach to social spam detection. Second, we have designed and evaluated a system for automatically detecting and profiling content polluters on Twitter. During our seven-month long study we have shown how effectively our system lures many abusive Twitter accounts into following our collection of social honeypots. We see how these content polluters reveal key distinguishing characteristics in their behavior, leading to the development of robust classifiers.

4. EXTRACTING COORDINATED CAMPAIGNS*

4.1 Introduction

In this chapter, we move from individual accounts that threaten social systems to coordinated campaigns.

There is growing evidence that tightly-organized strategic campaigns are growing in significance [95, 143]. One example is the development of sites like SubvertAndProfit (www.subvertandprofit.com), which claims to have access to "25,000 users who earn money by viewing, voting, fanning, rating, or posting assigned tasks" across social media sites. Related services can be found at fansandinvites.com, socioniks.com, and usocial.net. Even within the great firewalls of China, we have witnessed the emergence of the so-called "Wang Luo Shui Jun" or "Online Water Army" (e.g., http://shuijunwang.com). According to a recent CCTV report [21], online mercenaries in China help their customers by (i) promoting a specific product, company, person or message; (ii) smearing the competitor or adversary or competitors' products or services; or (iii) deleting unfavorable posts or news articles. Most online "mercenaries" work part-time and are paid around 5 US cents per action.

User-driven campaigns – often linked by common "talking points" – appear to be growing in significance and reach with the commensurate rise of massive-scale social systems. However, there has been little research in detecting these campaigns "in the wild". While there has been some progress in detecting isolated instances of long-form fake reviews (e.g., to promote books on Amazon), of URL-based spam

*Reprinted with permission from "Campaign Extraction from Social Media" by Kyumin Lee, James Caverlee, Zhiyuan Cheng, and Daniel Z. Sui, 2013. *ACM Transactions on Intelligent Systems and Technology*, Copyright 2013 by ACM. Reprinted with permission from "Content-Driven Detection of Campaigns in Social Media" by Kyumin Lee, James Caverlee, Zhiyuan Cheng, and Daniel Z. Sui, 2011. *Proceedings of the 20th ACM international conference on Information and knowledge management*, 551-556, Copyright 2011 by ACM.

in social media, and in manipulating recommender systems [44, 57, 73, 82, 91, 92, 100, 116, 128, 156], there is a significant need for new methods to support web-scale detection of campaigns in social media.

Hence, we focus in this chapter on detecting one particular kind of coordinated campaign – those that rely on "free text" posts, like those found on blogs, comments, forum postings, and short status updates (like on Twitter and Facebook). For our purposes, a campaign is a collection of users and their posts bound together by some common objective, e.g., promoting a product, criticizing a politician, or inserting disinformation into an online discussion. Our goal is to link messages with common "talking points" and then extract multi-message campaigns from large-scale social media. Detecting these campaigns is especially challenging considering the size of popular social media sites like Facebook and Twitter with 100s of millions of unique users and the inherent lack of context in short posts.

Concretely, we propose and evaluate a content-based approach for identifying campaigns from the massive scale of real-time social systems. The content-driven framework is designed to effectively link free text posts with common "talking points" and then extract campaigns from large-scale social media. Note that text posts containing common "talking points" means the contents of the posts are similar or the same. We find that over millions of Twitter messages, the proposed framework can identify 100s of coordinated campaigns, ranging in size up to several hundred messages per campaign. The campaigns themselves range from innocuous celebrity support (e.g., fans retweeting a celebrity's messages) to aggressive spam and promotion campaigns (in which handfuls of participants post hundreds of messages with malicious URLs). Through an experimental study over millions of Twitter messages we identify five major types of campaigns – Spam, Promotion, Template, News, and Celebrity campaigns – and we show how these campaigns may be extracted with

high precision and recall. We also find that the less organic campaigns (e.g., Spam and Promotion) tend to be driven by a higher ratio of messages to participants (corresponding to a handful of accounts "pumping" messages into the system). Based on this observation, we propose and evaluate a user-centric campaign detection approach. By aggregating the messages posted by a single user, we find that the method can successfully discover cross-user correlations not captured at the individual message level (e.g., for two users posting a sequence of correlated messages), resulting in more robust campaign detection. In addition, we analyze each campaign type's temporal behavior to see the possibility to automatically determine a campaign's campaign type.

4.2 Content-Driven Campaign Detection

In this section, we describe the problem of campaign detection in social media, introduce the data, and outline the metrics for measuring effective campaign detection.

4.2.1 Problem Statement

We consider a collection of n participants across social media sites $U = \{u_1, u_2, \ldots, u_n\}$, where each participant u_i may post a time-ordered list of k messages $M_{u_i} = \{m_{i1}, m_{i2}, \ldots, m_{ik}\}$. Our hypothesis is that among these messages and users, there may exist *coordinated campaigns*.

Given the set of users U, a *campaign* M_c can be defined as a collection of messages and the users who posted the messages: $M_c = \{m_{ij}, u_i | u_i \in U \cap m_{ij} \in M_{u_i} \cap theme(m_{ij}) \in t_k\}$ such that the campaign messages belong to a coherent theme t_k. Themes are human-defined logical assignments to messages and application dependent. For example, in the context of spam detection, a campaign may be defined as a collection of messages with a common target product (e.g., Viagra).

In the context of astroturf, a campaign may be defined as a collection of messages promoting a particular viewpoint (e.g., the veracity of climate change). Additionally, depending on the context, a message may belong to one or multiple themes. For the purposes of this chapter and to focus our scope of inquiry, we consider as a theme all messages sharing similar "talking points" as determined by a set of human judges.

4.2.2 Data

To evaluate the quality of a campaign detection approach, we would ideally have access to a large-scale "gold set" of known campaigns in social media. While researchers have published benchmarks for spam webpages [145, 136], ad-hoc text retrieval [142], and other types of applications [135, 24, 77], we are not aware of any standard social media campaign dataset. Hence, we take in this chapter a twofold approach for message level campaign detection: (i) a small-scale validation over hand-labeled data; and (ii) a large-scale validation over 1.5 million Twitter messages for which ground truth is not known.

- CD_{Small}: First, we sample a small collection of messages (1,912) posted to Twitter in October 2010. Over this small *campaign dataset* (CD_{Small}), two judges labeled all pairs of the 1,912 tweets as sharing similar "talking points" or not, finding 298 pairs of messages sharing similar "talking points". Based on these initial labels, the judges considered all combinations of messages that may form campaigns consisting of four messages or more, and found 11 campaigns ranging in size from four messages to eight messages. While small in size, this hand-labeled dataset allows us to evaluate the precision and recall of several campaign detection methods.

- CD_{Large}: Second, we supplement the small dataset with a large collection of messages (1.5 million) posted to Twitter between October 1 and October 7,

2010. We sampled these messages using Twitter's Streaming API, resulting in a representative random sample of Twitter messages. Over this large *campaign dataset* (CD_{Large}), we can test the precision of the campaign detection methods and investigate the types of campaigns that are prevalent in-the-wild. Since we do not have ground truth knowledge of all campaigns in this dataset, our analysis will focus on the campaigns detected for which we can hand-label as actual campaigns or not.

Additionally, we also consider a *user-based* dataset, in which all of the messages associated with a single user are aggregated:

- CD_{User} Since the datasets CD_{Small} and CD_{Large} are collected by a random sample method from Twitter (meaning most users were represented by only one or two messages), we collected a user-focused dataset from Twitter consisting of 90,046 user profiles with at least 20 English-language messages, resulting in 1.8 million total messages.

4.2.3 Metrics

To measure the effectiveness of a campaign detection method, we use variations of average precision, average recall, and the average F_1 measure. The average precision (AP) for a campaign detection method is defined as:

$$AP = \frac{1}{n} \sum_{i=1}^{n} \frac{\max CommonMessages(PC_i, TCs)}{|PC_i|}$$

where n is the total number of predicted campaigns by the campaign detection method, PC is a predicted campaign, and TC is an actual (true) campaign. $MaxCommonMessage$ function returns the maximum of the number of common messages in both the predicted campaign i (PC_i) and each of the actual (true)

campaigns (TCs). For example, suppose a campaign detection method identifies a three-message campaign: $\{m_1, m_{10}, m_{30}\}$. Suppose there are two actual campaigns with at least one message in common: $\{m_{30}, m_{38}, m_{40}\}$ and $\{m_1, m_{10}, m_{35}, m_{50}, m_{61}\}$. Then the Precision is $max(2,1)/3 = 2/3$. In the aggregate, this individual precision will be averaged with all n predicted campaigns.

Similarly, we can define the average recall (AR) as:

$$AR = \frac{1}{n}\sum_{i=1}^{n}\frac{\max CommonMessages(PC_i, TCs)}{|TC_j|}$$

where n is the number of the predicted campaigns, and TC_j is a true campaign which has the largest common messages with the predicted campaign i (PC_i). Continuing the example from above, the Recall would be $max(2,1)/5 = 2/5$.

Finally, we can combine precision and recall as the average F_1 measure (AF):

$$AF_1 = \frac{2 * AP * AR}{AP + AR}$$

An effective campaign detection approach should identify predicted campaigns that are composed primarily of a single actual campaign (i.e., have high precision) and that contain most of the messages that actually belong to the campaign (i.e., have high recall). A method that has high precision but low recall will result in only partial coverage of all campaigns available (which could be especially disastrous in the case of spam or promotional campaigns that should be filtered). A method that has low precision but high recall may identify nearly all messages that belong to campaigns but at the risk of mislabeling non-campaign messages (resulting in false positives, which could correspond to mis-labeled legitimate messages as belonging to spam campaigns).

4.3 Campaign Detection: Framework and Methods

In this section, we describe the high-level approach for extracting campaigns from social media, present the message and user level campaign detection in detail, and discuss a MapReduce-based implementation for efficient campaign detection.

4.3.1 Overall Approach

To detect coordinated campaigns, we explore in this chapter several content-based approaches for identifying campaigns. Our goal is to find methods that can balance both precision and recall for effective campaign detection. In particular, we propose a content-driven campaign detection approach that views social media from two perspectives:

- *Message Level:* In the first perspective, we view each message as a potential member of a campaign. Our goal is to identify a campaign as a collection of its constituent messages. In this way, we can identify related message as shown in Figure 4.1. Given a set of messages (6 messages in the example), our goal is to build a message graph in which a node represents a message and if the similarity of a pair of messages is larger than a threshold (τ) then an edge exists between the pair of messages. Note that the similarity of a pair of messages means how much the pair of messages is similar in terms of number of common tokens, and a token can be defined as a n-gram word or n-gram character depending on a message similarity identification algorithm. In this way, we can identity significant subgraphs as campaigns, which should reflect multiple messages sharing the same key "talking points".

- *User Level:* In the second perspective, rather than viewing the message as the core component of a campaign, we view *each user* as a potential member

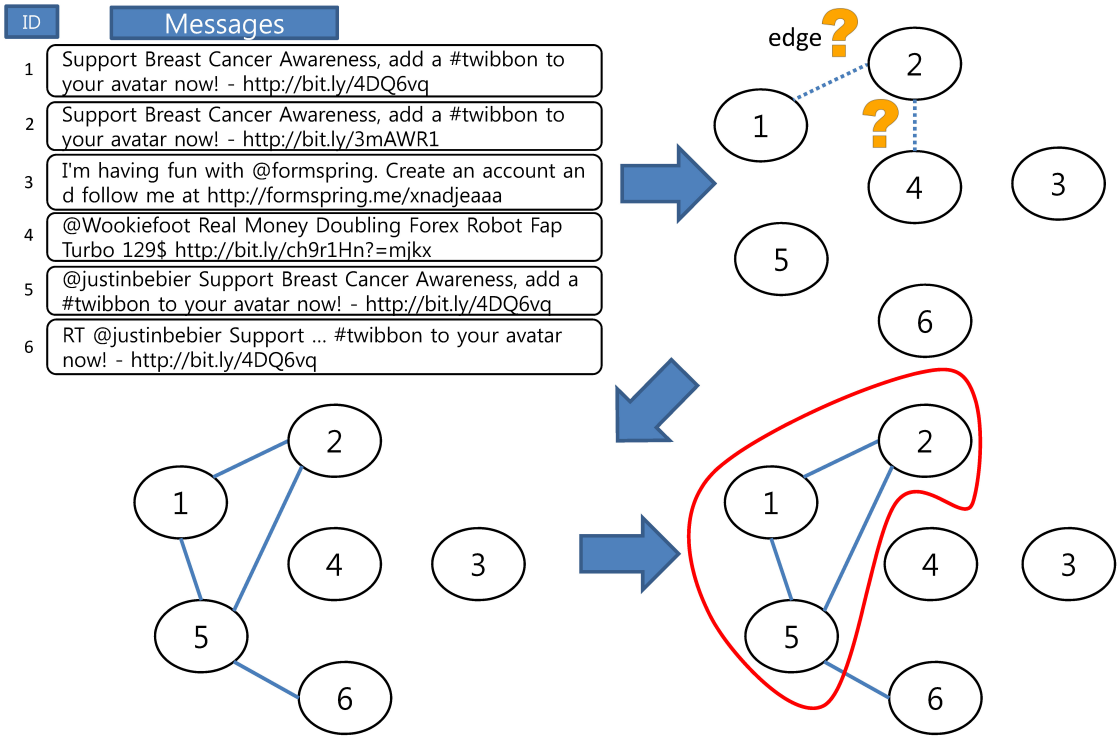


Figure 4.1: Overall approach showing how to identify campaigns given a list of messages

of a campaign. In this way, a campaign is composed of constituent users. This second perspective may be more reasonable in the case of campaigns that span multiple messages posted by a single user, or in the case of campaigns in which evidence of the campaign is clear at the user-level but perhaps not at the individual message level (say, in cases of 3 spam accounts that post similar messages in the aggregate, although no two individual messages may share the same talking points). For this perspective, we construct a graph, but where nodes represent users and their aggregated messages. Edges exist between users based on some overall measure of their similarity.

In the following, we detail these two approaches – at the message level and at the user level – in great detail.

4.3.2 Message Level Campaign Detection

For the task of message level campaign detection, we consider a graph-based framework, where we model messages in social media as a *message graph*. Each node in the message graph corresponds to a message; edges correspond to some reasonable notion of content-based correlation between messages, corresponding to pairs of messages with similar "talking points." Formally, we have:

Definition 1 (Message Graph)**.** *A message graph is a graph* $G = (V, E)$ *where every message in* M *corresponds to a vertex* m_{ix} *in the vertex set* V*. An edge* $(m_{ix}, m_{jy}) \in E$ *exists for every pair of messages* (m_{ix}, m_{jy}) *where* $corr(m_{ix}, m_{jy}) > \tau$*, for a measure of correlation and some parameter* τ*.*

A message graph which links unrelated messages will necessarily result in poor campaign detection (by introducing spurious links). Traditional information retrieval approaches for document similarity (e.g., cosine similarity [88], KL-divergence [89]) as well as efficient near-duplicate detection methods (e.g., Shingling [15], I-Match [25] and SpotSigs [130]) have typically not been optimized for the kind of short posts of highly-variable quality common in many social media sites (including Facebook and Twitter). Concretely we consider six approaches for measuring whether messages share similar "talking points":

- *Unigram Overlap:* The baseline unigram approach considered two messages to be correlated if they have higher Jaccard similarity than a threshold after we extract unigrams from each message and compute Jaccard similarity. The Jaccard coefficient between the unigrams of each pair of messages A and B is

used to measure the similarity of the pair of messages:

$$Jaccard(A, B) = \frac{|A \cap B|}{|A \cup B|} \leq \frac{min(|A|, |B|)}{max(|A|, |B|)}$$

- *Edit Distance:* An alternative is to consider the edit distance between two messages, that is, two messages are correlated if the number of edits to transform one message into the other is less than some threshold value. Concretely, we adopt the Levenshtein distance as a metric for measuring the amount of difference between two messages [81]. The distance is the minimum number of edits required in order to transform one message into the other.

- *Euclidean Distance:* Another similarity metric is Euclidean distance which is the length of the line segment connecting two vectors (two messages in this context). First convert messages to vectors in the vector space model and then compute their distance. The smaller Euclidean distance between two messages is, the more similar they are.

- *Shingling:* As an exemplar of near-duplicate detection, Broder's shingling algorithm [15] views a document d as a sequence of words $w_1\ w_2\ w_3\ \ldots\ w_n$, where n is the number of words in d. It extracts unique k-grams $\{g_1, g_2, \ldots, g_m\}$, such that m is the number of unique k-grams. For easy processing and reduction of storage usage, each g_i is encoded by 64-bit Rabin fingerprints F. The encoded value is called a shingle. Now, d's shingles S = $\{s_1, s_2, s_3, \ldots s_m\}$, such that s_i is a shingle (i.e., a signature) and $s_i = F(g_i)$. The Jaccard coefficient between the shingles of each pair of documents A and B is used to measure the similarity of the pair of documents.

  If the similarity score of a pair of documents (messages) is higher than a thresh-

old, they will be considered as near-duplicates (and hence, correlated messages for our purposes).

- *I-Match:* In contrast to Shingling, the I-Match [25] approach explicitly leverages the relative frequency of terms across messages. First, it defines an I-Match lexicon L based on a message frequency of each term in a collection of documents (i.e., Twitter messages). Usually, L consists of a bag of words (i.e., terms or unigrams) which have mid-idf values in the collection. I-Match extracts unigrams U from a document d and only use some unigrams P, which have mid-idf values in the collection (i.e., $P = L \cap U$). The idea behind this approach is that infrequent and too frequent terms are not helpful to detect near duplicate documents. Then, I-Match sorts P and concatenates it in order to make a single string, which is then encoded to a single hash value h by SHA-1; in our case, pairs of messages with identical hash values shall be considered correlated messages.

- *SpotSigs:* The final approach we consider is SpotSigs [130], which observes that noisy content, such as navigational banners and advertisements in web pages, may result in poor performance of traditional Shingling-based methods. By observing that stopwords rarely occur in the noisy content, SpotSigs scans a document to find stopwords as antecedents (anchors), and extracts special k-grams called "spot signatures", one of which consists of an antecedent and a k-gram after the antecedent, excluding stopwords. A hash function is applied to detect identical duplicates.

It is of course an open question how well each of these methods performs toward the ultimate goal of identifying campaigns in social media. Hence, we shall investigate experimentally in Section 4.4 each of these approaches for determining pairwise

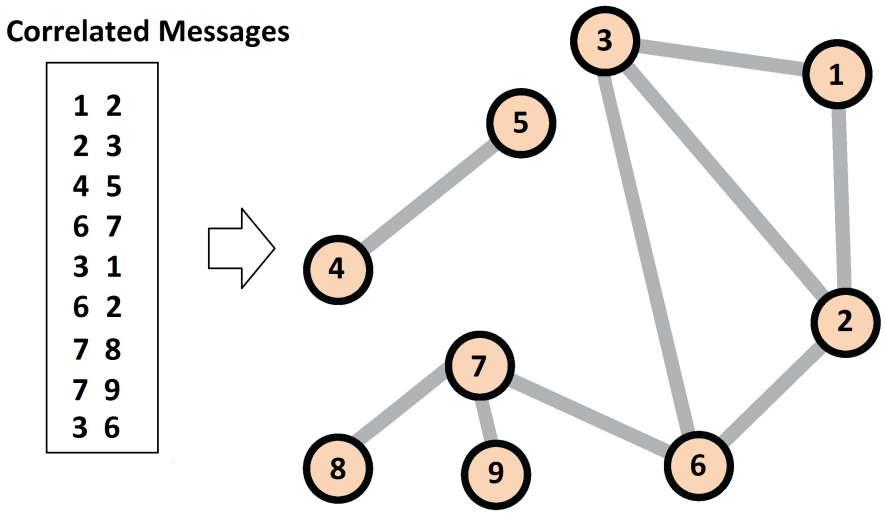


Figure 4.2: In a messages graph, a node represents a message and there exists an edge between correlated messages. This figure shows an example of a message graph before extracting campaigns.

message correlation which guides the formation of the message graph.

Given a message graph, we propose to explore three graph-based approaches for extracting campaigns:

- (i) loose extraction;
- (ii) strict extraction; and
- (iii) cohesive extraction.

Experimentally, we compare these graph-based approaches versus a traditional k-means clustering approach and reach poor results for clustering as compared to the graph methods. For now, we focus our attention on extracting content-driven campaigns via graph mining.

4.3.2.1 Loose Campaign Extraction

The first approach for content-driven campaign detection is what we refer to as *loose campaign extraction*. The main idea is to identify as a logical campaign all chains of messages that share common "talking points". In this way, the set of all loose campaigns is the set of all maximally connected components in the message graph:

Definition 2 (Loose Campaign)**.** *A loose campaign is a subgraph $s = (V', E')$, such that s is a maximally connected component of G, in which s is connected, and for all vertices m_{ix} such that $m_{ix} \in V$ and $m_{ix} \notin V'$ there is no vertex $m_{jy} \in V'$ for which $(m_{ix}, m_{jy}) \in E$.*

As an example, Figure 4.2 illustrates a collection of 10 messages, edges corresponding to messages that are highly correlated, and the two maximal components (corresponding to loose campaigns): {1, 2, 3, 6, 7, 8, 9} and {4, 5}. Such an approach to campaign detection faces a critical challenge, however: not all maximally connected components are necessarily campaigns themselves (due to long chains of tangentially-related messages). For example, a chain of similar messages A–B–C–...–Z, while displaying local similarity properties (e.g., between A and B and between Y and Z) will necessarily have low similarity across the chain (e.g., A and Z will be dissimilar since there is no edge between the pair, as in the case of messages 9 and 1 in Figure 4.2). In practice, such maximally connected components could contain disparate "talking points" and not strong campaign coherence.

4.3.2.2 Strict Campaign Extraction

A natural alternative is to constrain campaigns to be maximal cliques, what we call *strict campaigns*:

Definition 3 (Strict Campaign)**.** *A strict campaign* $s' = (V'', E'')$ *in a message graph* $G = (V, E)$*, in which* $V'' \subseteq V$ *and* $E'' \subseteq E$*, such that for every two vertices* m_{ix} *and* m_{jy} *in* V''*, there exists an edge* $(m_{ix}, m_{jy}) \in E''$ *and the clique cannot be enlarged by including one more adjacent vertex (corresponding to a message in* M*).*

To identify these strict campaigns, we can first identify all loose campaigns – by identifying all maximally connected components over the message graph, we can prune from consideration all singleton messages and are left with a set of candidate campaigns. Over these candidates, we can identify the strict campaigns through maximal clique mining. However, discovering all maximal cliques from a graph is an NP-hard problem (i.e., the time complexity is exponential). Finding all maximal cliques takes $O(3^{n/3})$ in the worst case where n is the number of vertices [133]. Over large graphs, even with parallelized implementation over MapReduce-style compute clusters, the running time is still $O(3^{n/3}/m)$ in the worst case, where n is the number of vertices and m is the number of reducers [154].

And there is still the problem that even with a greedy approximation, strict campaign detection may overconstrain the set of campaigns, especially in the case of loosely-connected campaigns. Returning to the example in Figure 4.2, the maximal cliques {1, 2, 3} and {2, 3, 6} would be identified as strict campaigns, but perhaps {1, 2, 3, 6, 7} form a coherent campaign even though the subgraph is not fully-connected. In this case the strict approach will identify multiple overlapping campaigns and will miss the larger and (possibly) more coherent campaign. In terms of our metrics, the expectation is that strict campaign detection will favor precision at the expense of recall.

4.3.2.3 Cohesive Campaign Extraction

Hence, we also consider a third approach which seeks to balance loose and strict campaign detection by focusing on what we refer to as *cohesive campaigns*, which relaxes the conditions of maximal cliques:

Definition 4 (Cohesive Campaign)**.** *Given a message graph G = (V,E), a subgraph G' is called a* ***cohesive campaign*** *if the number of edges of G' is close to the maximal number of edges with the same number of vertices of G'.*

The intuition is that a cohesive campaign will be a dense but not fully connected subgraph, allowing for some variation in the "talking points" that connect subcomponents of the overall campaign. There are a number of approaches mining dense subgraphs [55, 46, 144] and the exact solution is again NP-hard in computation complexity, so we adopt a greedy approximation approach following the intuition in [144]. The approach to extract cohesive campaigns requires a notion of maximum co-clique $CC(m_{ix}, m_{jy})$ for all neighbors:

Definition 5 (Maximum co-clique: $CC(m_{ix}, m_{jy})$)**.** *Given a message graph G = (V,E), the maximum co-clique $CC(m_{ix}, m_{jy})$ is the (estimated) size of the largest clique containing both vertices m_{ix} and m_{jy}, where $m_{jy} \in V$ and m_{jy} is a neighbor vertex of m_{ix} (i.e., they are connected).*

Considering all of a vertex's neighbors, we define the largest of the maximum co-cliques as $C(m_{ix})$:

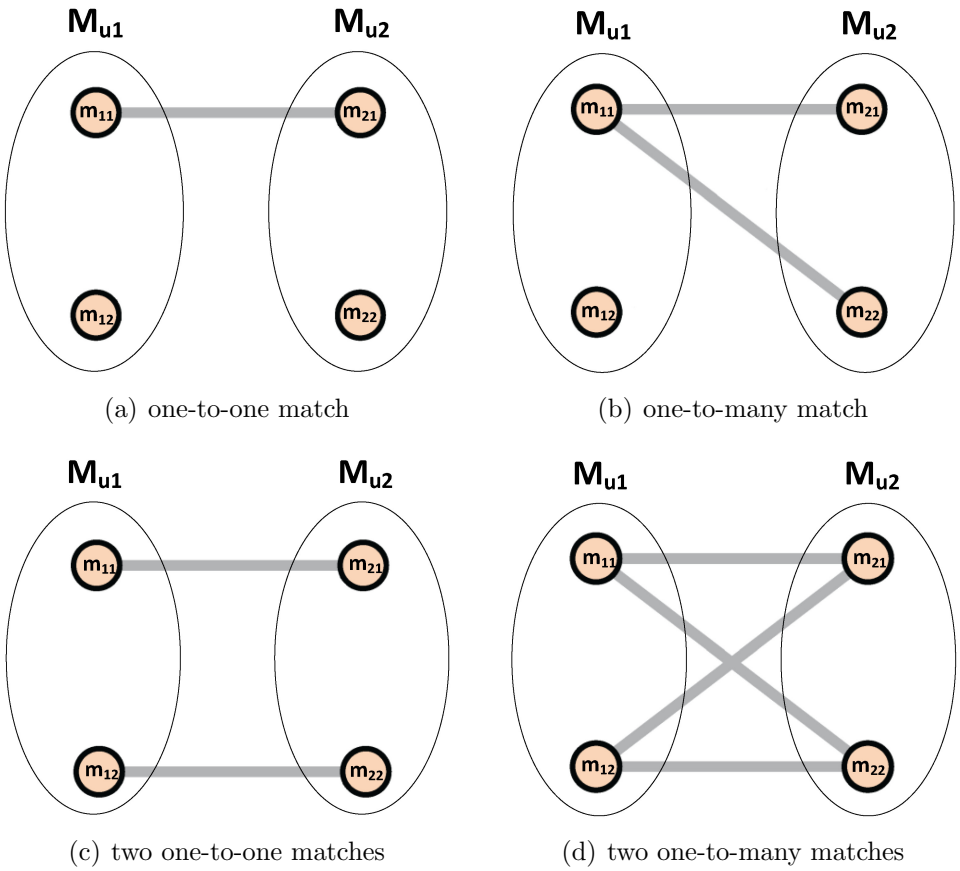
Definition 6 ($C(m_{ix})$)**.** *Then, $C(m_{ix})$ is the largest value between m_{ix} and any neighbor m_{jy}, formally defined as $C(m_{ix}) = max\{CC(m_{ix}, m_{jy}), \forall m_{jy} \in Neighbor(m_{ix})\}$.*

With these definitions in mind, our approach to extract cohesive campaign is as follows:

1. Estimate each vertex's $C(m_{ix})$: In the first step, our goal is to estimate the C values for every vertex in a candidate campaign which indicates the upper bound of the maximum clique size the vertex belongs to. Starting at a random vertex m_{ix} in s, we compute the maximum co-clique size $CC(m_{ix}, m_{jy})$, where $m_{jy} \in V'$ and m_{jy} is a neighbor vertex of m_{ix}. Then, we compute $C(m_{ix})$. We insert m_{jy} into a priority queue and sort all m_{jy} by $CC(m_{ix}, m_{jy})$. Next, we greedily advance to the m_{jy}, which has the largest $CC(m_{ix}, m_{jy})$ among all m_{jy}, and remove it from the queue. Finally, we compute $C(m_{jy})$. We repeat this procedure for every vertex in the candidate campaign. At the conclusion of this procedure, we have an estimated $C(m_{ix})$ for every vertex.

2. Cohesive campaign extraction: Given the estimated $C(m_{ix})$ for every vertex in a candidate campaign, by considering the order in which the greedy algorithm in Step 1 encounters each vertex, we can consider consecutive neighbors as potential members of the same coherent campaign. Intuitively, the $C(m_{ix})$ values should be high for vertices in dense subgraphs but should drop as the algorithm encounters nodes on the border of the dense subgraph, then rise again as the algorithm encounters vertices belonging to a new dense subgraph. We identify the first vertex with an increasing $C(m_{ix})$ over its neighbor as the initial boundary of a cohesive campaign. We next include all vertices between this first boundary up to and including the vertex with a $C(m_{ix})$ value larger than or equal to some threshold (= the local peak value * λ). By tuning λ to 1, the extracted cohesive campaigns will be nearly clique-like; lower values of λ will result in more relaxed campaigns (i.e., with less density). We repeat this procedure until we extract all cohesive subgraphs in the candidate campaign.

The output of the cohesive campaign extraction approach is a list of cohesive



(a) one-to-one match (b) one-to-many match

(c) two one-to-one matches (d) two one-to-many matches

Figure 4.3: Four matches of correlated messages between user u_1 and u_2

campaigns, each of which contains a list of vertices forming a cohesive subgraph.

4.3.3 User Level Campaign Detection

We turn our attention to a *user-aggregated* perspective. In the message level campaign detection in the previous subsection, we have viewed all messages without consideration for *who* is posting the messages. By also considering user-level information, we are interested to see how this impacts campaign detection. The intuition is that by aggregating the messages posted by a single user, we may discover cross-user correlations not captured at the individual message level (e.g., for two users posting a sequence of correlated messages), leading to more robust campaign detection.

Definition 7 (User-Aggregated Message Graph)**.** *A user-aggregated message graph is a graph* $G_u = (V, E)$ *where* V *is a collection of* n *users' aggregate messages* $V = \{M_{u_1}, M_{u_2}, ...M_{u_n}\}$*. An edge* $(M_{u_i}, M_{u_j}) \in E$ *exists for every pair of vertices* (M_{u_i}, M_{u_j}) *in* V *where* $confidence\ (Mu_i, Mu_j) > threshold$*, for some measure of confidence and threshold. In the confidence computation, message similarity for every pair of messages* (m_{ix}, m_{jy}) *is computed where* $corr(m_{ix}, m_{jy}) > \tau$*,* $m_{ix} \in M_{u_i}$*,* $m_{jy} \in M_{u_j}$ *and* $M_{u_i}, M_{u_j} \subseteq M$*, for some measure of correlation and some parameter* τ*.*

An important challenge is to define the correlation across vertices in the user-aggregated message graph, since each vertex now represents multiple messages (and so straightforward adoption of the message-level correlation approach is insufficient). For example the two users in Figure 4.3 could have several different degrees of message-level correlation, based on the overlap between their messages. In the figure, we show messages $M_{u_1} = \{m_{11}, m_{12}\}$, and $M_{u_2} = \{m_{21}, m_{22}\}$ from two users u_1 and u_2 respectively. An edge represents that a pair of two messages between M_{u_1} and M_{u_2} are correlated.

To compute user-based correlation, we propose a measure called *confidence* that aggregates message-message correlation and reflects (i) that one edge in a one-to-many match receives same weight comparing to the edge in a one-to-one edge; (ii) that extra edges in a one-to-many match receive less weight than the weight for the edge in a one-to-one match, but still credits the one-to-many match for more evidence of user-based correlation.

Concretely, we calculate confidence in the following way: Given two users u_1 and u_2 and their latest k messages $M_{u_i} = \{m_{i1}, m_{i2}, \ldots, m_{ik}\}$ where i is a user id (i.e., 1 or 2 in our example). First, we compute pairwise message correlation across M_{u_1}

and M_{u_2}, where pairs are $P = \{m_{1x}, m_{2y} | 1 \leq x, y \leq k\}$. If the correlation of a pair in P is larger than threshold τ, we consider the pair to be correlated. By continuing this procedure for each pair in P, we have correlated pairs P' and can calculate: (1) the number of pairs in P', $N = |\{m_{1x}, m_{2y} | corr(m_{1x}, m_{2y}) \geq \tau, 1 \leq x, y \leq k\}|$; and (2) the minimum n between number of distinct messages belonging to P' in M_{u_1} and number of distinct messages belonging to P' in M_{u_2}, where $n = MIN(|\{m_{1x}|$ $m_{1x} \in M_{u_1}$ and $M_{1x} \in P'\}|,|\{m_{2y}|$ $m_{2y} \in M_{u_2}$ and $m_2 j \in P'\}|)$. Now, we define that *confidence* as:

$$confidence = \alpha n + (1 - \alpha)(N - n)$$

where α is the weight for the only edge in a one-to-one match or one edge in a one-to-many match, and $1 - \alpha$ is the weight for each of the extra edges in a one-to-many match. We assigned 0.95 to α to balance between αn and $(1 - \alpha)(N - n)$. Returning to Figure 4.3(a),(b),(c),(d), we have {N=1, n=1, confidence=0.95}, {N=2, n=1, confidence=1}, {N=2, n=2, confidence=1.9}, and {N=4, n=2, confidence=2} showing that in order of user-based correlation a < b < c < d.

4.3.4 MapReduce-Based Implementation

To support scalable identification of correlated messages, we implement the proposed approach over the MapReduce framework, which was introduced by Google to process large datasets on a cluster of machines [32]. In MapReduce style programming, each task is divided into two sub-functions: (1) a mapper: a sequence of data is inserted to a computation to generate partial results; and (2) a reducer: the results are then aggregated. We implemented our correlated message identification approach on Hadoop [4] which can facilitate the handling of large scale social message data.

The implementation consists of three MapReduce jobs, illustrated in Figure 4.4

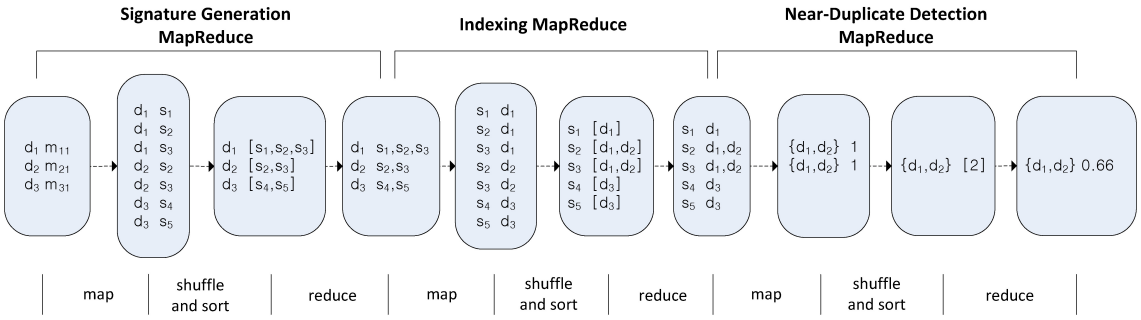


Figure 4.4: Logical data flow of the three MapReduce jobs for identifying correlated messages

with the following notation: (1) d_k is an auto-increasing message ID for a message; (2) m_{ij} indicates the j_{th} message from user u_i; (3) a near-duplicate detection algorithm generates three signatures (s_1, s_2, s_3) from the message m_{11}; (4) { } means a tuple and [ ] means a list. To calculate the correlation of the Jaccard coefficient (we use Jaccard coefficient in this example, but use Overlap coefficient in the experiments), we calculate each message's number of signatures in the map function of the signature generation job and pass the information associated to the message ID to later jobs. The near-duplicate detection returns pairs of near-duplicate messages (e.g., m_{11} and m_{21} have 0.66 similarity). To test the gains from a MapReduce-based implementation, we ran the message correlation component over 1.5 million Twitter messages as a MapReduce job on a small nine-node cluster and as a single-threaded (non MapReduce) job on a single machine. The MapReduce job took only 7 minutes as compared to one day in the non-MapReduce approach, indicating the gains from parallelization.

4.4 Experimental Study

In this section, we explore campaign discovery over social media through an application of the framework to messages and user-aggregated messages sampled from

Twitter. For message level campaign detection, we begin by examining how to accurately and efficiently construct the campaign message graph, which is the critical first step necessary for campaign detection. We find that a short-text modified Shingling-based approach results in the most accurate message graph construction. Based on this finding, we next explore campaign detection methods over the small hand-labeled Twitter dataset, before turning our sights to analysis of campaigns discovered over the large (1.5 million messages) Twitter dataset. Based on the insights learned from the experiments in message level campaign detection, we run user level campaign detection to see whether we can find more evidence of spam and other coordinated campaigns. In the end of this section, we analyze the temporal patterns of campaigns, which suggest the potential of predicting a campaign's category based on its temporal pattern.

4.4.1 Message Level

We begin by examining message graph construction, which is the critical first step necessary for campaign detection.

4.4.1.1 Message Graph Construction

Recall that each node in the message graph corresponds to a message; edges correspond to some reasonable notion of "relatedness" between messages corresponding to human-labeled similar "talking points". Our first goal is to answer the question: can we effectively determine if two messages are correlated (i.e., algorithmically determine if they share similar "talking points") across hundreds of millions of short messages for constructing the message graph in the first place? This step is critical for accurate message graph formation for discovering campaigns.

Using the small *campaign dataset* (CD_{Small}), we consider the 298 pairs of messages sharing similar "talking points" (as determined by human judges) as the ground

truth for whether an edge should appear in the message graph between the two messages. We can measure the effectiveness of a message correlation method by precision, recall, and F_1. Precision (P) is the fraction of predicted edges that are correct:

$$\frac{\text{\# of correctly predicted edges}}{\text{\# of predicted edges}}$$

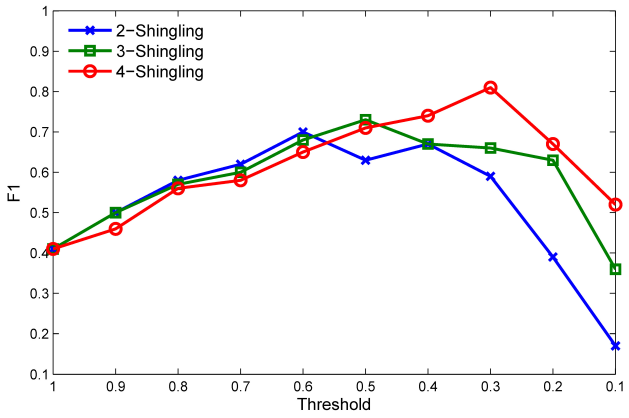
Recall (R) is the fraction of correct edges that are predicted:

$$\frac{\text{\# of correctly predicted edges}}{\text{\# of edges}}$$

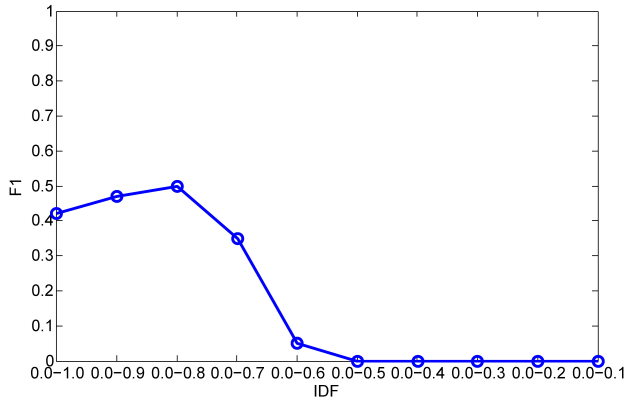
The F_1 measure balances precision with recall: $\frac{2PR}{P+R}$.

Identifying Correlated Messages: We investigate the identification of correlated messages through a comparative study of the six distinct techniques described in Section 4.3: unigram-based overlap between messages, edit distance, Euclidean distance and three representative near-duplicate detection algorithms (Shingling [15], I-Match [25], SpotSigs [130]). The near-duplicate detection approaches such as Shingling, I-Match and SpotSigs have shown great promise and effectiveness by web search engines to efficiently identify duplicate web content, but their application to inherently short messages lacking context is unclear.

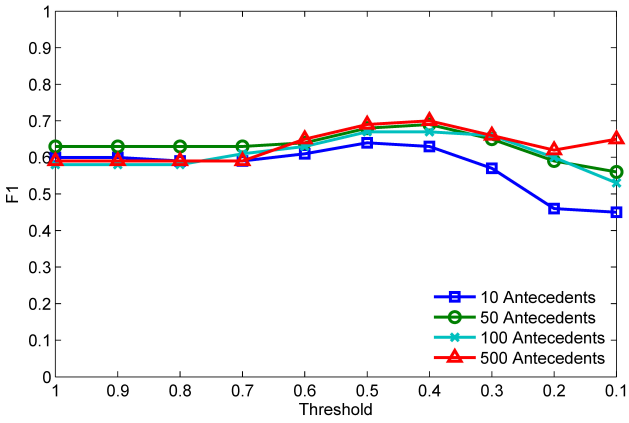
To evaluate each approach, we considered a wide range of parameter settings. For example, the quality of Shingling depends on the size of the shingle (2, 3, 4). I-Match requires minimum and maximum IDF values; we varied the min and max IDF values over the range [0.0, 1.0] in 0.1 increments and considered all possible pairs (e.g., min = 0.1, max = 0.6). SpotSigs requires a number of antecedents (which we varied across 10, 50, 100, and 500) and a specification of what antecedents will be used. As the authors of SpotSigs [130] did in their experiments, we used stopwords as antecedents. And across all approaches, we must also set a predefined threshold



(a) Shingling



(b) I-Match



(c) SpotSigs

Figure 4.5: Performance of Shingling, I-Match and SpotSigs with different parameter value

value τ, above which a pair of messages are considered correlated (and hence and edge should appear in the message graph).

With this large parameter space in mind, we show in Table 4.1 the results across all approaches that optimizes the F_1 score (The details of performance of Shingling, I-Match and SpotSigs with different parameter value are shown in Figure 4.5).

Table 4.1: Identifying correlated messages

Approach	**F_1**	**Precision**	**Recall**
Unigram ($\tau = 0.8$)	0.63	0.97	0.46
Edit Distance ($\tau = 11$)	0.54	0.97	0.38
Euclidean Distance ($\tau = 5$)	0.61	0.99	0.44
4-Shingling ($\tau = 0.3$)	**0.81**	0.89	0.73
I-Match (IDF=[0.0, 0.8])	0.50	0.53	0.47
SpotSigs (#A=500, $\tau = 0.4$)	0.70	0.77	0.64

We see that the baseline Shingling approach performs the best, with an $F_1 = 0.81$. In contrast, both I-Match and SpotSigs performed much worse (0.50, 0.70), in sharp contrast to their performance in near-duplicate detection of web pages (with F_1 near 95%) [130, 161]. While these approaches work well in news articles and web pages (relatively long text), they do not work well for short text. We also observe that unigram, edit distance and Euclidean distance based methods perform poorly, primarily due to their low recall. This indicates that short messages that do share common "talking points" may be missed by these approaches which emphasize on only minor syntactic changes across messages.

Refining Shingling: Based on these results, we further explore refinements to the baseline Shingling approach. First, we vary the base tokenization unit for message comparison, which is especially critical for short messages. We consider three general approaches for extracting tokens to generate shingles: (i) word-based k-grams,

in which k consecutive words are treated as base tokens; (ii) character-based k-grams, in which k consecutive characters are treated as base tokens. As compared to word-based k-grams, character-based k-grams generate more tokens but offer finer granularity of measuring message correlation; and (iii) orthogonal sparse bigrams, introduced by Cormack et al. [27] for lexically expanding a short message by generating sparse bigrams by the number of intervening words, each of which we denote by "?". For example, "lady gaga is unique person" generates sparse bigrams: lady + gaga, lady + ? + is, lady + ? + ? + unique, gaga + is, gaga + ? + unique, gaga + ? + ? + person, is + unique, is + ? + person, unique + person.

Finally, we note that straightforward application of the Jaccard coefficient over short messages may underestimate the degree of overlap between two messages, resulting in the mislabeling of correlated messages as unrelated. For example, suppose we apply 4-shingling to the following two messages, splitting each message on whitespace and punctation:

- Here's How Apple's iPad Is Invading The Business World (AAPL, RIMM, MSFT) - San Francisco Chronicle: `http://bit.ly/dhqDGf`
- Here's How Apple's iPad Is Invading The Business World (AAPL, RIMM, MSFT) `http://bit.ly/d3ClTj`

With 18 and 15 shingles, respectively, and 11 shingles in common, the Jaccard coefficient will identify a correlation of only 0.5 (11 / (18+15-11)), even though the two messages are nearly identical. With a typical threshold τ of 0.6 or above, these two messages, though clearly correlated would not be properly identified. Hence, we propose as a measure of correlation the overlap coefficient:

$$corr_{overlap}(A, B) = \frac{|A \cap B|}{min(|A|, |B|)}$$

which in this case results in a correlation value of $11/15 = 0.73$. In general, smaller number of words in two messages will give us higher Jaccard and overlap coefficients diverge. Experimentally, we evaluate the impact of these approaches on the quality of correlated message identification.

Table 4.2: Refinements to shingling

Approach	$\mathbf{F}_1$	**Precision**	**Recall**
4-Shingling ($\tau = 0.3$)	0.81	0.89	0.73
Character k-grams ($k = 6$, $\tau = 0.6$)	0.74	1	0.59
OSB. ($\tau = 0.5$)	0.68	0.6	0.79
With Short Message Overlap	**0.88**	0.92	0.83

Interestingly, as seen in Table 4.2, neither character-based k-grams nor orthogonal sparse bigrams, which have shown promise in other short text domains, performed as well as shingling or the short-message optimized approach presented in this chapter. We conjecture that word-based tokens can capture similar messages well compared to character k-grams and orthogonal sparse bigrams which may generate too many features, leading to message correlation confusion. The short message overlap optimization, however, results in the best results and so we shall use this as a core approach for generating the message graphs in all subsequent experiments.

4.4.1.2 Campaign Detection over Small Data

In the previous set of experiments, we evaluated several approaches to measuring message correlation. Now we turn our attention to evaluating campaign detection methods. We begin in this section with the small dataset (which recall allows us to measure precision and recall against ground truth) before considering the large dataset.

Over the hand-labeled campaigns in CD_{Small}, we apply the three graph-based campaign extraction methods: (i) loose; (ii) strict; and (iii) cohesive, over the message graph generated via the best performing message correlation method identified in the previous section. We also compare campaign extraction using a fourth approach based on text clustering. For this non-graph-based approach, we consider k-means clustering, where each message is treated as vector with 10K bag-of-words features, weighted using TF-IDF, with Euclidean distance as a distance function. We vary the choice of k value, and report the best result.

Table 4.3: Effectiveness comparison of campaign detection approaches

Approach	**NumC**	**F_1**	**Precision**	**Recall**
Loose	12	0.962	0.986	0.940
Strict	12	0.906	0.907	0.904
Cohesive	11	**0.963**	0.977	0.950
k-means	5	0.89	1	0.805

Table 4.3 presents the experimental results of the four campaign detection approaches. The cohesive campaign detection approach found 11 campaigns ($NumC$) like the ground truth, but missed a message in two campaigns. The strict approach found 12 campaigns, missed one message in a true campaign, and divided a true campaign to two predicted campaigns because the approach due to the strict campaign rule (all nodes in a campaign should be completely connected). The loose approach found 12 campaigns, one of which is not an actual campaign (false positive) and some predicted campaigns contain dissimilar messages due to long chains. The k-means clustering algorithm found only 5 campaigns. Overall, the cohesive and strict approaches outperformed the loose and cluster-based approaches. In practice, the ideal approach should return the same number of campaigns as the ground truth and

do so quickly. In this perspective, the cohesive approach would be preferred over the strict approach because the number of its campaigns is the same with the ground truth, and it is relatively faster than the strict approach.

4.4.1.3 Campaign Detection over Large Data

We next examine campaign extraction from the large Twitter dataset, CD_{Large}. Can we detect coordinated campaigns in a large message graph with 1.5 million messages? What kind of campaigns can we find? Which graph technique is the most effective to find campaigns?

Message Graph Setup: Based on the best message graph construction approach identified in the previous section, we generated a message graph consisting of 1.5 million vertices (one vertex per message). Of these, 1.3 million vertices are singletons, representing messages without any correlated messages in the sample (and hence, not part of any campaign). Based on this sample, we find 199,057 vertices have at least one edge; in total, there are 1,027,015 edges in the message graph.

Identifying Loose Campaigns: Based on the message graph, we identify as *loose campaigns* all of the maximally connected components, which takes about 1 minute on a single machine (relying on a breadth-first search with time complexity $O(|E| + |V|)$). Figure 4.6 shows the distribution of the size of the candidate campaigns on a log-log scale. We see that the candidate campaign sizes approximately follows a power law, with most candidates consisting of 10 or fewer messages. A few candidates have more than 100 messages, and the largest candidate consists of 61,691 messages. On closer inspection, the largest candidate (as illustrated in Figure 4.7) is clearly composed of many locally dense subgraphs and long chains. Examining the messages in this large candidate, we find many disparate topics (e.g., spam messages, Justin Bieber retweets, quotes, Facebook photo template) and no strong candidate-wide

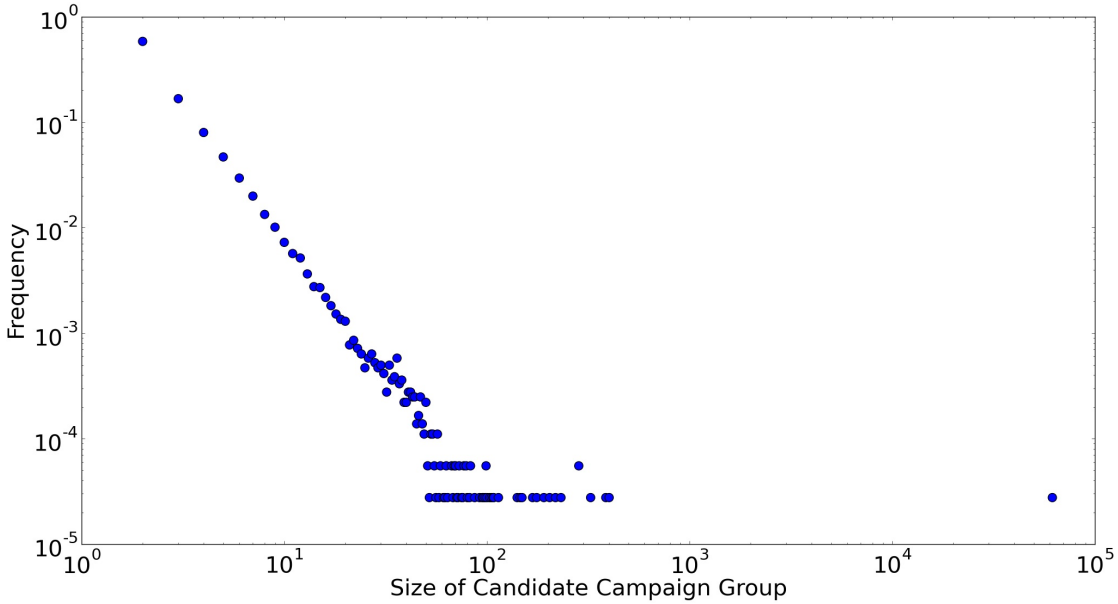


Figure 4.6: This figure depicts the distribution of the size of candidate campaigns on a log-log scale. It follows a power law.

theme, as we would expect in a coherent campaign.

Identifying Strict Campaigns: To refine these candidates, one approach suggested in Section 4.3 is *strict campaign detection*, in which we consider only maximal cliques as campaigns (in which all message nodes in a subgraph are connected to each other). While maximal clique detection may require exponential time and not be generalizable to all social message datasets, in this case we illustrate the maximal cliques found even though it required ∼7 days of computation time (which may be unacceptable for campaign detection in deployed systems). Considering the top-10 strict campaigns discovered in order of size: [559, 400, 400, 228, 228, 227, 227, 217, 217, 214], we find high overlap in the campaigns discovered. For example, the 2nd and 3rd strict campaigns (each of size 400) have 399 nodes in common. Similarly, the 4th, 5th, 6th, 7th, and 10th strict campaigns have over 200 nodes in common,

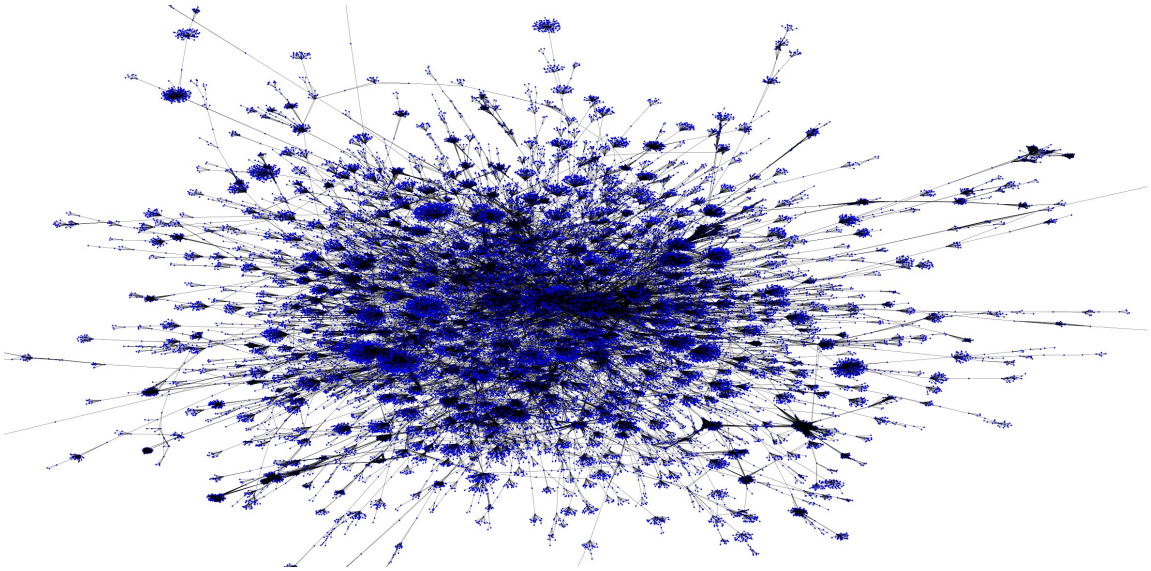


Figure 4.7: This figure depicts a candidate with 61,691 vertices. A blue dot and a black line represent a vertex and an edge, respectively. The area in the center is dark because most vertices in the center are very densely connected.

suggesting that these five different strict campaigns in essence belong to a single coherent campaign (see Figure 4.8). This identification of multiple overlapping strict campaigns – due to noise, slight changes in message "talking points", or other artifacts of short messages – as well as the high cost of maximal clique detection suggests the cohesive campaign detection approach may be preferable.

Identifying Cohesive Campaigns: We next applied the *cohesive campaign extraction* approach to the set of candidate campaigns corresponding to maximal connected components. We assign λ to 0.95 and use the CSV tool [144] for an efficient implementation of computing each vertex m_{ix}'s $C(m_{ix})$ by mapping edges and vertices to a multidimensional space. Although computing $C(m_{ix})$ of all vertices takes $\mathrm{O}(|V|^2 \log |V| 2^d)$ where d is a mapping dimension, the performance for real datasets is typically sub-quadratic. Figure 4.9 shows the distribution of the size of the cohesive campaigns in a log-log scale. Like the candidate campaign sizes, we see that the

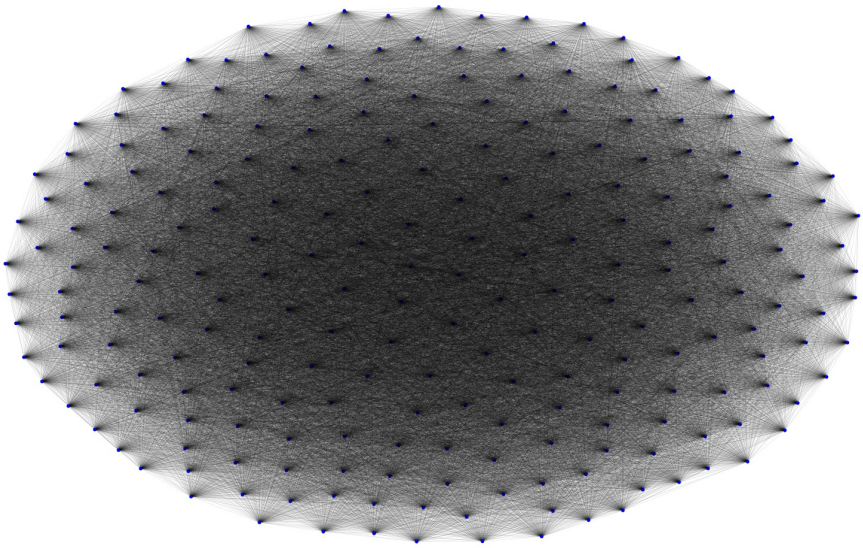


Figure 4.8: An example dense subgraph campaign: the center area is dark because vertices in the area are very densely connected; this subgraph is almost fully connected except a few vertices. While strict campaign detection identifies 5 different maximal cliques, cohesive campaign detection identifies a single coherent campaign including all vertices.

cohesive campaigns follow a power law. Since the cohesive campaign extraction approach can isolate dense subgraphs, we see that the large 61,691 message candidate has been broken into 609 sub-components. Compared to strict campaign detection, the cohesive campaign extraction approach required only 1/7 the computing time on single workstation.

Examining the top-10 campaigns (shown in Table 4.4) we see that the cohesive campaign detection approach overcomes the limitations of strict campaign detection by combining multiple related cliques into a single campaign (recall Figure 4.8). The biggest campaign contains 560 vertices and is a spam campaign. The "talking point" of this campaign is an Iron Man 2 promotion of the form: "#Monthly Iron Man 2 (Three-Disc Blu-ray/DVD Combo + Digital Copy) ... http://bit.ly/9L0aZU",

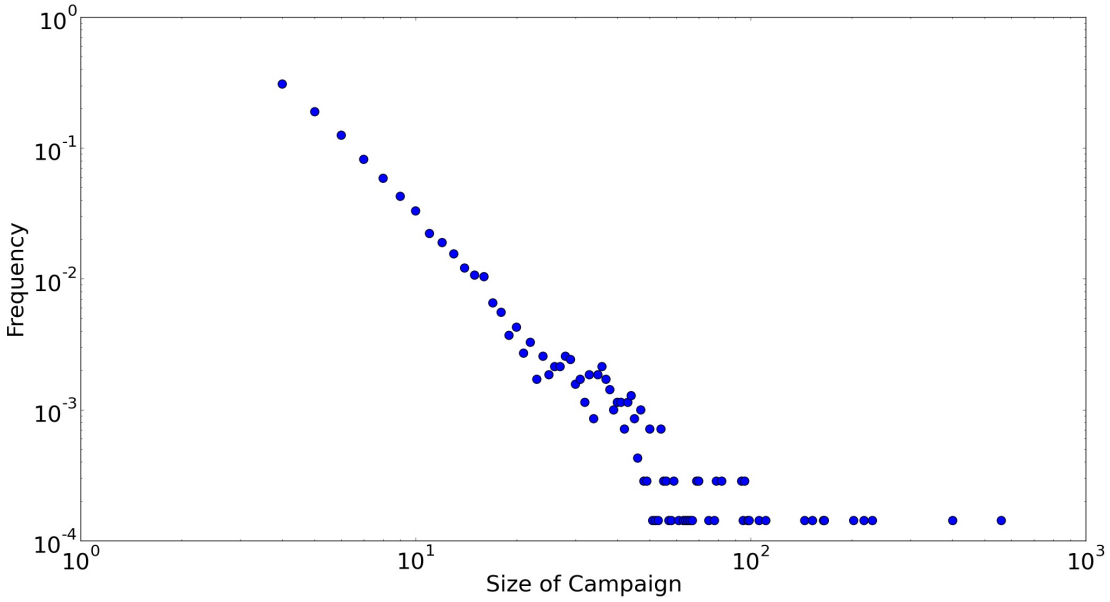


Figure 4.9: This figure depicts the distribution of the size of cohesive campaigns on a log-log scale. It also follows a power law.

though individual messages vary the exact wording and inserted link.

Based on a manual inspection of the identified campaigns, we categorize the campaigns into five categories:

- *Spam Campaigns*: These campaigns typically post duplicate spam messages (changing @username with the same payload), or embed trending keywords; often with a URL linking to a malware website, phishing site or a product website. Example: "Want FREE VIP, 100 new followers instantly and 1,000 new followers next week? GO TO `http://alturl.com/bpby`".

- *Promotion Campaigns*: Users in these campaigns promote a website or product. Their intention is to expose it to other people. Example: "FREE SignUp!!! earn $450 Per Month Do NOTHING But Getting FREE Offers In The Mail!! `http://budurl.com/PPLSTNG`".

Table 4.4: Top-10 largest campaigns

Msgs	Users	Talking Points
560	34	Iron Man 2 spam
401	390	Facebook photo template
231	231	Support Breast Cancer Research (short link)
218	218	Formspring template
203	197	Chat template (w/ link)
166	166	Support Breast Cancer Research (full link)
165	154	Quote "send to anyone u don't regret meeting"
153	153	Justin Bieber Retweets
145	31	Twilight Movie spam
111	111	Quote "This October has 5 Fridays ..."

- *Template Campaigns*: These are automatically-generated messages typically posted by a third-party service. Example: "I'm having fun with @formspring. Create an account and follow me at `http://formspring.me/xnadjeaaa`".

- *News Campaigns*: Participants post recent headlines along with a URL. Example: "BBC News UK: Rwanda admitted to Commonwealth: Rwanda becomes the 54th member of the Commonwealth g.. `http://ad.vu/nujv`".

- *Celebrity Campaigns*: Users in these campaigns send messages to a celebrity or retweet a celebrity's tweet. Example: "@justinbieber please follow me i love youuu<3".

Some of these campaigns are organic and the natural outgrowth of social behavior, e.g., a group of Justin Bieber fans retweeting a message, or a group posting news articles of interest. On closer inspection, we observe that many of the less organic campaigns (e.g., spam and promotion campaigns) are driven by a higher ratio of messages to participants. For example in Table 4.4, the Iron Man 2 spam campaign consists of 560 messages posted by only 34 different participants. In contrast, the

Justin Bieber retweet campaign consists of 153 messages posted by 153 different participants.

4.4.2 User Level

Based on this observation – of a handful of accounts aggressively promoting particular "talking points" in Twitter – we next turn to user-aggregated campaign detection. By collapsing multiple messages from a single user in the user-aggregated message graph, do we find more evidence of spam and other coordinated campaigns (since edges correspond to users with highly-correlated messages)? What impact does the confidence threshold have on campaign detection?

4.4.2.1 Data and Setup

Since the dataset for the previous study was based on a random sample of Twitter (meaning most users were represented by only one message), we use a user-focused dataset CD_{User} from Twitter consisting of 90,046 user profiles with at least 20 English-language messages. Based on these messages, we constructed a user-aggregated message graph where each vertex corresponds to a user and an edge exists between all users passing a threshold confidence value. For a threshold of 3.8 (i.e., $n = 4$) we find 2,301 vertices with at least one edge, and a total of 89,294 edges in the user-aggregated message graph.

4.4.2.2 Campaign Detection

Following the campaign framework in Section 4.3.2.3, we find 303 candidate campaigns illustrated in Figure 4.10. Applying the cohesive campaign extraction approach we find 62 campaigns with at least four users. Through manual inspection, we labeled each of the 62 campaigns according to campaign type (see Figure 4.11). We observe that spam and template campaigns are major campaign types in the all

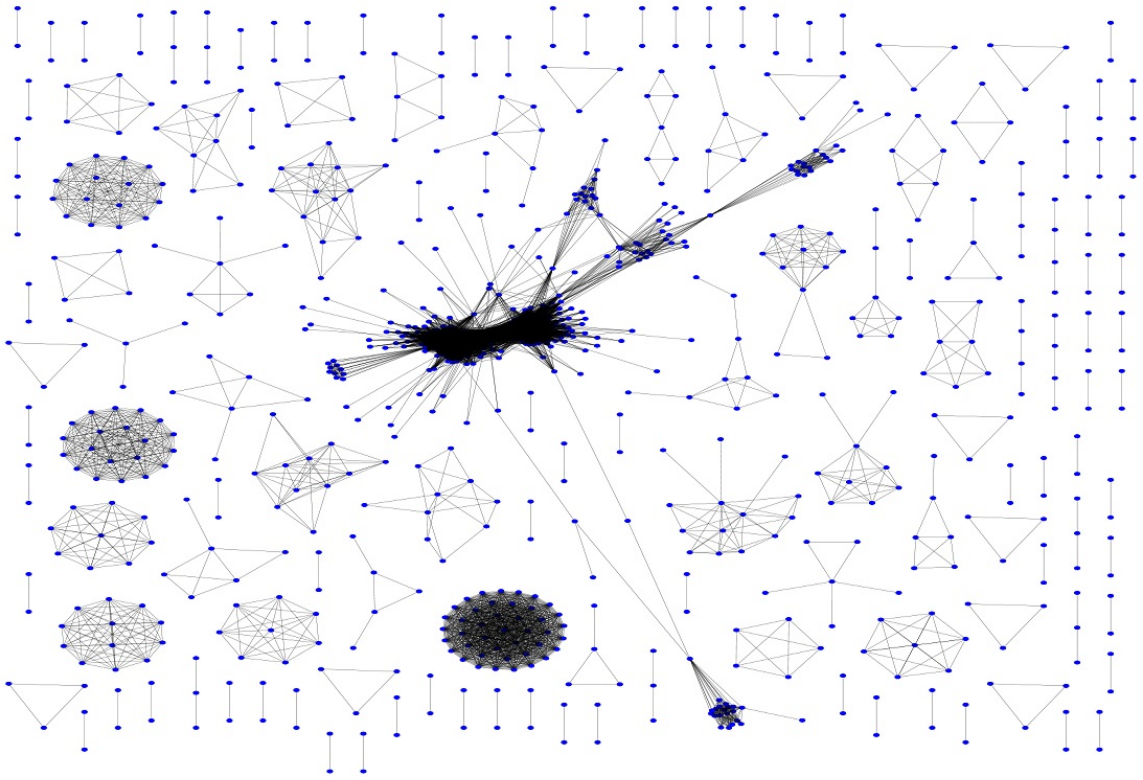


Figure 4.10: 303 candidate campaigns in the user-aggregated message graph

three partitions divide by ranges of the size.

We next analyze whether different campaign category has significantly different content/terms in messages. To identify significant terms for the users in each category type, we identify terms with high mutual information for each campaign category. Mutual information is a standard information theoretic measure of "informativeness" and, in our case, can be used to measure the contribution of a particular term to a category of campaign. Concretely, we build a unigram language model for each category of campaign by aggregating all messages by all users belonging to a particular campaign category (e.g, all users participated in spam campaign). Hence, mutual information is measured as: $MI(t,c) = p(t|c)p(c)log\frac{p(t|c)}{p(t)}$ where $p(t|c)$ is the probability that a user which belongs to category c has posted a message containing

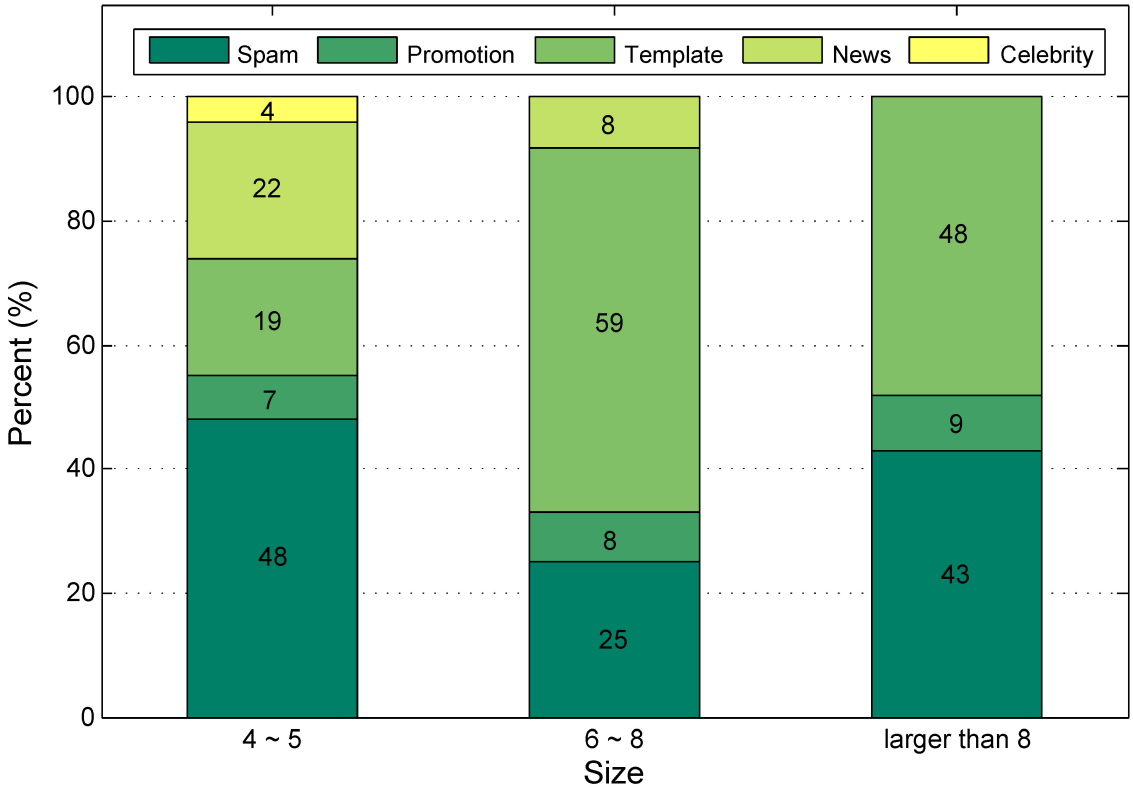


Figure 4.11: Campaign type distribution (threshold = 3.8)

term t, $p(c)$ is the probability that a user belongs to category c, and $p(t)$ is the probability of term t over all categories. That is, $p(t) = count(t)/n$. Similarly, $p(t|c)$ and $p(c)$ can be simplified as $p(t|c) = count(c,t)/count(c)$ and $p(c) = count(c)/n$ respectively, where $count(c,t)$ denotes the number of users in category c which also contain term t, and $count(c)$ denotes the number of users in category c.

Table 4.5 shows the top-10 significant terms for each campaign category. In spam campaigns, we observe that spammers have posted messages regarding increasing followers via a software service. An example message is "Hey Get 100 followers a day using http://yumurl.com/p74ZY6. Its super fast!". Note that the Twitter Safety team considers promoting such automated friend software as spam [139]. Promo-

Table 4.5: Top-10 significant terms for each campaign category

Category	Top 10 Terms				
spam	followers	100	day	site	fast
	check	twitter	account	upload	twtmuzik
promotion	iq	broadcasting	stickam	stream	quiz
	michael	#140kingofpop	jackson	free	woot
news	media	social	bbc	engadget	windows
	#news	apple	android	africa	iphone
template	video	#epicpetwars	xbox	chat	#tinychat
	joined	people	playing	youtube	live
celebrity	@justinbieber	follow	justin	bieber	love
	mee	song	plss	hiii	dream

tion campaigns promote particular links or products. An example message is "if you like iq quize's then check out this free iq quiz http://tiny.cc/amazingfreeiqquiz #donttrytoholla". Messages of the users in the news campaign contain hot keywords (e.g., social, media, android and iphone) or media name (e.g., bbc, engadget). The significant terms in template campaigns describe a user's status (playing, xbox) or reflect a service being used (chat, #tinycat and live). Users participating in celebrity campaigns often post messages targeting a particular celebrity (e.g., @justinbieber) expressing love or asking for the celebrity to reciprocate and follow the user.

Table 4.6: Campaign categories for low confidence threshold and high confidence threshold

Category	**Low**	**High**
Spam	42%	65%
Promotion	8%	3%
Template	37%	29%
News	11%	3%
Celebrity	2%	0%

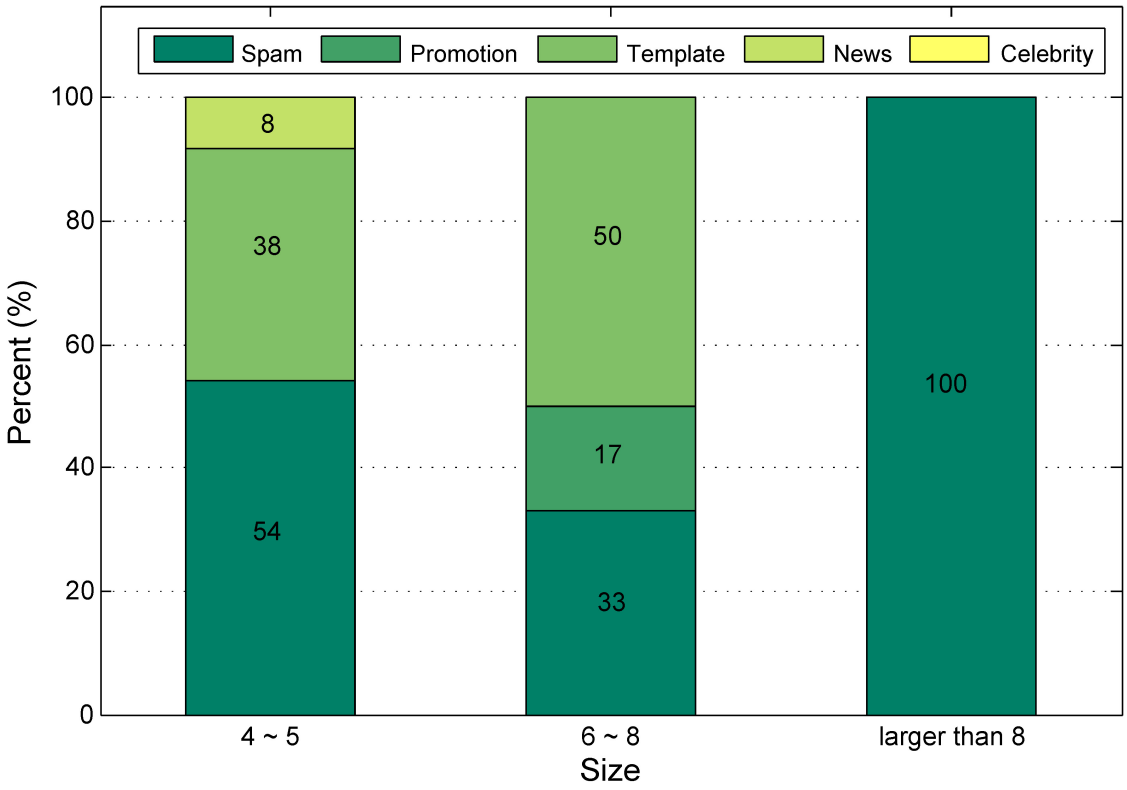


Figure 4.12: Campaign type distribution (threshold = 9.5)

4.4.2.3 Varying the Confidence Threshold

Now, we are interested in how the confidence threshold influences the campaigns detected. A higher confidence corresponds to more tightly-correlated users (pairs who tend to post a sequence of similar messages), and would perhaps suggest a strategic rather than organic campaign. When we increase the confidence threshold to 9.5 (i.e., $n = 10$) we find 28 campaigns as shown in Figure 4.12. Compared to the lower confidence threshold, the proportion of spam campaigns increases to 65% compared with 42% in the previous experiment (see Table 4.6). Second, we see that for campaigns of the largest size, all are spam campaigns. This indicates that the confidence threshold can be an effective tunable knob for identifying strate-

gic campaigns in large-scale social media. Overall, these user-aggregated message graph results show that content-based campaign detection can effectively identify campaigns of multiple types at low confidence and specifically of spam campaigns at high confidence.

4.4.3 Temporal Analysis

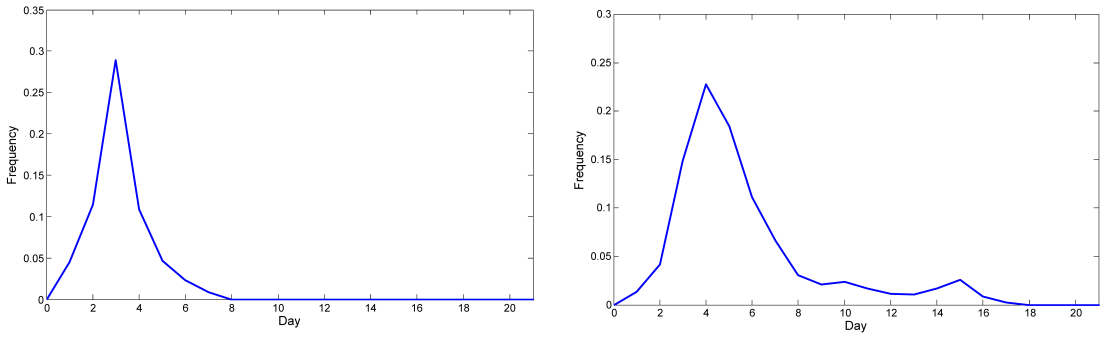
We next analyze temporal behaviors of the cohesive campaigns. Especially, we study each cohesive campaign category's temporal behaviors to see whether each campaign category has different temporal behavior. Using CD_{Large} (one week data) may be not enough to study temporal patterns because of sparse data. In order to overcome this sparsity, we extend the one week data to three weeks data collected between October 1 and October 21, 2010 (again, we used Twitter Streaming API which allows us to collect randomly 1% of all messages. If we can access all messages generated on Twitter, we may just need to use 1 week data or even shorter data for the temporal analysis.)

For temporal analysis, we selected the top-50 cohesive campaigns detected in Section 4.4.1.3, and added similar messages in the extended dataset into each campaign[†]. Then, we manually labeled the top-50 cohesive campaigns to one of four categories: spam, promotion, celebrity and template. The campaign category distribution is shown in Table 4.7.

Table 4.7: Categories of Top-50 cohesive campaigns

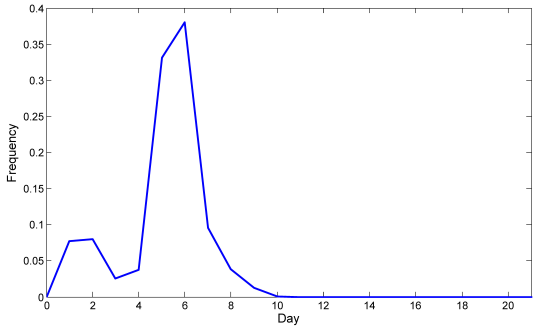
Category	**Percent**	**Category**	**Percent**
Spam	26%	Promotion	6%
Celebrity	34%	Template	34%

[†]There was no news campaign in the top-50 cohesive campaigns.

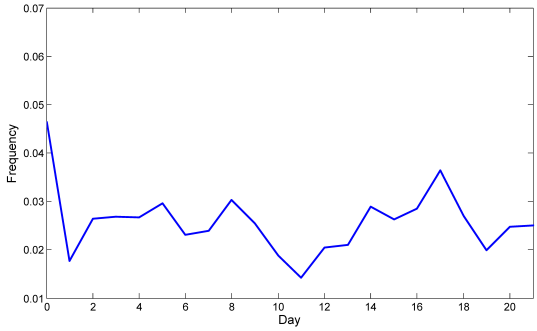


(a) Spam

(b) Promotion



(c) Celebrity



(d) Template

Figure 4.13: Average temporal graphs of four campaign categories

For temporal behavior analysis, a cohesive campaign is represented by a time series vector $T_a = (T_{a1}, T_{a2}, \ldots, T_{an})$. Each value in the vector denotes a number of messages belonging to the campaign in a time unit (e.g., 1 day). Likewise, we create 50 time series (campaign vectors) based on 1 day unit. To make a time series graph smooth (less fluctuated), we use two days moving average. For example, given a time series $T_a = (T_{a1}, T_{a2}, T_{a3}, \ldots, T_{an})$, two days moving average of T_a is $T'_a = (\frac{T_{a1}+T_{a2}}{2}, \frac{T_{a2}+T_{a3}}{2}, \ldots, \frac{T_{an-1}+T_{an}}{2})$.

We use dynamic time warping barycenter averaging (DBA) which is a global technique for averaging a set of sequences [103]. Compared to approaches like balanced hierarchical averaging or sequential hierarchical averaging, DBA avoids some of the

deficiencies of these alternative [98].

Figure 4.13 presents an average time series of each campaign type calculated by DBA. Spam campaigns have a sharp spike, reflecting how spammers post many similar messages at the beginning and then reduce the frequency of messages or change payload to avoid being caught by Twitter administrators. Users in promotion campaigns post messages over a longer period, suggesting that promotion and spam campaigns (though closely related) may reveal distinctions in their temporal patterns to support automatic differentiation. Celebrity campaigns have two spikes and then the frequency drops off. We conjecture that this phenomenon happens as people quickly retweet a celebrity's message (the first spike) and then the retweet passes through those user's social networks and is echoed (the second spike). Template campaigns have different temporal patterns from the others. As we can expect a temporal pattern of template campaigns, messages forming template campaign are posted constantly and statically over time. This phenomenon makes sense because these messages are posted by third party services or tools. Overall, each type of campaigns has different temporal pattern. This temporal analysis reveals the possibility to automatically classify a campaign type by its temporal pattern.

4.4.4 Summary

Through the above experiments, we found that it is possible to detect content-based campaigns in message and user levels in social media. Also, we found five campaign categories–spam, promotion, template, celebrity and news campaigns. The proposed cohesive campaign detection approach outperformed loose and strict campaign detection approaches and k-means clustering approach in terms of effectiveness and efficiency. The most encouraging results are the messages posted by users who participate in negative campaigns (spam and promotion campaigns) have higher con-

tent similarity. Temporal analysis of campaigns reveals that each campaign type has different temporal pattern, showing us the possibility to automatically determine a campaign's category.

4.5 Summary

In this chapter, we have investigated the problem of campaign detection in social media. We have proposed and evaluated an efficient content-driven graph-based framework for identifying and extracting campaigns from the massive scale of real-time social systems. We have found six campaign types (spam, promotion, template, celebrity, news and babble), and analyzed temporal behaviors of various campaign types.

5. COMBATING THREATS TO COLLECTIVE ATTENTION*

5.1 Introduction

We now turn to the third and final threat to social systems investigated in this dissertation. This new threat – which targets collective attention – is an emerging threat made possible by large-scale systems connecting millions of people.

Collective attention – exemplified by breaking news, viral videos, and popular memes that captivate the attention of huge numbers of users – is one of the cornerstones of large-scale social systems. As Wu and Huberman have noted, *collective attention* describes how "attention to novel items propagates and eventually fades among large populations" [155]. In the context of social media, an item – be it a video, web page, image – attracts the interest of a small group, then gathers a larger following as additional attention focuses on it, then (in some cases) exploding across social media to large-scale attention, and then finally fading in interest. Popular examples include YouTube videos that accumulate millions of views in a few days, memes attracting huge audiences on Reddit (http://www.reddit.com) and 4chan (http://www.4chan.org), spikes in search volume on Google and Twitter following breaking news, and so forth. As a result, many researchers have begun examining these phenomena, to model their dynamics, lifecycles, and future spread, e.g., [47, 54, 78, 79, 118].

Guided by the knowledge that collective user interest may quickly coalesce, malicious users have begun threatening the quality of information associated with this collective attention. As illustration, consider these three recent examples of *collective*

*Reprinted with permission from "Combating Threats to Collective Attention in Social Media: An Evaluation" by Kyumin Lee, Krishna Kamath, and James Caverlee, 2013. *Proceedings of the 7th International AAAI Conference on Weblogs and Social Media*, Copyright 2013 by AAAI.

Figure 5.1: Example YouTube video designed to capitalize on collective interest during and immediately after the London Olympics Opening Ceremony

attention spam found in large-scale social systems:

- *YouTube:* In the immediate aftermath of the London Olympics Opening Ceremony on July 27, 2012, we found that four of the top-five videos returned for the YouTube query "london olympics opening ceremony 2012" were videos tagged with keywords associated with the London Olympics Opening Ceremony, but that were expressly designed to promote an unrelated spammer-controlled website. Figure 5.1 shows one example, which includes a URL linking to a spam website.

- *Twitter:* Twitter publishes the current most-trending topics, and so spammers have been observed abusing this signal of collective user interest by "trend-

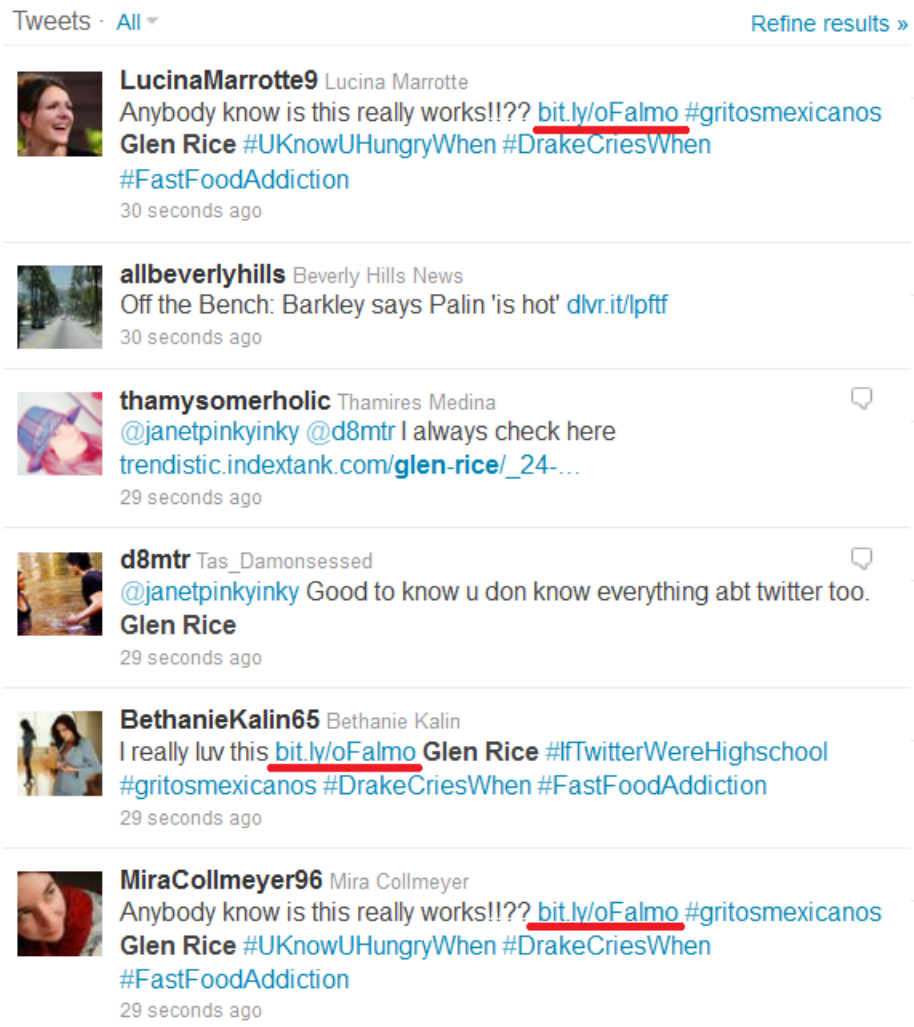


Figure 5.2: Spam messages targeting the Twitter trending topic "Glen Rice"

stuffing" these popular topics with spam messages including malicious URLs [59]. Figure 5.2 shows a sample search result for the trending topic "Glen Rice" for which three out of the most recently posted six messages are spam. All three spam messages include the same URL and multiple trending topics, but are posted from multiple accounts, adding to the growing evidence (e.g., [115, 132]) that spammers strategically post to Twitter in an organic-like way to simulate the behavior of non-spam users.

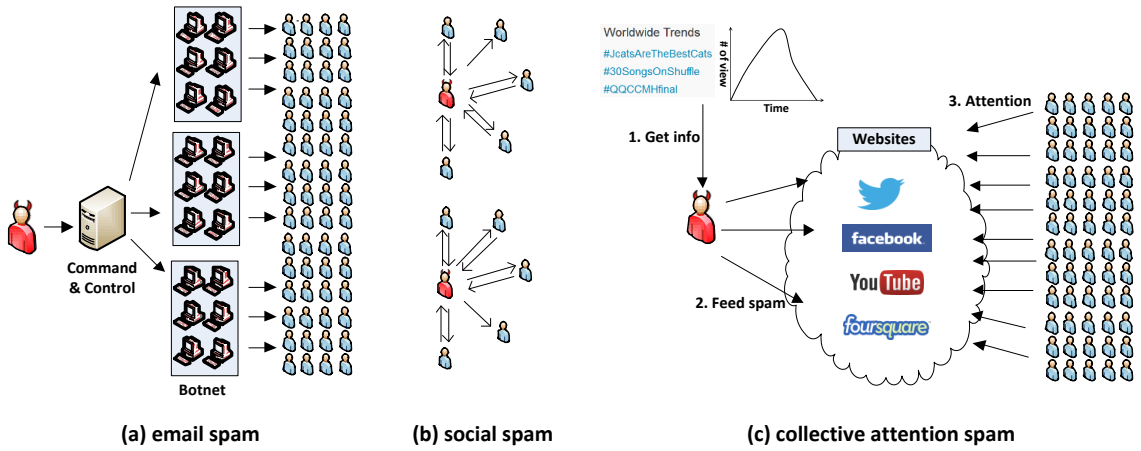


Figure 5.3: Three spam approaches. Collective attention spam relies on the users themselves to seek out the content where the spam will be encountered.

- *Facebook:* We have found many examples of popular profiles on Facebook – including celebrities, popular musical groups, and brand-name products – that have attracted spam photos and other spam content. Spammers can easily target these profile pages when interest is focused on these high-interest profiles, since many popular profile pages support media uploads by fans, which can be abused by collective attention spammers.

In contrast to traditional email spam and social spam, *collective attention spam relies on users themselves to seek out the content where the spam will be encountered.* In email spam, as illustrated in Figure 5.3(a), the spammer relies on a bulk attack based on the hope that a small percentage of users who are contacted will actually click on a link in an email. Social spam, as illustrated in Figure 5.3(b), is typically a more targeted attack than email spam, and relies on some social mechanism for coupling a spammer with an intended target (e.g., becoming friends in a social network, following a user on Twitter). In contrast, collective attention spam in Figure 5.3(c) targets users *who are already inherently interested in the topic.* In this

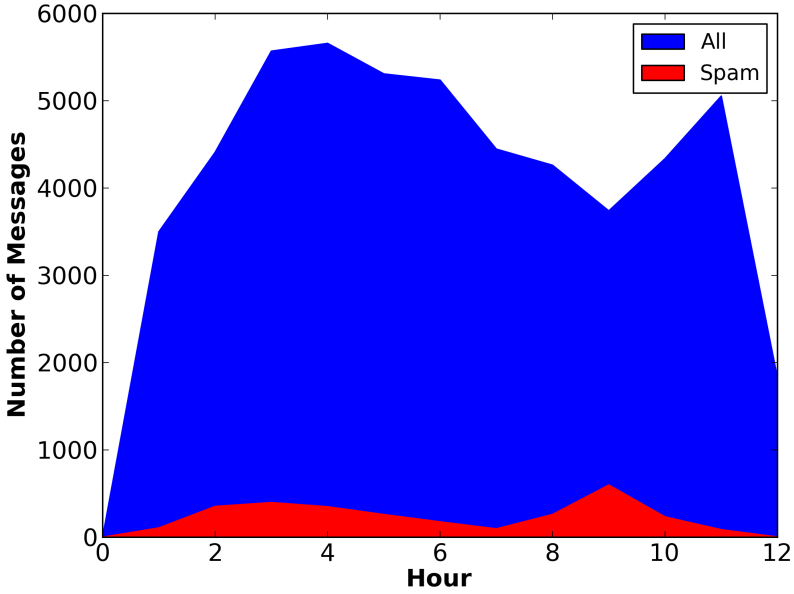


Figure 5.4: All messages and spam messages associated with a popular topic "#LookAtMeNow". Spammers quickly started posting spam messages as the topic became popular.

way, users themselves have self-selected for interest in the topic and made themselves susceptible to collective attention spam. For example, as users become interested in a popular YouTube video and the view count increases, spammers can insert spam comments. As users begin monitoring Twitter for a trending topic, spammers (with knowledge of these popular topics derived directly from Twitter) can insert spam messages associated with the topic (e.g., by including the relevant hashtag). To illustrate, Figure 5.4 depicts the incidence of spam for the Twitter trending hashtag "#LookAtMeNow" over its 12-hour lifespan, where we can observe that spammers quickly posted spam messages as the topic becomes popular (details of the data and how spam was labeled provided in the following section).

While email and social spam have been the subject of considerable study, there is a significant gap in our understanding of the susceptibility of social systems to

collective attention threats. Our goal in this chapter is to begin to understand this phenomenon better, building on our preliminary effort to detect collective attention spam reported in [75]. How susceptible are social systems to malicious attacks? What strategies by malicious users are most effective? And least effective? How do users of a system access items of interest and how does this affect their exposure to threats? Can a system automatically inoculate itself from emerging attacks? What kinds of countermeasures can be deployed and how effective are they at limiting the effectiveness of malicious users?

5.1.1 Our Approach

Answering these questions is challenging. Large-scale social systems are typically proprietary and responsible to their current user base, so it is infeasible to automatically "stress-test" such a system by subjecting it to hundreds or thousands of malicious users. An alternative is to take a representative snapshot of a system and measure the current level of threats in the system and characterize their reach and effectiveness. However, this approach alone may not be suitable for understanding the system's future state, as social systems are constantly evolving. Hence, we take a two fold approach. First, we take a *data-driven modeling* approach, in which we simulate a large-scale social system based on parameters derived from a real system. In this way, we can vary system parameters – like the fraction of malicious users in the system, their strategies, and the countermeasures available to system operators – to explore the resilience of these systems to threats to collective attention. Based on our data-driven model, we identify two possible countermeasures – a rule-based filter and a supervised detector. Since many instances of collective attention are bursty and unexpected, it is unclear if such countermeasures – though effective in simulation – may be effectively deployed based on the first moments of a bursting

phenomenon in a real system. Hence, we pair the data-driven model with a *comprehensive evaluation over a Twitter system trace*, in which we evaluate the effectiveness of countermeasures deployed based on the first moments of a bursting phenomenon in a real system.

5.1.2 Summary of Key Findings

In summary, this chapter presents the first comprehensive study of collective attention spam in social systems.

- Through our data-driven model, we find that social systems are extremely susceptible to collective attention spam. With spammers accounting for only 5% of all users, we find that every legitimate user can be exposed to spam. At even higher spammer penetration, the social system becomes unusable with spam dominating.

- We find that strategically organized spammers can collude to selectively push particular spam payloads, increasing the exposure of legitimate users to spam content.

- On a positive note, we find that the countermeasures deployed early in the lifecycle of a collective attention attack can dramatically reduce the amount of spam in the system. Through testing over 20 million Twitter messages, we validate the model findings and see that these countermeasures can effectively identify threats to collective attention early in the lifecycle with 98% accuracy, reducing "spamness" up to 73% and providing a shield for unsuspecting social media users.

5.2 A Data-Driven Model for Studying Collective Attention Threats

In this section, we present a data-driven modeling approach for simulating collective attention and threats. Our goal is to answer questions about the susceptibility of social systems to collective attention threats and to explore techniques for limiting this impact. We begin by describing how both good and bad users post content to the social system, and how the system itself supports information access. We describe how the model is seeded and validated, and then we investigate (i) threats from individual spammers; (ii) threats from coordinated spammers; and (iii) finally, we examine countermeasures. In the following section, we revisit these model-driven results through an experimental study over a real Twitter trace.

5.2.1 System Model

We consider a social system of interest $\mathcal{S}$, consisting of a set of content items $\mathcal{C}$ (e.g., Facebook status updates, photos, videos, etc.), a set of topics $\mathcal{T}$ for which each content item is associated (e.g., the "London Olympics" topic, the "Steve Jobs" topic, etc.), and a set of users $\mathcal{U}$, who participate in the system by posting and viewing content items. For example, a user in $\mathcal{U}$ may post a tweet "Thank you #SteveJobs The world will miss you", where the tweet is associated with the topic indicated by the hashtag #SteveJobs. Similarly, a user may post a video to YouTube associated with the "London Olympics" topic by including a tag or descriptive text at the time of upload. We use the symbols u, c, and t to denote a user in $\mathcal{U}$, a content item in $\mathcal{C}$, and a topic in $\mathcal{T}$.

5.2.2 Posting Model

To populate a social system, we initialize the system with a set of topics and a set of users. To model users in a social system, we define two sets of users: a good user

set U^+ and a bad user set U^-. Good users post content items that are associated with a "correct" topic. Bad users, on the other hand, post content items that are irrelevant to the topic they are associated with. For example, a bad user may post a spam video, but intentionally describe it as a "London Olympics" video. When users post to the system, we assume they have access to both the set of topics $\mathcal{T}$ as well as the current subset of "popular" topics $\mathcal{T}_{pop}$ (in practice, these popular topics may be known to users via prior knowledge or explicitly advertised by the system, as in the case of Twitter trending topics or popular YouTube videos). The system proceeds in step-wise fashion; at each time increment, users generate content items according to a particular posting model. Good users post according to the *good user model*:

Good User Model:

for each user $u \in U^+$ do

with probability γ^+ decide to post:

with probability δ^+:

select a popular topic $t \in \mathcal{T}_{pop}$ and relevant item c;

else:

select at random a topic $t \in \mathcal{T}$ and relevant item c.

At each time increment, a good user chooses to post something with the user content generation probability γ^+. If a user decides to post a content item, an already popular topic is selected with probability δ^+; alternatively, the user decides to post to a random topic. A bad user follows a similar process, but always posts spam content items:

Bad User Model:

for each user $u \in U^-$ do

with probability γ^- decide to post:

with probability δ^-:

a popular topic $t \in \mathcal{T}_{pop}$ and spam item c;

else:

select at random a topic $t \in \mathcal{T}$ and spam item c.

Notice that the user content generation probability γ and the popular topic probability δ may vary between the good and bad user models. As part of our data-driven simulation, we will vary these parameters to reflect different spammer behaviors. For example, a spammer may adopt a high rate of content generation relative to good users (e.g., $\gamma^- \gg \gamma^+$) in an attempt to flood the system with spam content. Alternatively, a spammer seeking to maximize their potential audience may choose to focus only on popular topics and so adopt a popular topic probability much greater than the good user model (e.g., $\delta^- \gg \delta^+$).

5.2.3 Collective Attention Access Models

Given the approach for populating a social system, we now consider how users access the content posted in the system. We assume that users monitor topics by one of two methods:

- *By Recency*: In the first access model, users interested in a topic access the k-most recently posted items related to the topic. This recency approach is akin to the "Most Recent Uploads" functionality on YouTube, viewing comments associated with a blog by their posting order (from recent to oldest), and Twitter's basic search.

- *By Relevance*: The second access model imposes a relevance ordering over

content items associated with a topic. This relevance-based approach may incorporate the popularity of an item (e.g., rank images in order of the number of clicks they have accumulated), content and link-based ranking (e.g, applying IR principles), or learning-to-rank methods. For modeling purposes, we assume that content items are ranked by their occurrence count, with all duplicates removed to maintain diversity (i.e., item c_i posted 20 times is ranked first; item c_j posted 10 times is ranked second; and so on).

User interest in a topic is based on the amount of content items posted to the topic. So, if topic t_i is the most popular topic according to the good and bad user models, then it will be monitored by the most users. In this way, as items become more bursty, collective attention in them rises accordingly.

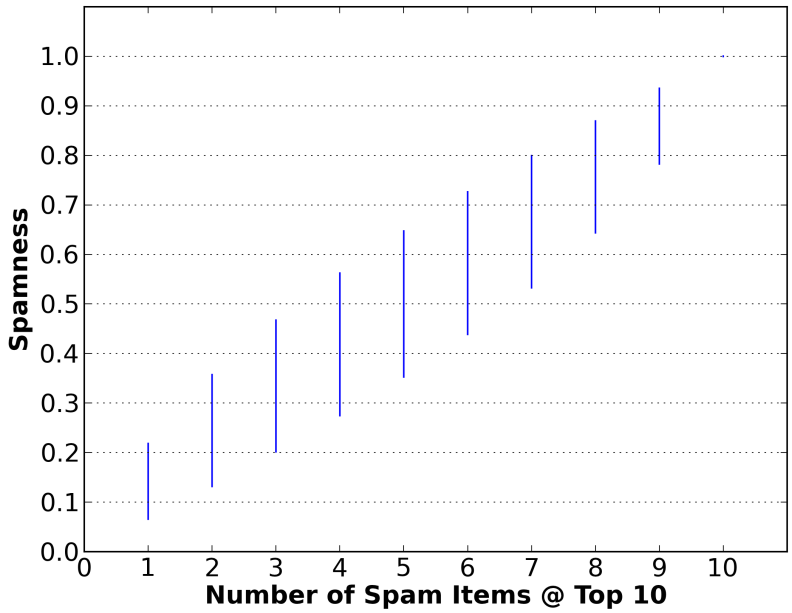


Figure 5.5: Spamness variation by a number of spam items in top 10 search result

5.2.4 Measuring Spam Impact

To evaluate the impact of bad users on inserting spam into the system, we measure the overall *spamness*, which is a measure similar to NDCG@k [64]. Note that NDCG@k is a metric to measure the quality of top k search result. For a user accessing topic t, we have:

$$Spamness(t,k) = \frac{\sum_{i=1}^{k} w(c_i) * \frac{1}{log_2(1+i)}}{Norm(k)}$$

where

$$w(c_i) = \begin{cases} 1, & \text{if } c_i \text{ is a spam content item;} \\ 0, & \text{otherwise.} \end{cases}$$

and k is the number of items (e.g., messages or tweets) shown in a search result by a search system, and $Norm(k) = \sum_{i=1}^{k} \frac{1}{log_2(1+i)}$ is a normalizing constant. Spamness varies from 0 to 1, with 0 signifying no impact to 1 signifying all of the items viewed by a user are spam. If users view 10 items at a time ($k = 10$), spamness varies over the number of spam items in the top 10 search result as shown in Figure 5.5. With three spam items, spamness ranges between 0.200 and 0.469, depending on where the spam items are located in the search result; if they are positioned in the top, spamness will be high. As a rule-of-thumb we consider a *spamness of 0.2 or greater to indicate a high-level of spam*, corresponding to a user encountering 3 or more spam items for every 10 items encountered.

5.2.5 Seeding and Validating the Model

To accurately model real social systems for a *data-driven simulation*, we require baseline parameter settings. However, there are no standard datasets of collective attention spam. Hence, we sampled a collection of popular topics and their associated

Table 5.1: A sample of 101 popular topics. In total, there are ~13m messages, of which 3.7% are spam

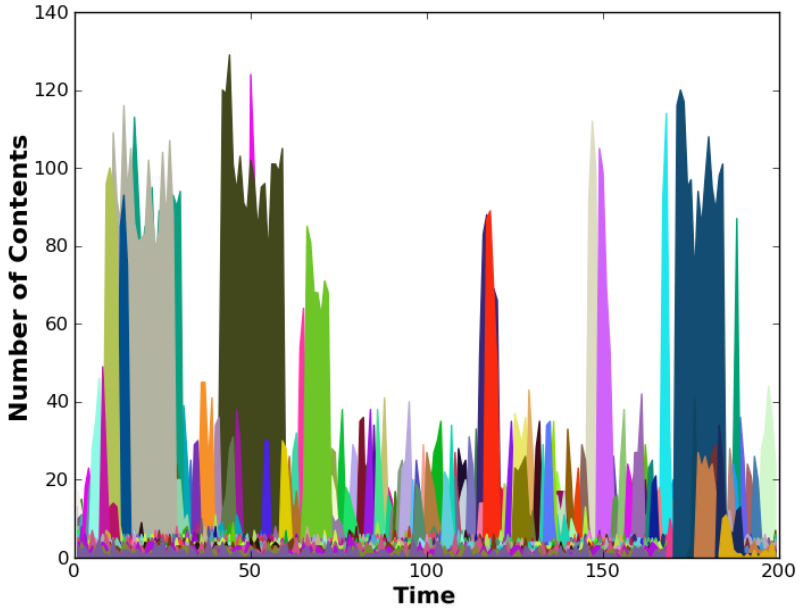
No.	Topic	Total Lifespan	# of messages
1	#SomeWhereInTheHood	12 hrs 48 mins	93,871 (2.5% spam)
2	#ThatOneEx	10 hrs 21 mins	58,217 (3.0% spam)
3	#thewayiseeit	18 hrs 06 mins	201,682 (4.6% spam)
4	#LawsMenShouldFollow	16 hrs 30 mins	181,524 (4.0% spam)
5	#DearOOMF	18 hrs 18 mins	139,559 (1.1% spam)
6	#stupidquestion	7 hrs 38 mins	60,139 (0.7% spam)
7	#EnoughIsEnough	10 hrs 20 mins	71,259 (1.6% spam)
8	#DearHair	17 hrs 06 mins	123,823 (4.5% spam)
9	#fightingwords	15 hrs 37 mins	123,953 (3.3% spam)
10	#ThingsMenShouldntTextEachOther	10 hrs 21 mins	91,398 (2.9% spam)
11	#ThatHighMoment	23 hrs 37 mins	169,632 (2.3% spam)
12	#TheyNeedToBringBack	34 hrs 34 mins	222,175 (2.1% spam)
13	#YouNeedToShutUp	24 hrs 45 mins	94,464 (3.0% spam)
14	#WhatYouShouldKnowAboutMe	23 hrs 48 mins	249,534 (2.1% spam)
15	#ThingsThatGetMePissed	23 hrs 28 mins	236,488 (1.9% spam)
16	#MeAndYouCantDate	21 hrs 49 mins	209,341 (2.4% spam)
17	#HowToMakeItInAmerica	30 hrs 57 mins	124,608 (4.5% spam)
...	...	...	...
84	#ThingsIAlwaysSeeOnMyTL	19 hrs 20 mins	88,685 (3.5% spam)
85	#FavoriteWaleLyric	6 hrs 00 mins	50,577 (1.2% spam)
86	Happy Halloween Everyone	7 hrs 54 mins	72,799 (1.5% spam)
87	#GamesToDescribeSex	6 hrs 24 mins	32,106 (3.0% spam)
88	#ThingsLongerThanKimsMarriage	41 hrs 22 mins	302,389 (3.4% spam)
89	#ThingsOnMyMind	17 hrs 38 mins	154,335 (2.5% spam)
90	#6wordstories	13 hrs 14 mins	119,558 (1.2% spam)
91	#10ThingsIFindAttractive	14 hrs 22 mins	101,741 (1.3% spam)
92	#GrandTheftAutoMemories	19 hrs 49 mins	118,261 (3.1% spam)
93	#2011musictaughtme	21 hrs 10 mins	187,845 (2.3% spam)
94	#LiesThatAlwaysWorked	26 hrs 28 mins	219,382 (6.3% spam)
95	#TwitterPeopleILove	39 hrs 12 mins	153,024 (2.0% spam)
96	#ivealwayswantedto	47 hrs 42 mins	350,043 (1.1% spam)
97	#ThingsICantLiveWithout	19 hrs 27 mins	140,761 (2.5% spam)
98	#DoctorsBetterThanConradMurray	12 hrs 02 mins	68,370 (12.2% spam)
99	#WhaILove	24 hrs 02 mins	174,695 (3.1% spam)
100	#hometownslogans	22 hrs 02 mins	59,529 (5.5% spam)
101	#ThingsThatYouShouldKnow	13 hrs 09 mins	95,542 (3.6% spam)

messages from Twitter between September and November 2011. We polled Twitter's *trending topics* every 5 minutes and collected the messages associated with each trending topic. In total, we collected 19,275,961 messages posted by 3,989,563 users across 354 trending topics.

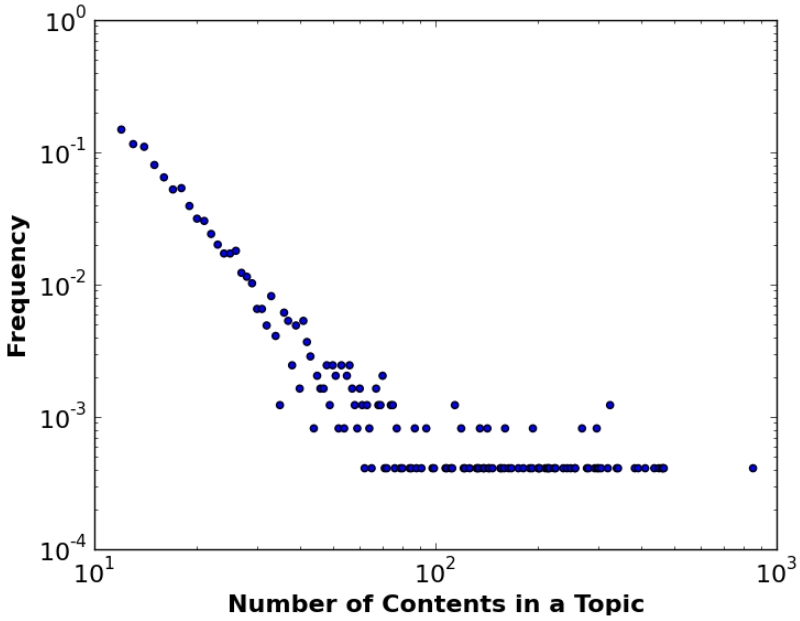
But how many of these messages are actually spam? It is important to find a baseline estimate so that the model parameters can be seeded realistically. To assess the amount of spam in the dataset, we systematically checked whether a user associated with a message had been suspended by Twitter for engaging in spam behaviors. If an account is suspended, Twitter will redirect the request to a standard "suspension" page: http://twitter.com/account/suspended. Not all suspended accounts may have actually engaged in spam, so we further assessed these accounts. Concretely, we randomly sampled 200 messages each from the messages posted by suspended accounts and from those posted by non-suspended accounts. Two human judges manually labeled the 400 messages as either spam or non-spam. From the non-suspended accounts, 199 out of 200 messages sampled were labeled as non-spam messages. From the suspended accounts, 187 out of 200 messages sampled were labeled as spam messages. Based on this high accuracy, we make the simplifying assumption that all messages posted by suspended users are indeed spam so that all ~19 million messages can be automatically labeled.

A sample from the top-101 topics with the most messages is shown in Table 5.1. Together, these topics account for 12,954,965 messages. A topic has on average 132,725 messages and 3.7% of them are generated by spammers, who account for around 1.5% of all accounts in the dataset.

Following the observed spam amount in the real data, we set the fraction of spammers in the system as 1.5%. We then varied the content generation probability (γ), and probability of picking popular topics (δ) to find an initial model setting that em-



(a) Topic distribution



(b) Log-log graph

Figure 5.6: The left figure depicts a topic distribution generated by the model. Each color denotes a topic. The right figure depicts a log-log graph showing the frequency of number of content items associated with each topic in the simulation data. The heavy-tailed distribution is similar to bursty social media.

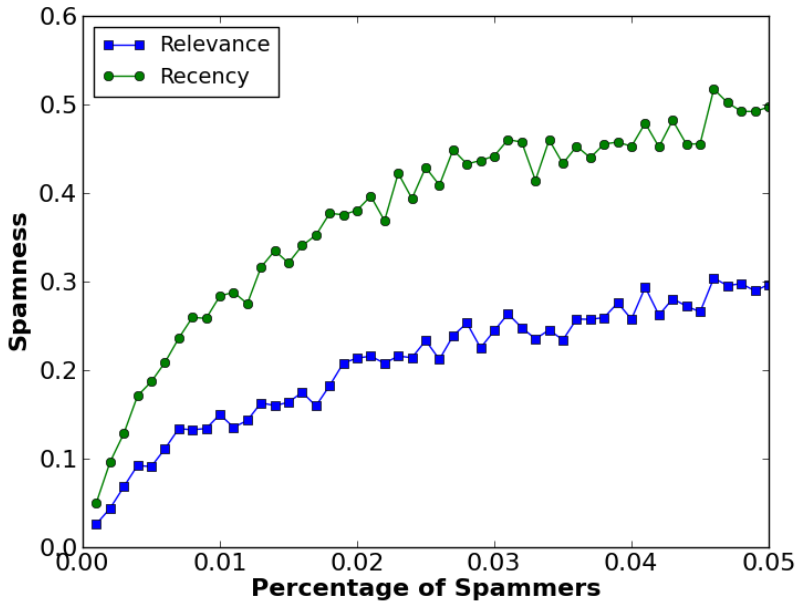


Figure 5.7: Evaluating the impact of increasing the fraction of spammers in the system from 0 to 5% (i.e., $0 \sim 0.05$ in the x-axis)

ulated the real data distribution. Arriving at initial settings of $\gamma^+ = 0.1$, $\gamma^- = 1.0$, $\delta^+ = 0.4$, and $\gamma^- = 0.75$, we arrive at a topic distribution shown in Figure 5.6(a) following the heavy-tailed distribution as shown in Figure 5.6(b), which is similar to the expected distribution of bursty social media. Note that these initial settings fit our intuition, with bad users posting more often than good users and posting exclusively to popular topics. We find that small changes to these parameters make little qualitative difference to the conclusions drawn in the following. Based on these initial parameter settings, we next explore the following research questions: how susceptible are social systems to malicious attacks? what strategies by malicious users are most effective (e.g., individual attack, group-based coordinated attack, or combination of individual attack and coordinated attack)? What kinds of countermeasures can be deployed and how effective are they at limiting the effectiveness of malicious users?

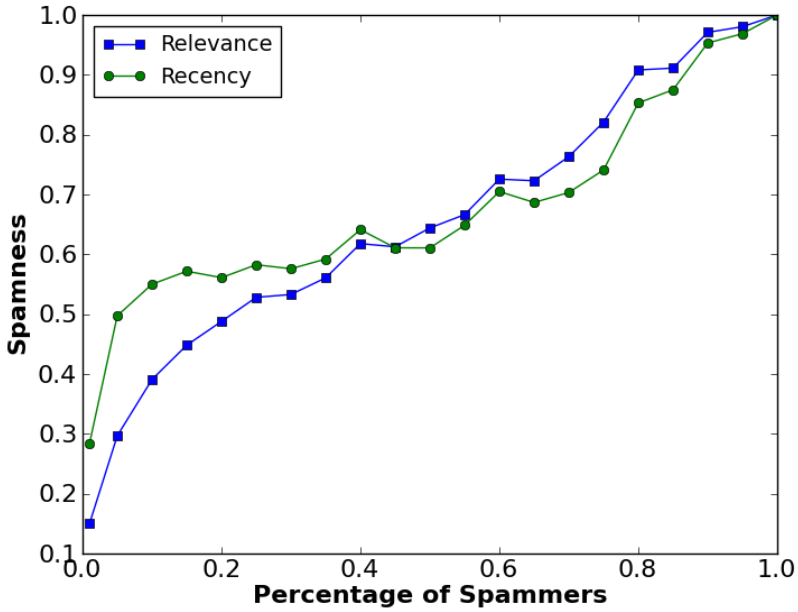


Figure 5.8: Evaluating the impact of increasing the fraction of spammers in the system from 0 to 100% (i.e., $0 \sim 1$ in the x-axis)

5.2.6 Threats from Individual Spammers

We've seen in one example system (Twitter) that about 1.5% of users are collective attention spammers. Suppose this fraction of spammers increases. What impact will this have on the amount of spam that legitimate users are exposed to? For this first experiment, we vary the fraction of spammers from 0 to 100%, (we keep the same γ and δ, but increase the fraction of spammers). We see in Figure 5.7 that naturally, the spamness of the system increases with an increasing number of spammers. Interestingly, the recency-based access approach fairs significantly worse than the relevance-based one, crossing the spamness threshold of 0.2 when less than 1% of all users are spammers. The relevance-based approach is less susceptible to spam since individual spammers cannot selectively *push* particular items; in contrast so long as users access the most-recent items, spammers can easily insert spam items that will

be viewed. Although the relevance-based approach is more resistant to spammers, if the fraction of spammers were to increase only slightly to 2%, then the spamness threshold would be passed. As the fraction of spammers increases beyond 5%, we see in Figure 5.8 that neither access approach can significantly limit the amount of spam in the system, with both approaches near or above a spamness of 0.5 with just 20% spammers. At even higher ranges, presumably the social system would become unusable and unappealing to legitimate users, with spam dominating.

5.2.7 Threats from Coordinated Spammers

The threat so far has considered individual spammers who do not coordinate their actions; that is, there is no common spam payload shared across multiple spammers for perhaps increasing its reach. Hence, in this next experiment we consider a coordinated spam approach in which spammers are assigned to a group which is associated with a common pool of spam payloads. For the following experiment, we assume that spammers share a common pool of spam payloads, and we vary the number of spam payloads.

Using this coordinated approach, we observe in Figure 5.9 that the recency-based approach is largely unaffected, but that it remains highly susceptible to spam. The relevance-based approach shows that spammers have a potential "sweet spot" for targeting spam. At a low number of payloads, the spamness is relatively low since the spammers promote a few payloads which possibly pollute one or two out of the top-k results. As the number of payloads increases, the coordinating spam group can achieve an impact equal to or even better than under the recency-based approach. However, as the number of payloads continues to increase, the effectiveness for the coordinating spam group falls, because the power promoting payloads is distributed across too many payloads, meaning no single one can penetrate the top-k, and hence

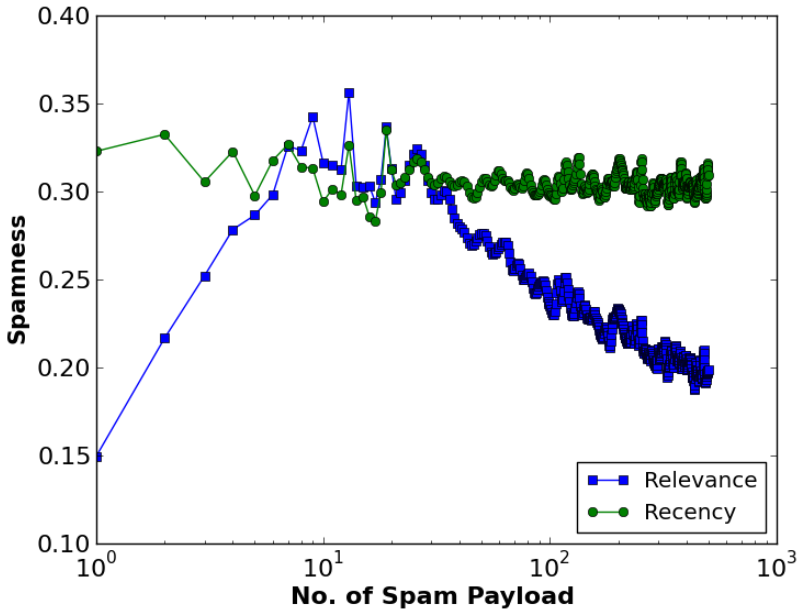


Figure 5.9: Coordinated Spam: By focusing their efforts, groups can achieve even higher impact.

be exposed to end users interested in the topic.

What if spammers adopt a mixed strategy, balancing between the individual and the coordinated approach? Figure 5.10 compares the robustness of the two access approaches to a mixed spam strategy. We observe the continued poor resistance of the recency-based approach. To effectively target the relevance-based approach, spammers need only adopt very little collusion (i.e., with 20% adopting the group strategy, spamness passes the 0.20 threshold). At even higher-levels of collusion ($\geq$80%), spammers are even more effective than under the recency-based approach, further confirming the dangers of strategically organized spammers.

5.2.8 Countermeasures

So far we have seen that the relevance-based access approach is generally more resistant than recency to collective attention spam, but that both are extremely

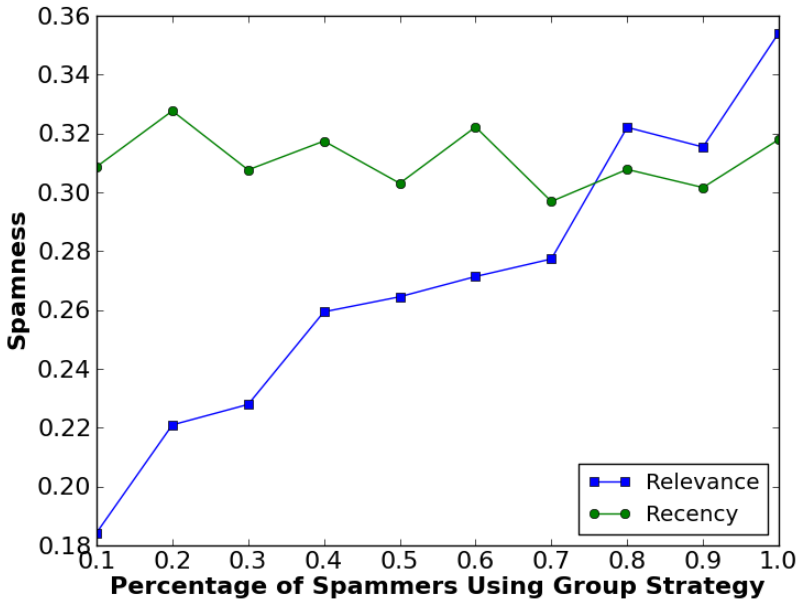


Figure 5.10: With as few as 20% adopting the group strategy, spamness passes the 0.20 threshold.

susceptible to only slight changes in the fraction of spammers and to strategic efforts to coordinate spam behavior. We now consider the impact of countermeasures to collective attention spam to better understand under what scenarios spam may be detected and filtered. We consider two countermeasures:

5.2.8.1 Countermeasure 1: Rule-Based Filtering

The first is a rule-based filtering approach, which is potentially easy to deploy in a real-system, but that may not be adaptable to changes in behavior by malicious users. We consider a simple rule that considers the ratio of users to content items:

$$PayloadScore(t,p) = 1 - \frac{\#\ of\ distinct\ users}{\#\ of\ content\ items}$$

where t and p denote a topic and a payload, respectively. The rule-based filtering approach counts # of content items containing a payload p in the topic t and #

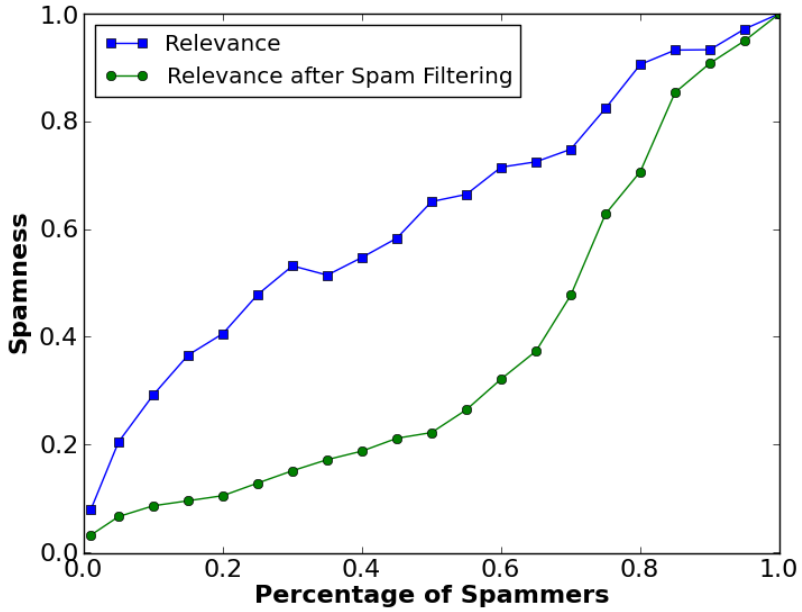


Figure 5.11: Applying a simple rule-based countermeasure greatly reduces spamness, but is not effective against strategic behavior.

of distinct users who generate the content items, and then filters out content items exceeding a threshold. The intuition is that collective attention spammers may strategically use common payloads, so if fewer users post more of the same item (e.g., a common URL or spam image) they can be filtered out.

Setting a threshold of 0.1 and applying this countermeasure to the recency-based approach makes little difference since the spamness is already so high (as we saw in previous experiments). However, applying this countermeasure to the relevance-based approach results in a dramatic reduction in spamness as shown in Figure 5.11. While encouraging, it is not obvious that such improvements could be observed in practice, with spammers strategically changing their behavior. We'll revisit the effectiveness of such a rule-based countermeasure in the following section.

5.2.8.2 *Countermeasure 2: Supervised Classification*

A second countermeasure is a spam detector relying on supervised classification principles. The intuition is that system operators may be able to sample evidence of spam early in the lifecycle of a collective attention phenomenon (e.g., sampling and labeling spam tweets from a trending topic). Based on this early evidence, perhaps an effective classifier can be quickly deployed for filtering out subsequent spam. To evaluate such an approach, we consider two detectors: a low-accuracy spam detector that can only filter out 40% of all spam items as they enter the system, and a high-accuracy spam detector that can filter out 90% of all spam items. As an example, a low-accuracy detector may be built on imperfect crowdsourced spam labeling, while a high-accuracy detector may have been refined over large carefully curated spam datasets.

We show in Table 5.2, the hypothetical performance of two detectors versus the baseline (no countermeasure) case over a 90 minute "run" of the system model. At each one-minute time unit, users post content, the detectors are applied, and the spamness of the results from the access approaches are calculated. We see over the 90 minutes that even the low-accuracy spam detector achieves good results, pushing the spamness well below the 0.2 threshold. The high-accuracy performs very well, with spamness below 0.06 in all cases. When increasing the fraction of spammers in the system, we find similarly robust results suggesting that effective countermeasures are a necessity for countering threats to collective attention in social media.

5.3 Countermeasure Deployment on Twitter

Based on the data-driven model, we have identified the need for collective attention spam countermeasures. Though effective in simulation, it is unclear of such countermeasures are achievable in real social systems. Since many instances of col-

Table 5.2: Evaluating the potential effectiveness of a low-accuracy (40%) and a high-accuracy (90%) collective spam detector

Access Approach	Avg	Min	Max
Recency	0.228	0.198	0.279
+ low-accuracy detection (40%)	0.120	0.102	0.156
+ high-accuracy detection (90%)	0.041	0.030	0.052
Relevance	0.176	0.148	0.215
+ low-accuracy detection (40%)	0.115	0.099	0.138
+ high-accuracy detection (90%)	0.036	0.027	0.044

lective attention are bursty and unexpected, it is difficult to build spam detectors to pre-screen them before they arise. Hence, in this section we study the viability of quickly deploying collective spam countermeasures based on the first moments of a bursting phenomenon. We examine the Twitter trace described in Section 5.2.5, consisting of 101 topics associated with 13 million messages. We investigate when a countermeasure may be optimally deployed to a trending topic. Early deployment of a supervised classifier has the potential to greatly reduce spam subsequently associated with the topic, but at a risk of learning only a limited model and resulting in less robust classification (resulting in higher false positives and false negatives). Late deployment has less potential to reduce the total amount of spam (since presumably most of it will have already arrived by the time of deployment), but will be more robust in its detection.

5.3.1 Metrics

To evaluate the quality of a countermeasure, we augment the spamness measure with several standard spam metrics: accuracy, false positive rate (FP) and false negative rate (FN). In the confusion matrix in Table 5.3, a represents the number of correctly classified spam items, b represents the number of spam items misclassified as non-spam, c represents the number of non-spam items misclassified as spam, and

d represents the number of correctly classified non-spam items. Accuracy is the fraction of correct classifications: $(a+d)/(a+b+c+d)$. The False Positive rate is $c/(c+d)$, and the False Negative rate is $b/(a+b)$.

Table 5.3: Confusion matrix

		Predicted	
		Spam	Non-spam
Actual	Spam	a	b
	Non-spam	c	d

Additionally, we measure the *total spam detected (TSD)* over a topic's lifespan:

$$TSD_{topic}(\%) = \frac{\#\ of\ detected\ spam}{total\ \#\ of\ spam\ in\ the\ topic}$$

The goal of a countermeasure is to reduce the most amount of spam, so *total spam detected* complements the traditional measures of accuracy, false positive rate, and false negative rate. For example, a countermeasure that is deployed late in the lifecycle of a topic may be very robust, with high accuracy and low false positives and false negatives, but may only detect a small fraction of all spam. Why? Because most of the spam occurred *before* the countermeasure was ever deployed. An effective countermeasure should balance accuracy and the other measures with the total spam detected, so that unsuspecting users are shielded from spam.

5.3.2 Countermeasure 1: Rule-Based Filtering

We begin by considering a static rule-based filtering approach, based on the principles described in the previous section. In our observations of Twitter trending topics, we see that many spam messages contain a common advertisement or URL

payload. In contrast, messages posted by legitimate users are more varied. For example, for the topic #DearHair, we noticed similar messages of the form:

@9rappermode9 OMG, #DearHair RT Have you seen this??WTF how could it happen with hair?? : http://t.co/xcPx6JFe

@_enoughsaid_ OMG, #DearHair RT Have you seen this??WTF how could it happen with hair?? : http://t.co/fVD4UAbC

where both URLs redirect to the same spam destination.[†]

We can define the *payload* as the message content after eliminating all hashtags, usernames, and URLs. In the example, the payload is *OMG, RT Have you seen this? ?WTF how could it happen with hair??*. With this payload definition and the simple payload score rule as presented in the previous section: $PayloadScore(t, p) = 1 - \frac{\#\ of\ distinct\ users}{\#\ of\ contents}$, we evaluate how many spam messages can be detected from the Twitter trace. In the best case, with a threshold of 0.1, we find that only 20% of all spam messages across all 13 million messages can be filtered (i.e., the average TSD is 20%).

While the space of all potential rules is large, we can see that a rule-based approach is likely to be insufficient by itself to reduce collective attention spam. Hence, we next explore in greater detail the supervised classification approach, which promises potentially more adaptability to ongoing collective spam prevention.

[†]To further illustrate the potential impact of collective attention spam, we accessed the bitly records for these URLs. URLs in 102 messages redirect to the same destination via various bitly URLs. One of the bitly URLs had been clicked a total of 1,424 times, indicating the effectiveness of targeting collective attention (available at http://bitly.com/usaend+).

5.3.3 Countermeasure 2: Supervised Classification

We next investigate the viability of a supervised classifier for detecting collective attention spam that targets popular topics. Our goal is to predict whether a message m posted to a trending topic (i.e., by including the associated hashtag or keyword) is a spam message through a classifier c:

$$c : m \rightarrow \{spam, non - spam\}$$

Our classification approach is that given a set of messages associated with a popular topic, we create a training set containing messages generated before a deployment time x since the topic has become popular, and the rest of the messages associated with the topic belong to a testing set. We create multiple pairs of training and testing sets for different hourly deployment times. For example, for a trending topic with a 10-hour lifespan, we consider deploying the countermeasure at hour 1, at hour 2, and so on up to hour 9. In this way, we independently create 9 training sets, each of which contains messages posted during the first 1 hour, 2 hours, and so on up to 9 hours, respectively. Corresponding to the training sets, we create 9 testing sets containing the rest of messages.

Since collective attention spam targets topics as they become popular, detecting these spam messages as soon as possible is very important. Our goal is to explore the trade-off between early deployment and late deployment. Under what circumstances does a supervised classifier filter collective attention spam? For the classifier, we adopt a decision tree based Random Forest classifier as a supervised learning method following previous success reported in [77].

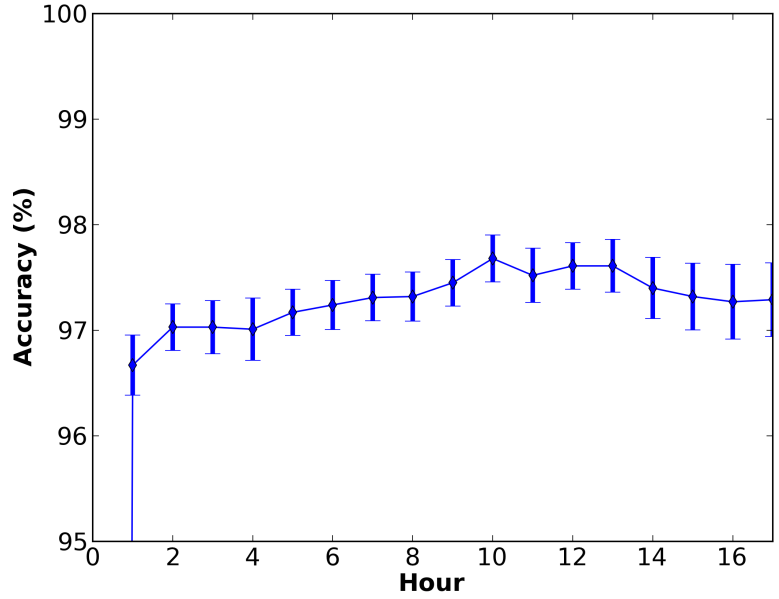
Table 5.4: Top 10 features

Feature	χ^2 Value	Avg Spam	Avg Good
# of URLs	56,795	0.67	0.01
length of message	13,700	85.27	75.7
length of payload	11,398	46.86	48.43
# of words in payload	6,497	9.13	10.31
# of words in message	6,407	10.63	11.03
# of hashtags	3,343	1.25	1.1
# of @mentions	3,162	0.1	0.54
is retweet	2,115	0.06	0.38
has exclamation mark	1,797	0.23	0.14
has question mark	843	0.08	0.04

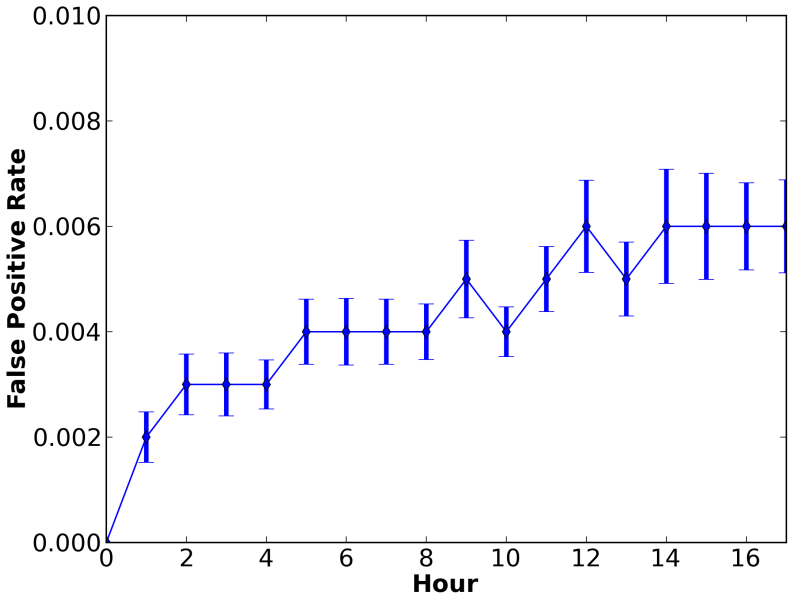
5.3.3.1 Feature Selection

Before building a classifier, finding good features is very important for high accuracy. We build classifiers based on 10 features extracted from each message: (1) # of URLs; (2) # of hashtags; (3) # of @mentions; (4) is a message retweeted?; (5) does a message contain a question mark?; (6) does a message contain an exclamation mark?; (7) the length of a message; (8) the number of words in a message; (9) the length of a payload (again, given a message, we first remove @mention, URLs and hashtags and call the remaining text a payload); and (10) the number of words in a payload.

In order to measure whether each feature has power to distinguish between spam and non-spam messages, we compute its χ^2 value. If a feature has a positive χ^2 value, it will have distinguishing power. Table 5.4 presents the average χ^2 values of the 10 features across 101 topics. We observed that all features have power to distinguish between spam and non-spam messages. For example, we see that the number of URLs per message is 0.67 for spam, but only 0.01 for non-spam messages.



(a) Accuracy



(b) False positive rate

Figure 5.12: Evaluating Countermeasure 2: Supervised Classification. Average accuracy and false positive rate reported across 101 topics.

Table 5.5: On average, the supervised classifier countermeasure achieves 98% accuracy, detecting 50% of all spam messages

Topic	Tr. Time	Acc	FP	FN	TSD
#SomeWhere...	2 hrs	99.09	0.003	0.247	72.43
#ThatOneEx	8 hrs	98.85	0.002	0.152	62.11
#thewayiseeit	5 hrs	99.29	0.003	0.113	47.67
#LawsMenSh...	4 hrs	99.51	0.003	0.083	49.01
#DearOOMF	11 hrs	99.42	0.001	0.231	57.39
#stupidque...	7 hrs	99.83	0	0.055	41.33
#EnoughIsE...	2 hrs	99.38	0.001	0.241	67.93
#DearHair	2 hrs	98.71	0.002	0.248	70.98
#fightingw...	6 hrs	99.4	0.001	0.092	73.04
#ThingsMen...	1 hr	99.72	0.001	0.064	81.54
#ThatHighM...	10 hrs	99.37	0.002	0.195	61.19
#TheyNeedT...	5 hrs	99.48	0.001	0.163	71.11
#YouNeedTo...	1 hr	99.24	0.001	0.216	74.89
#WhatYouSh...	3 hrs	99.19	0.001	0.324	62.35
#ThingsTha...	6 hrs	99.41	0.001	0.24	63.28
#MeAndYouC...	8 hrs	98.71	0.001	0.391	48.1
#HowToMake...	13 hrs	98.71	0.001	0.288	55.4
...	...	...	...	...	...
#FavoriteW...	4 hrs	98.32	0.004	0.401	55.41
Happy Hallo...	5 hrs	99.07	0.002	0.484	21.03
#GamesToDe...	3 hrs	97.9	0.004	0.46	47.95
#ThingsLon...	10 hrs	97.08	0.004	0.624	34.78
#ThingsOnM...	2 hrs	97.75	0.004	0.709	28.49
#6wordstor...	5 hrs	98.6	0.001	0.916	6.14
#10ThingsI...	12 hrs	98.23	0.008	0.261	48.59
#GrandThef...	1 hrs	97.23	0.003	0.769	22.52
#2011music...	10 hrs	96.6	0.008	0.682	26.17
#LiesThatA...	16 hrs	86.7	0.002	0.922	6.93
#TwitterPe...	6 hrs	99.41	0.003	0.187	64.2
#ivealways...	15 hrs	98.99	0.002	0.75	17.84
#ThingsICa...	3 hrs	98.66	0.005	0.331	54.59
#DoctorsBe...	3 hrs	97.75	0.006	0.096	76.82
#WhatILove	3 hrs	98.17	0.005	0.496	36.33
#hometowns...	5 hrs	96.97	0.013	0.344	46.61
#ThingsTha...	8 hrs	98	0.006	0.312	33.09
Average	**5 hrs**	**97.57**	**0.007**	**0.384**	**50.16**

5.3.3.2 Detection Across 101 Topics

Next, we build a collective attention spam classifier over each of the 101 popular topics and evaluate them. For each topic, we build a classifier every hour since the topic has become popular. In total, we built 2,020 classifiers for 101 topics (i.e., 2,020 classifiers = 101 topics * 20 classifiers). The first question is whether spam messages detected in the early stages may accurately identify spam that follows as a topic becomes popular. Hence, in Figure 5.12(a) we report the average classification accuracy for training sets of varying time windows. We measure accuracy for each topic independently and then report the average accuracy in each hour. That is, 1 hour in the x-axis means that the training set consists of messages posted within 1 hour after the topic became a trending topic (and hence, made available to spammers as a potential target), and the testing set consists of messages posted after 1 hour. The y-axis shows the accuracy when we use the training set to build a classifier and predict labels of the messages in the testing set. This experiment emulates a real deployment scenario of such a collective attention spam detector, in which partial data is available for predicting future spam. Notice that as the training set grows in size the classification result becomes better. Figure 5.12(b) shows the false positive rate – indicating how many real non-spam messages are classified as spam messages by the classifier. Overall, the false positive rate is low.

As we have discussed, however, the goal is not only to have high accuracy and low false positives, but also to detect more spam messages as early as possible. In Table 5.5 we present a sample of the detection results, along with the average result across all 101 topics. Each topic's best training time varies depending on the volume of generated messages and the number of spam messages before the training time. Overall, building a classifier with the first five hours' messages gives

us 97.57% accuracy, 0.006 FP, 0.384 FN and 50.16% total spam detected (i.e., how many spam messages out of all spam messages in the topic a classifier detected correctly). Not only does this countermeasure outperform the rule-based filtering approach (50% TSD versus 20% TSD), but it has the advantage of being adaptable to future spammer behaviors (so long as the feature set is maintained). We also observe a high variability in the TSD across topics; some topics are easy for spam detection (with $TSD > 80\%$), while others are very difficult. This suggests that our preliminary feature set could be refined to better target these difficult-to-detect cases.

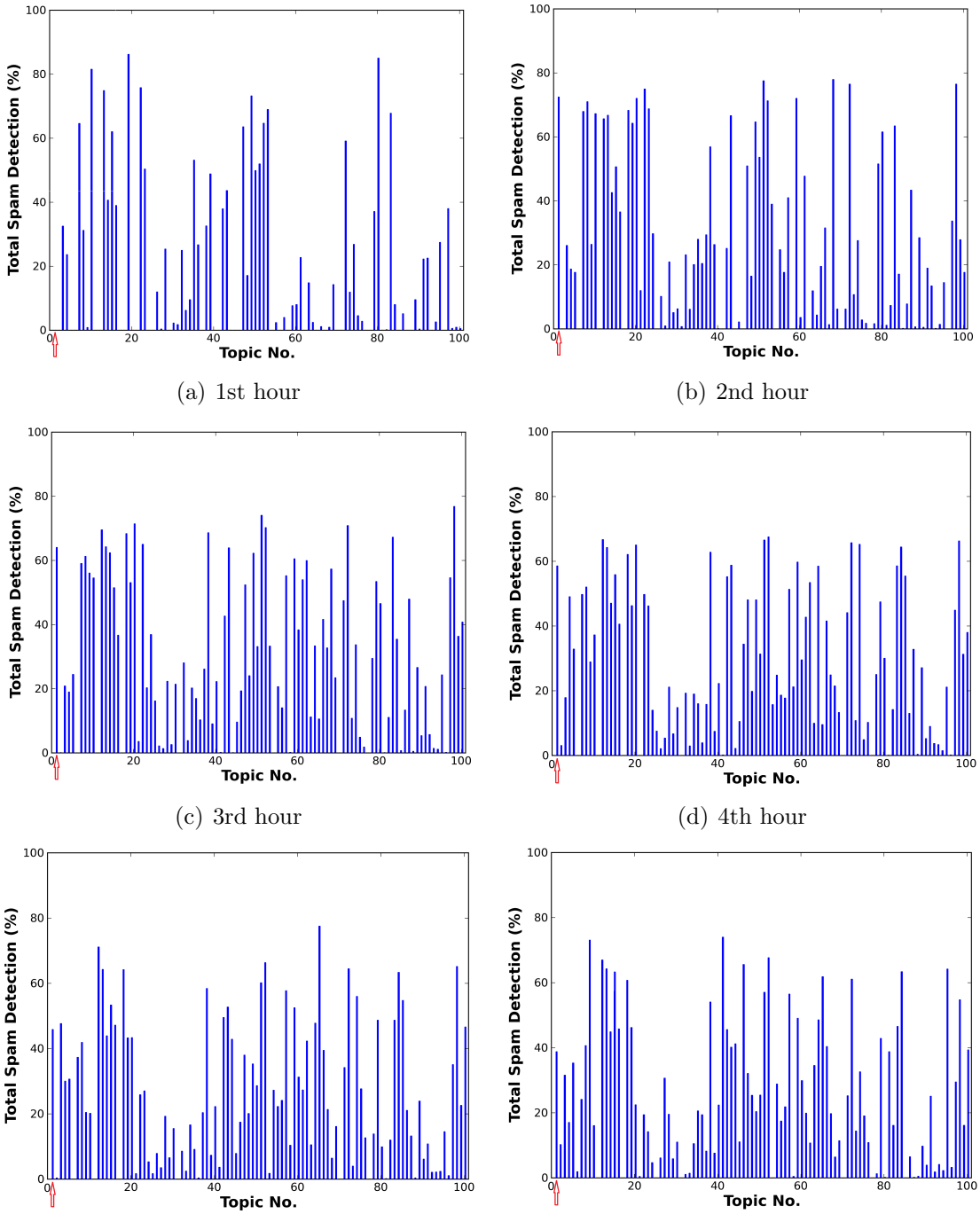
To provide a bit more insight into the performance of this countermeasure, we show in Figure 5.13 the total spam detection percent of each topic per hour. Considering the TSD of the first topic #SomeWhereInTheHood in the red arrow from 1st to 6th hour, its TSD values increase initially (from $0 \rightarrow 72$) and then progressively decrease (from $72 \rightarrow 64 \rightarrow 58 \rightarrow 45 \rightarrow 38\%$). As the deployment time is delayed, the number of spam messages and detected spam messages by the classifier decreases since most have already been inserted into the system and viewed by users.[‡]

5.3.4 Combining Countermeasures

Finally, we consider the effectiveness of combining both countermeasures (rule-based + supervised classification). Does rule-based filtering detect spam messages that a classifier would misclassify? For this combination, we first apply the rule-based filter and then apply the supervised classifier to the remaining messages.

Table 5.6 presents the evaluation result of the combined spam detection approach across 101 topics, achieving 97.63% accuracy, 0.007 FP, 0.359 FN and 54.89% TSD. We can observe that the combined approach outperformed either the rule-based

[‡]Note that the reason why TSD in the 1st hour is 0 in this case is because no spam messages have been observed yet.



(a) 1st hour

(b) 2nd hour

(c) 3rd hour

(d) 4th hour

(e) 5th hour

(f) 6th hour

Figure 5.13: Evaluating Countermeasure 2: Supervised Classification. Figures illustrate the total spam detected across all 101 topics for increasing deployment times.

Table 5.6: Combining countermeasures. We see a slight improvement in total spam detected compared to strictly applying the supervised classifier (from 50% to 55%)

Accuracy (%)	FP	FN	TSD (%)
97.63	0.007	0.359	54.89

filtering approach and the supervised classification approach. Compared to the supervised classifier alone, the combined approach results in: accuracy is increased by 0.06%, false negative rate decreases by 0.025 (lower is better), and total spam detected increases by 4.73%. The false positive rate is the same in both cases.

Finally, we evaluate this combined approach from the perspective of our users accessing collective attention information in the system. Returning to the spamness measure (again, which indicates the prevalence of spam items in the top-k results accessed by users), we evaluate the quality of the recency-based and relevance-based information access approaches both with and without the combined countermeasure.

For this experiment, we assume that a user issues a topic as a query (a hashtag in Twitter domain or a phrase) once per minute. For the recency approach, the system returns the 10-most recently posted messages. For the relevance-based approach, the system first retrieves all relevant messages posted within the past one hour and then ranks messages (by grouping popular payloads, ranking by their occurrence count, and then removing duplicates to maintain diversity). For each approach, we measure the spamness once per minute.

Figure 5.14 shows the average spamness reduction change rate for each access method (recency and relevance) by comparing the spamness without the combined countermeasure (A) to the spamness with the combined countermeasure (B). The spamness change rate is averaged across all 101 topics as $\frac{(A-B)}{A}$. In the aggregate, we find that the combined countermeasure reduces spamness by 59% for the recency-

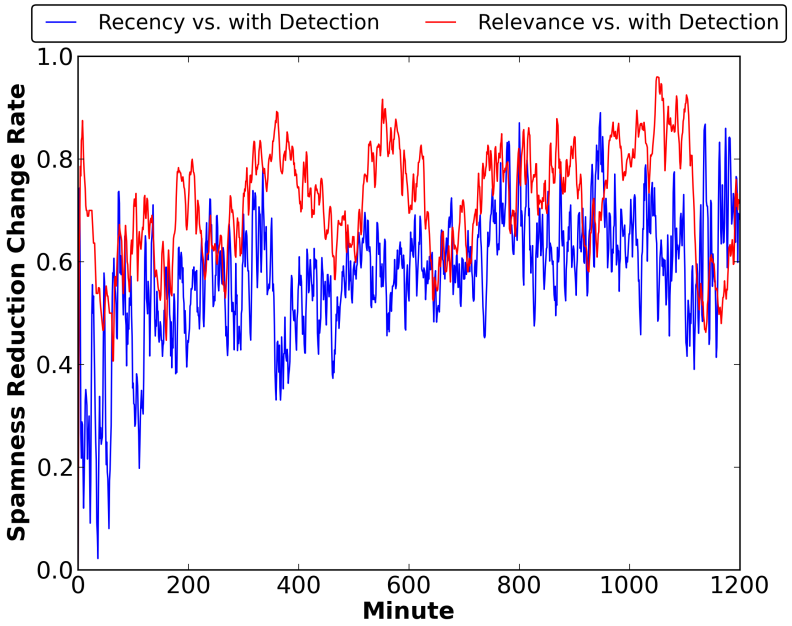


Figure 5.14: (Color) Spamness reduction change rate after applying the combined countermeasure. Spamness falls by 59% for the recency-based approach and by 73% for the relevance-based one.

based approach and by 73% for the relevance-based approach.

5.3.5 Summary and Discussion

Through our twofold approach – data-driven modeling coupled with evaluation over a system trace – we have seen that social systems are extremely susceptible to collective attention spam. With spammers accounting for only 5% of all users, we have found that every legitimate user can be exposed to spam. At even higher spammer penetration, the social system becomes unusable with spam dominating. We have also seen how this threat to collective attention can be augmented through strategically coordinated spammer behaviors to selectively push particular spam payloads, increasing the exposure of legitimate users to spam content. While daunting, we have seen preliminary evidence that carefully-crafted countermeasures may be

effective deterrents to collective attention spam – based on high accuracy (up to 98%) and total spam detected (up to 73%) with a low false positive rate (meaning few non-spam messages are incorrectly labeled). We found that it is possible to filter collective attention spam messages by learning from early-age spam messages in a topic. And since the countermeasures using rules and supervised classification are relatively lightweight, these methods can be applied for near real-time spam filtering.

An open question is how to verify that the spam messages in the first few hours used to bootstrap the learning approach are indeed spam. In our continuing work, we're considering two approaches:

- *Blacklists and URL Filtering:* One possible approach is filtering spam messages by URLs because most spam messages contain URLs. In one encouraging line, Thomas et al. [132] revealed that spammers forming major spam campaigns have reused URLs in their spam messages. In a similar direction, Grier et al. [49] estimated whether blacklists can filter spam on social networking sites and found that 8% of spam messages on these sites can be filtered by blacklists. Since new URLs are often generated and propagated through social networking sites before these URLs are included to the blacklists, Thomas et al. [131] proposed a real-time system that crawls URLs as they are submitted to web services and determines whether the URLs direct to spam. Hence, there is an opportunity to bootstrap this URL-based evidence to protect against collective attention spam.

- *Crowd Workers:* Another possible approach is using crowd workers in crowd-sourcing sites to label samples of early messages containing a popular topic. In recent years, researchers have studied how to use crowds for such large-scale distributed labeling tasks. Early promise has been reported in the information

retrieval (e.g., [1]) and database communities (e.g., [40]), signaling the potential of crowd-labels as a viable source. One possibility is to integrate a collective attention early warning monitor akin to Bernstein's work [9] on integrating crowds into a word processor.

While using crowd workers to bootstrap the detection of collective attention spam seems promising, researchers have recently identified the use of these systems to create and propagate spam campaigns. As Thomas et al. [132] found that major spam campaigns on Twitter are related with affiliates services such as Clickbank and Amazon where spammers get some money when users click URLs connecting to the affiliated service pages. Wang et al. [143] discovered that spammers have started recruiting workers in crowdsourcing sites and ask them to propagate URLs or memes like a phrase promoting a product or website. These human-powered spammers (in contrast to traditional script or bot-controlled spam accounts) are potentially quite troublesome to detect. And yet, integrating evidence from crowdsourcing sites (e.g., which accounts in a crowdsourcing site are linked to spammers in a social media site) with a collective attention spam detector could provide an additional signal for continued detection success.

5.4 Summary

In this chapter, we have presented a dual study of the robustness of social systems to collective attention threats through both a *data-driven modeling* approach and *deployment over a real system trace*. We have explored the resilience of large-scale social systems to threats to collective attention, observing that relevance-based access methods are more robust than recency-based ones and that only slight increases in the fraction of spammers in a system can fundamentally disrupt the quality of information. We have identified two countermeasures – rule-based filtering and su-

pervised classification – and demonstrated their effectiveness at filtering spam during the early development of a bursting phenomenon in a real system.

6. TRACKING AND REVEALING CROWDSOURCED MANIPULATION*

6.1 Introduction

Complementing our development and countermeasures to three threats – (i) social spam; (ii) coordinated campaigns; and (iii) collective attention spam, we now investigate one emerging source of these threats. Our goal is to uncover the ecosystem of crowdturfers.

Crowdsourcing systems have successfully leveraged the attention of millions of "crowdsourced" workers to tackle traditionally vexing problems. From specialized systems like Ushahidi (for crisis mapping), Foldit (for protein folding) and Duolingo (for translation) to general-purpose crowdsourcing platforms like Amazon Mechanical Turk and Crowdflower – these systems have shown the effectiveness of intelligently organizing large numbers of people.

However, these positive opportunities have a sinister counterpart: large-scale "crowdturfing," wherein masses of cheaply paid shills can be organized to spread malicious URLs in social media, form artificial grassroots campaigns ("astroturf"), and manipulate search engines. For example, it has been recently reported that Vietnamese propaganda officials deployed 1,000 crowdturfers to engage in online discussions and post comments supporting the Communist Party's policies [104]. Similarly, the Chinese "Internet Water Army" can be hired to post positive comments for the government or commercial products, as well as disparage rivals [125, 148]. Mass organized crowdturfers are also targeting popular services like iTunes [22] and attracting the attention of US intelligence operations [38]. And increasingly, these

*Reprinted with permission from "Crowdturfers, Campaigns, and Social Media: Tracking and Revealing Crowdsourced Manipulation of Social Media" by Kyumin Lee, Prithivi Tamilarasan, and James Caverlee, 2013. *Proceedings of the 7th International AAAI Conference on Weblogs and Social Media*, Copyright 2013 by AAAI.

campaigns are being launched from commercial crowdsourcing sites, potentially leading to the commoditization of large-scale turfing campaigns. In a recent study of the two largest Chinese crowdsourcing sites Zhubajie and Sandaha, Wang et al. [143] found that ∼90% of all tasks were for crowdturfing.

Hence, in this chapter we are interested to explore the ecosystem of crowdturfers. Who are these participants? What are their roles? And what types of campaigns are they engaged in? Unfortunately, crowdsourcing sites typically only reveal very limited information about each worker – like a username and a date joined – meaning that detailed analysis is inherently challenging. As a result, we propose to link workers to their activity in social media. By using this linkage, can we find crowd workers in social media? Can we uncover the implicit power structure of crowdturfers? Can we automatically distinguish between the behaviors of crowdturfers and regular social media users? Toward answering these questions, we make the following contributions in this chapter:

- We first analyze the types of malicious tasks and the properties of requesters and workers on Western crowdsourcing sites such as Microworkers.com, ShortTask.com and Rapidworkers.com. Previous researchers have investigated Chinese-based crowdsourcing sites; to our knowledge this is the first study to focus primarily on Western crowdsourcing sites.

- Second, we propose a framework for linking tasks (and their workers) on crowdsourcing sites to social media, by monitoring the activities of social media participants on Twitter. In this way, we can track the activities of crowdturfers in social media where their behavior, social network topology, and other cues may leak information about the underlying crowdturfing ecosystem.

- Based on this framework, we identify the hidden information propagation struc-

ture connecting these workers in social media, which can reveal the implicit power structure of crowdturfers identified on crowdsourcing sites. Specifically, we identify three classes of crowdturfers – professional workers, casual workers, and middlemen – and we demonstrate how their roles and behaviors are different in social media.

6.2 Analysis of Crowdturfing Tasks and Participants

In this section, we begin our study through an examination of the different types of crowdturfing campaigns that are posted in Western crowdsourcing sites and study the characteristics of both *requesters* (who post jobs) and *workers* (who actually perform the jobs).

We collected 505 campaigns by crawling three popular Western crowdsourcing sites that host clear examples of crowdturfing campaigns: Microworkers.com, ShortTask.com, and Rapidworkers.com during a span of two months in 2012. Almost all campaigns in these sites are crowdturfing campaigns, and these sites are active in terms of number of new campaigns. Note that even though Amazon Mechanical Turk is one of the most popular crowdsourcing sites, we excluded it in our study because it has only a small number of crowdturfing campaigns and its terms of service officially prohibits the posting of crowdturfing campaigns.[†] For the 505 sampled campaigns, each has multiple tasks, totaling 63,042 tasks.

6.2.1 Types of Crowdturfing Campaigns

Analyzing the types of crowdturfing campaigns available in crowdsourcing sites is essential to understand the tactics of the requesters. Hence, we first manually grouped the 505 campaigns into the following five categories:

[†]Perhaps surprisingly, Microworkers.com is ranked by Alexa.com at the 4,699th most popular website while Amazon Mechanical Turk is ranked 7,173.

- *Social Media Manipulation [56%]:* The most popular type of campaign targets social media. Example campaigns request workers to spread a meme through social media sites such as Twitter, click the "like" button of a specific Facebook profile/product page, bookmark a webpage on Stumbleupon, answer a question with a link on Yahoo! Answers, write a review for a product at Amazon.com, or write an article on a personal blog. An example campaign is shown in Figure 6.1, where workers are requested to post a tweet including a specific URL.

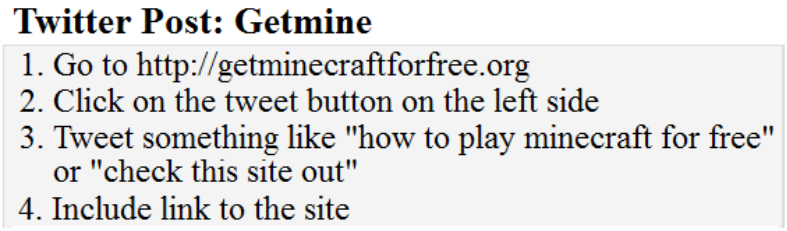


Figure 6.1: An example social media manipulation campaign

- *Sign Up [26%]:* Requesters ask workers to sign up on a website for several reasons, for example to increase the user pool, to harvest user information like name and email, and to promote advertisements.

- *Search Engine Spamming [7%]:* For this type of campaign, workers are asked to search for a certain keyword on a search engine, and then click the specified link (which is affiliated with the campaign's requester), toward increasing the rank of the page.

- *Vote Stuffing [4%]:* Requesters ask workers to cast votes. In one example, the requester asked workers to vote for "Tommy Marsh and Bad Dog" to get the

best blue band award in the Ventura County Music Awards (which the band ended up winning!).

- *Miscellany [7%]:* Finally, a number of campaigns engaged in some other activity: for example, some requested workers to download, install, and rate a particular software package; others requested workers to participate in a survey or join an online game.

We see that most crowdturfing campaigns target social media; for this reason, we will return in the following section with a framework for harvesting user activity in social media for further exploring the ecosystem of crowdturfing.

6.2.2 Requesters and Workers

We now turn to an analysis of the requesters who have posted these jobs and the workers who have actually completed them. Since this type of information is potentially quite revealing, both ShortTask.com and Rapidworkers.com do not reveal any information about their requesters and workers (aside from username). Luckily, Microworkers.com does provide payment information, country of origin, and other detailed characteristics of both requesters and workers. Hence, we collected 144 requesters' profiles and 4,012 workers' profiles from Microworkers.com – where all campaigns in our sample data are crowdturfing campaigns and other researchers have found that 89% of campaigns hosted at Microworkers.com are indeed crowdturfing [143].

6.2.2.1 Worker Characteristics

First, we analyzed the workers' profile information consisting of the country, account longevity, number of tasks done and profit (how much they have earned).

We found that the workers are from 75 countries. Figure 6.2(a) shows the top-10

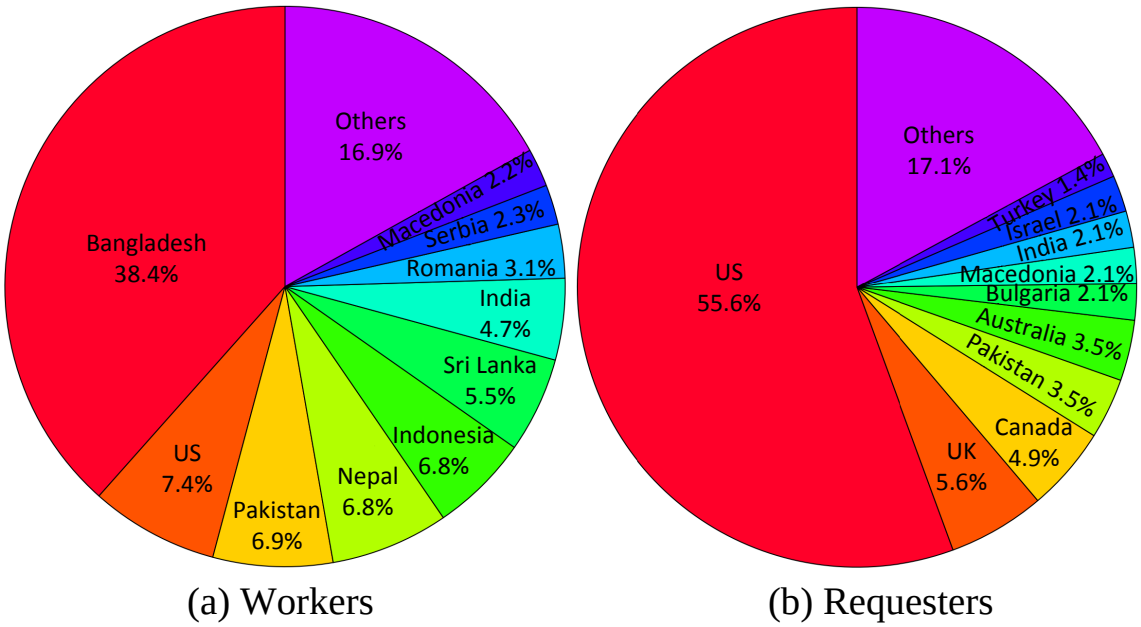


Figure 6.2: Top 10 countries of workers and requesters

Table 6.1: Characteristics of crowdturfing workers

	# of Tasks	Total Earned ($)	Longevity (day)
Max	24,016	3,699	1,215
Avg	738	117	368
Median	166	23	320
Min	10	1	5

countries which have the highest portion of workers. 83% of the workers are from these countries. An interesting observation is that a major portion of the workers in Microworkers.com are from Bangladesh – where 38% workers (1,539 workers) come from – whereas in Amazon Mechanical Turk over 90% workers are from the United States and India [119].

The 4,012 workers have completed 2,962,897 tasks and earned $467,453 so far, which suggests the entirety of the crowdturfing market is substantial. Interestingly, the average price per task is higher on a crowdturfing site (for Microworkers.com, the

average is $0.51) than on the legitimate Amazon Mechanical Turk where 90 percent of all tasks pay less than $0.10 [58].

Table 6.1 presents the maximum, average, median and minimum number of tasks done, how much they have earned, and the account longevity for the sampled workers. We observe that there are professional workers who have earned reasonable money from the site to survive. For example, a user who earned $3,699 for slightly more than 3 years (1,215 days) lives in Bangladesh where the GNI (Gross National Income) per capita is $770 in 2011 as estimated by the World Bank [134]. Surprisingly, she has earned even more money per year ($1,120) than the average income per year ($770) of a person in Bangladesh.

6.2.2.2 Requester Characteristics

Next, we examine the characteristics of those who post the crowdturfing jobs.

We found that requesters are from 31 countries. Figure 6.2(b) shows the top-10 countries which have the highest portion of requesters. Interestingly, 55% of all requesters are from the United States, and 70% of all requesters are from the English-speaking countries: United States, UK, Canada, and Australia. We can see an imbalance between the country of origin of requesters and of the workers, but that the ultimate goal is to propagate artificial content through the English-speaking web.

The requesters' profile information reveals their account longevity, number of paid tasks and expense/cost for campaigns. As shown in table 6.2, many workers have created multiple campaigns with lots of tasks (on an average – 68 campaigns and 7,030 paid tasks). The most active requester in our dataset initiated 4,137 campaigns associated with 455,994 paid tasks. In other words, he has spent a quarter million dollar ($232,557) – again a task costs $0.51 on an average. In total, 144 requesters have created 9720 campaigns with 1,012,333 tasks and have paid a half million dollars

Table 6.2: Characteristics of crowdturfing requesters

	# of campaigns	# of paid tasks	Longevity (day)
Max	4,137	455,994	1,091
Avg	68	7,030	329
Median	7	306	259
Min	1	0	3

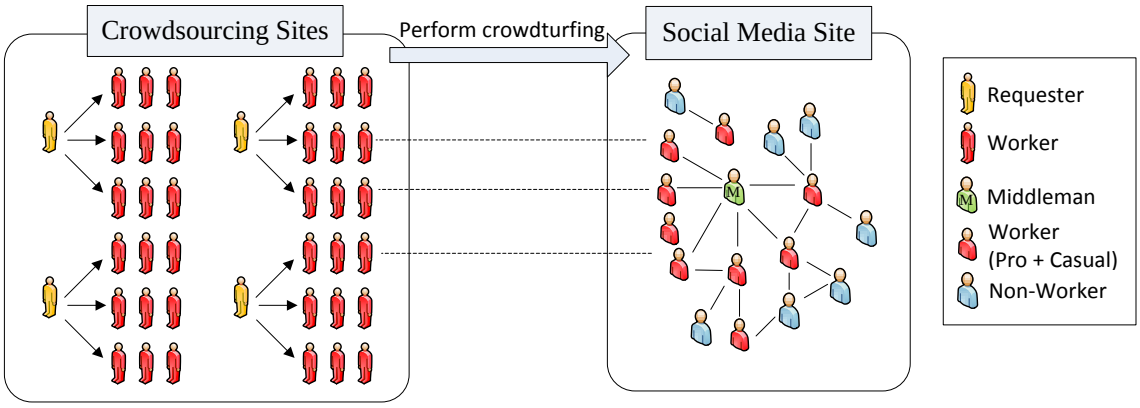


Figure 6.3: Linking crowdsourcing workers to social media

($516,289). This sample analysis shows us how the dark market is big enough to tempt users from the developing countries to become workers.

6.3 Down the Rabbit Hole: Linking Crowdsourcing Workers to Social Media

So far, we have seen that most crowdturfers target social media and that the crowdturfing economy is significant: with hundreds of thousands of tasks and dollars supporting it, based on just a fairly small sample. We now propose a framework for beginning a more in-depth study of the ecosystem of crowdturfing by linking crowdsourcing workers to social media. Specifically, we focus on Twitter-related campaigns and their workers. Of the social media targets of interest by crowdturfers, Twitter has the advantage of being open for sampling (in contrast to Facebook and

others). Our goal is to better understand the behavior of Twitter workers, how they are organized, and to find identifying characteristics so that we may potentially find workers "in the wild".

6.3.1 Following Crowd Workers onto Twitter

Based on our sample of 505 campaigns, we found that 65 specifically targeted Twitter. Of these, there were two types:

- *Tweeting about a Link:* These tasks ask the Twitter workers to post a tweet including a specific URL (as in the example in Figure 6.1). The objective is to spread a URL to other Twitter users, and thereby increase the number of clicks on the URL.

- *Following a Twitter User:* The second task type requires a Twitter worker to follow a requester's Twitter account. These campaigns can increase the visibility of the requester's account (for targeting larger future audiences) as well as impacting link analysis algorithms (like PageRank and HITS) used in Twitter search or in general Web search engines that incorporate linkage relationships in social media.

Next we identified the Twitter accounts associated with these workers (see the overall framework in Figure 6.3). For campaigns of the first type, we used the Twitter search API to find all Twitter users who had posted the URL. For campaigns of the second type, we identified all users who had followed the requester's Twitter account. In total, we identified 2,864 Twitter workers. For these workers, we additionally collected their Twitter profile information, most recent 200 tweets, and social relationships (followings and followers). The majority of the identified Twitter workers participated in multiple campaigns; we assume that the probability that

Table 6.3: Twitter dataset

Class	|User Profiles|	|Tweets|
Workers	2,864	364,581
Non-Workers	9,878	1,878,434

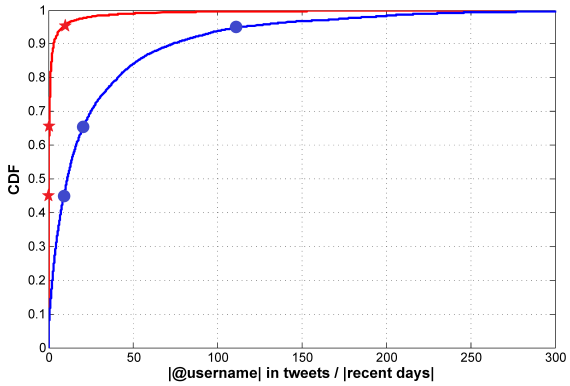
they tweeted a requester's URL or followed a requester's account by chance is very low.

In order to compare how these workers' properties are different from non-workers, we randomly sampled 10,000 Twitter users. Since we have no guarantees that these sampled users are indeed non-workers, we monitored the accounts for one month to see if they were still active and not suspended by Twitter. After one month, we found that 9,878 users were still active. In addition, we randomly selected 200 users out of the 9,878 users and manually checked their profiles, and found that only 6 out of 200 users seemed suspicious. Based on these verifications, we labeled the 9,878 users as non-workers. Even though there is a chance of a false positive in the non-worker set, the results of any analysis should give us at worst a lower bound since the introduction of possible noise would only degrade our results.

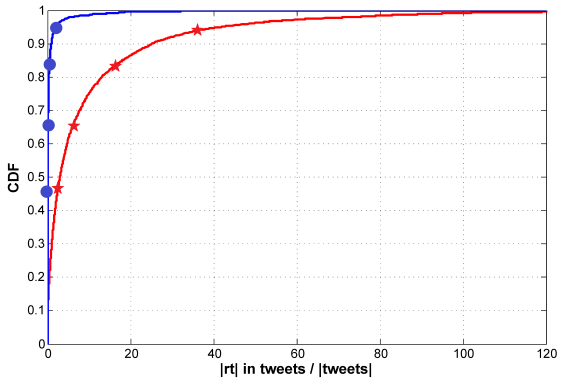
The basic property information of the workers and non-workers are shown in Table 6.3. In total, we collected 2,864 twitter workers' profile information, their 364,581 messages and their social relationships, and 9,878 twitter non-workers' profile information, their 1,878,434 messages and their social relationships.

6.3.2 Analysis of Twitter Workers: By Profile, Activity, and Linguistic Characteristics

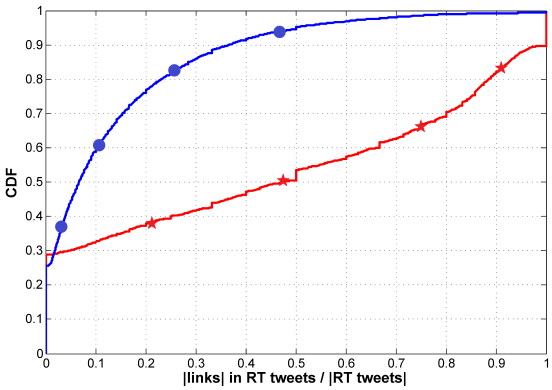
In this section we conduct a deeper analysis regarding the Twitter workers and non-workers based on their profile information, activity within Twitter, and linguistic information revealed in their tweets. Are workers on Twitter fundamentally different



(a) |@username| in tweets / |recent days|



(b) |rt| in tweets / |tweets|



(c) |links| in RT tweets / |RT tweets|

Figure 6.4: Three activity-based characteristics of workers (red line with stars) and non-workers (blue line with circles). Workers tend to mention few other users, but retweet more often, and include links more often than non-workers.

Table 6.4: Properties of workers

	|Followings|	|Followers|	|Tweets|
Min.	0	0	0
Max.	300,385	751,382	189,300
Avg.	5,519	6,649	2,667
Median	429	213	194

Table 6.5: Properties of non-workers

	|Followings|	|Followers|	|Tweets|
Min.	0	0	0
Max.	50,496	1,097,911	655,556
Avg.	511	1,000	10,128
Median	244	231	4,018

from regular users? And if so, in what ways? Note that our analysis that follows considers the entirety of the characteristics of these workers and not just the messages associated with crowdturfing campaigns.

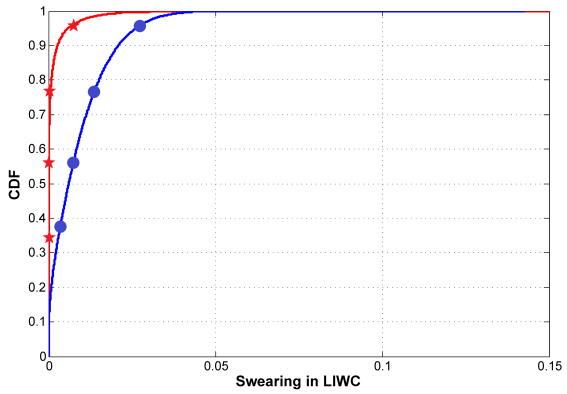
First, we compare profile information of workers and non-workers, especially focusing on the number of following, the number of followers, and the number of tweets. In Table 6.4 and 6.5, we can clearly observe that the average number of followings and followers of the workers are much larger than non-workers, but the average number of tweets of the workers is smaller than non-workers. Interestingly, workers are well connected with other users, and potentially their manipulated messages will be exposed to many users.

Next, we study how workers' activity-based characteristics differ from non-workers. We analyzed many activity-based features, including the average number of links per tweet, the average number of hashtags per tweet, and the average number of @username per tweet. In Figure 6.4, we report the cumulative distribution function for three clearly distinct activity-based characteristics: the average number of @user-

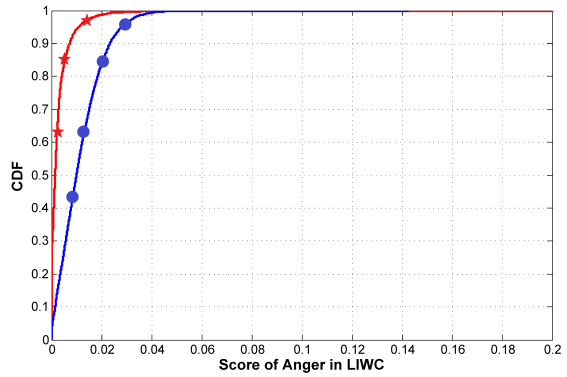
names per day during the recent days (in this case, the past month), the average number of retweet message per tweet, and the average number of links per retweet message.

We can clearly observe that workers rarely communicate with other users via @username while non-workers are often communicating with other users. This distinctive behavior makes sense because workers mostly post a message containing a meme or a URL instead of personally talking to another user. However, workers often retweet messages so that these messages may reach their followers and include links more often than non-workers.

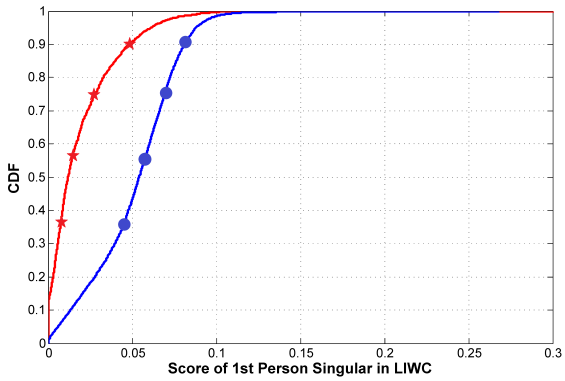
Next, we study the linguistic characteristics of the tweets posted by workers and non-workers. Do workers engage in different language use? To answer this question, we used the Linguistic Inquiry and Word Count (LIWC) dictionary, which is a standard approach for mapping text to psychologically-meaningful categories [102]. LIWC-2001 defines 68 different categories, each of which contains several dozens to hundreds of words. Given each user's tweets, we measured his linguistic characteristics in the 68 categories by computing his score of each category based on LIWC dictionary. First we counted the total number of words in his tweets (N). Next we counted the number of words in his tweets overlapped with the words in each category i on LIWC dictionary (C_i). Then, we computed his score of a category i as C_i/N. In Figure 6.5, we show the cumulative distribution functions for three of the most distinguishing linguistic characteristics: Swearing, Anger, and Use of 1st Person Singular. Interestingly, we see that workers tend to swear less, use anger less (e.g., they don't use words like "hate" or "pissed"), and use the 1st-person singular less than non-workers. That is, this linguistic analysis shows that workers are less personal in the messages they post than are non-workers. On one hand, this seems reasonable since workers intend to spread pre-defined manipulated content and URLs



(a) Swearing in LIWC



(b) Anger in LIWC



(c) 1st Person Singular

Figure 6.5: Three linguistic characteristics of workers (red line with stars) and non-workers (blue line with circles). Workers tend to swear less, use anger less, and use the 1st-person singular less than non-workers.

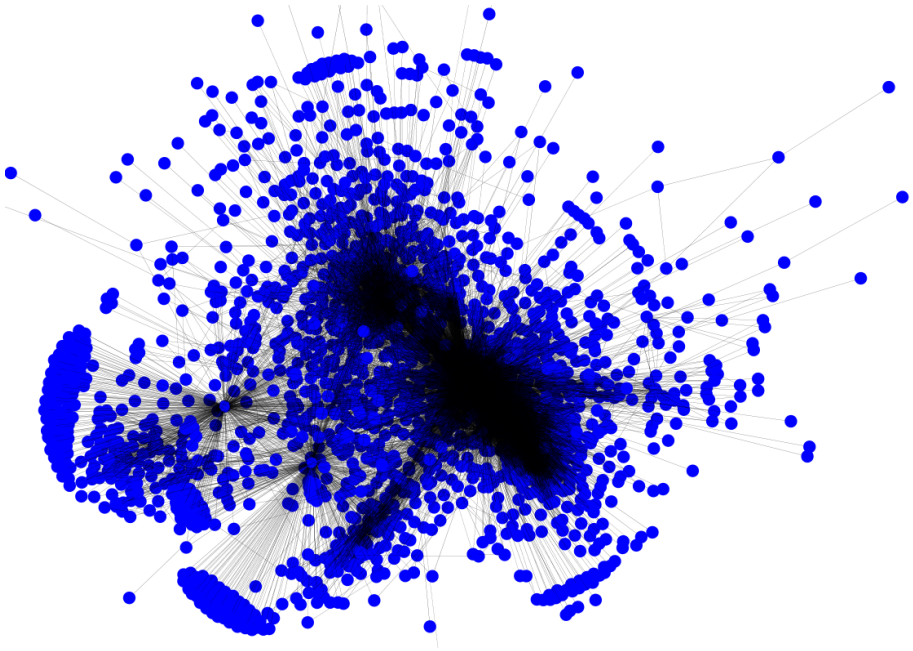


Figure 6.6: Network structure of all workers

(and hence worker tweets should not focus on themselves). However, recall that our data collection includes for each worker all of their recent tweets and not just their crowdturfing-related tweets; so this result may be surprising that the entirety of a worker's tweets show such a clear linguistic division from non-workers.

6.3.3 Network Structure of Twitter Workers

We next explore the network structure of workers by considering the social network topology of their Twitter accounts. What does this network look like? Are workers connected? More generally, can we uncover the implicit power structure of crowdturfers?

6.3.3.1 A Close-Knit Network

We first analyzed the Twitter workers' following-follower relationship to check whether they were connected to each other. Figure 6.6 depicts the induced network

structure, where a node represents a worker and an edge between two nodes represents that at least one of workers is following the other (in some cases both of them follow each other). Surprisingly, we observed that some workers are densely connected to each other, forming a closely knit network. We measured the graph density of the workers as $\frac{|E|}{|V|\times|V-1|}$ to compare whether these workers form a denser network than the average graph density of users in Twitter. Confirming what visual observation of the network indicates, we found that the workers' graph density was 0.0039 while Yang et al. [158] found the average graph density of users on Twitter to be 0.000000845, many orders of magnitude less dense.

6.3.3.2 Hubs and Authorities

We next examine who in the network is significant. Concretely, we adopted the well-known HITS [67] algorithm to identify the hubs (workers who follow many other workers) and authorities (workers who are followed by many other workers) of the network:

$$\overrightarrow{a} \leftarrow A^T \overrightarrow{h}$$
$$\overrightarrow{h} \leftarrow A \overrightarrow{a}$$

where $\overrightarrow{h}$ and $\overrightarrow{a}$ denote the vectors of all hub and all authority scores, respectively. A is a square matrix with one row and one column for each worker (user) in the worker graph. If there is an edge between worker i and worker j, the entry A_{ij} is 1 and otherwise 0. We iterate the computation of $\overrightarrow{h}$ and $\overrightarrow{a}$ until both $\overrightarrow{h}$ and $\overrightarrow{a}$ are converged. We initialized each worker's hub and authority scores as *1/n* – where n is the number of workers in the graph – and then computed HITS until the scores converged.

Table 6.6 and 6.7 present the top-10 hubs and top-10 authorities in the workers' following-follower graph. Interestingly, most top-10 hubs are top-10 authorities. It means that these workers are very well connected with other workers, following them and followed by them. The top hub and top authority is *NannyDotNet*, a user who has been both a requester of crowdturfing jobs and a worker on the jobs of others. The other nine workers have a large number of followings and followers. This behavior is similar with "social capitalists", who are eager to follow other users and increase a number of followers as noted in [45]. Even *_Woman_health*'s profile description shows "Always follow back within 24 hours", indicating her intention increasing a number of followers. Interestingly, their Twitter profiles are fully filled, sharing what they are working for or why they are using Twitter, location information and a profile photo.

Table 6.6: Top-10 hubs of the workers

Screen Name	|Followings|	|Followers|	|Tweets|
NannyDotNet	1,311	753	332
_Woman_health	210,465	207,589	33,976
Jet739	290,624	290,001	22,079
CollChris	300,385	300,656	8,867
familyfocusblog	40,254	39,810	22,094
tinastullracing	171,813	184,039	73,004
drhenslin	98,388	100,547	10,528
moneyartist	257,773	264,724	1,689
pragmaticmom	30,832	41,418	21,843
Dede_Watson	37,397	36,833	47,105

6.3.3.3 Professional Workers

In our examination of workers, we noticed that some workers engaged in many jobs, while others only participated in one or two. We call these workers who oc-

Table 6.7: Top-10 authorities of the workers

Screen Name	|Followings|	|Followers|	|Tweets|
NannyDotNet	1,311	753	332
_Woman_health	210,465	207,589	33,976
CollChris	300,385	300,656	8,867
familyfocusblog	40,254	39,810	22,094
tinastullracing	171,813	184,039	73,004
pragmaticmom	30,832	41,418	21,843
Jet739	290,624	290,001	22,079
moneyartist	257,773	264,724	1,689
drhenslin	98,388	100,547	10,528
ceebee308	283,301	296,857	169,061

casionally participate "casual workers", while we refer to workers in three or more campaigns as "professional workers". Since these professional workers often worked for multiple campaigns, understanding their behaviors is important to discern the characteristics of the quasi-permanent crowdturfing workforce.

Of the 2,864 workers in total, there were 187 professional workers who participated in at least 3 Twitter campaigns in our collection. Figure 6.7 depicts their network structure. We can clearly observe that these professional workers are also densely connected. Surprisingly, their graph density is 0.028 which is even higher than all workers' graph density (0.0039).

So far, we only looked at the relationship between these professional workers in their following-follower relationship (i.e., the restricted graph). Next we extend the following-follower relationship to all users (i.e., the open graph including all followings and followers of these professional workers). Table 6.8 and 6.9 present top-10 followings and followers of these professional workers, respectively.

One observation from Table 6.8 is that these professional workers commonly retweeted messages generated by the two users named *Alexambroz* and *0boy*. We conjecture that these users are *middlemen* who create messages to promote a web-

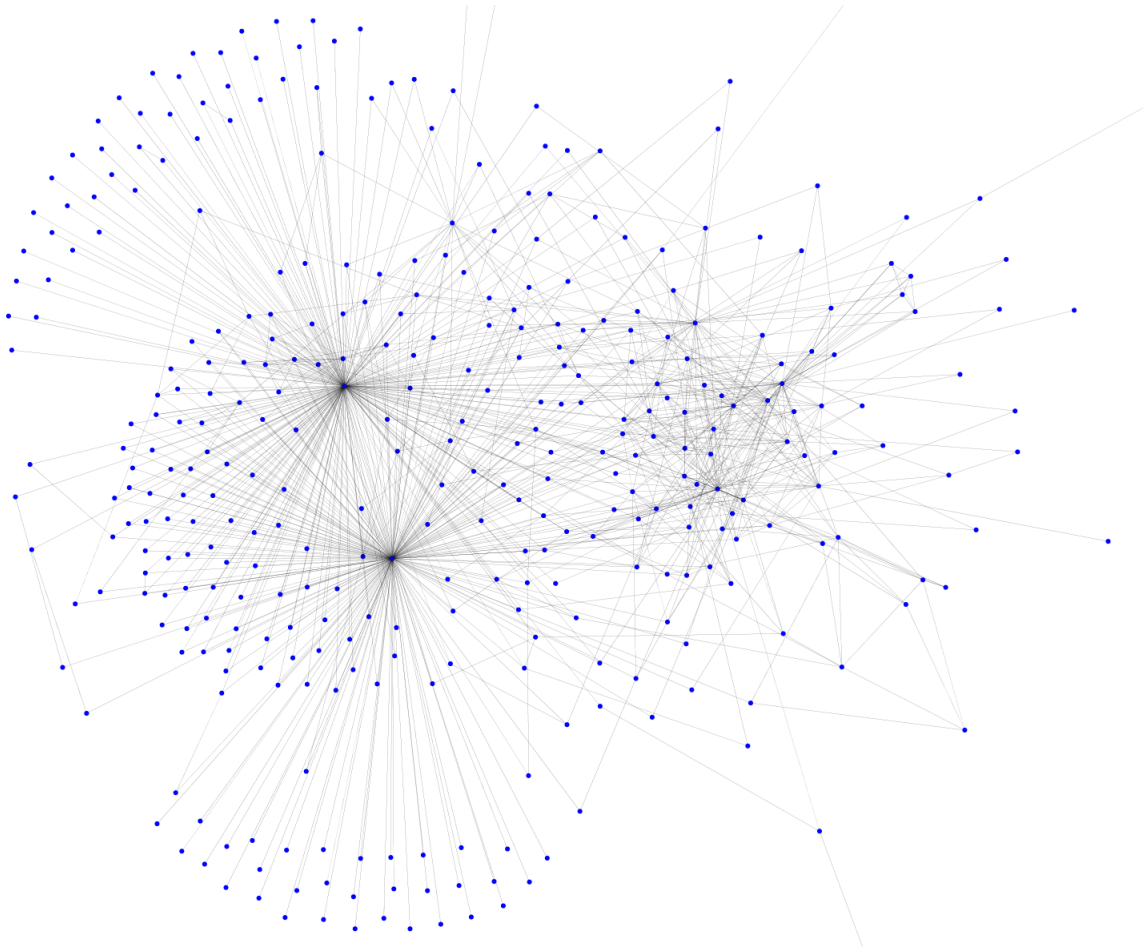


Figure 6.7: Network structure of professional workers

site or a meme. Since the professional workers follow the middlemen, they will receive the messages automatically (if a user A follows a user B, A will receive B's postings/messages automatically in Twitter) and retweet them to their followers so that the messages are exposed to these workers' followers. The middlemen and professional workers strategically use Twitter as a tool to effectively propagate targeted messages. In another perspective, since these professional workers follow the middlemen and retweet the middlemen's messages, the middlemen get higher rank in a link analysis method such as PageRank [14] and HITS [67]. As the result, the mid-

Table 6.8: Top-10 followings of the professional workers

Screen Name	Fre.	|Followings|	|Followers|	|Tweets|
Alexambroz	52	53	51,463	307
Talenthouse	51	14,369	99,880	13,356
Thestoryvine	47	230	127	97
Nettime	42	307	401	1,305
0boy	41	847	108,929	10,827
TheRealAliLee	38	323	9,060	1,509
consumeraware	37	10	845	60
WebsiteBuilderr	36	509	235	81
Ijsfondue	35	100	900	98
ProveeduriaT	33	87	171	312

Table 6.9: Top-10 followers of the professional workers

Screen Name	Fre.	|Followings|	|Followers|	|Tweets|
TrueTobacco	29	1,893	866	150
Honest_Solution	28	9,759	14,620	440
Choroibati	27	1,676	567	77
Mostafizurrr	26	34,229	36,809	1,612
YourSjop	24	3,610	3,236	6
SunnieBrees	23	89	56	7
TeamHustleBunny	21	88,331	99,038	9,129
Tarek0593	21	1,055	546	2,302
TinyGems	21	112,417	102,181	8,704
Checkdent	20	2,923	4,002	334

dlemen's profiles and messages will be ranked in the top position in a search result returned by a search engine like Google. These middlemen and professional workers are difficult to detect as evidences by the long lifespan of their accounts, compared with traditional Twitter spammers. For example, while middlemen's average lifespan and professional workers' average lifespan in our dataset are 1,067 days and 614 days, respectively (which are similar or even longer than regular users), twitter spammers' average lifespan is 279 days [77].

Table 6.9 shows the top-10 followers of the professional workers. *Honest_Solution*,

Choroibati, *Mostafizurrr* and *Tarek0593* followed many professional workers and they are also professional workers in our dataset, demonstrating that these professional workers are connected to each other. Why? We conjecture that they can increase their number of followers and pretend to be legitimate users. Another reason is that some crowdturfing tasks require a minimum number of followers to become eligible workers for certain tasks (e.g., at least larger than 100 followers) because these requesters want their URLs to reach more users. In addition, these professional workers are followed by some business accounts and random user accounts who will be potential victims.

6.3.4 Digging Deeper: Middlemen

We have seen that workers (and especially, professional workers) often retweet middlemen's messages so that more people including the followers of the workers were exposed to the messages. This observation naturally led us to study how to reveal middlemen. First, we investigated the messages of 187 professional workers and extracted retweeted messages containing a URL because the intention of middlemen and professional workers for spreading messages is not only to share the message, but also to tempt the message recipient (e.g., a follower of a professional worker) to visit a web page of the URL. Second, we counted how many professional workers retweeted each one of the extracted retweeted messages. Third, we sorted the extracted retweeted messages by descending order of the number of frequencies. Then, extracted an origin, who is a user (a potential middleman) creating and posting the original message of a retweeted message, from each retweeted message. Our hunch is that the more professional workers retweeted an origin's message, the higher probability to become a middleman the origin has because these professional workers make profit by posting or retweeting an astroturfing (artificial) meme or

Table 6.10: Top-10 middlemen with the number of professional workers

Middleman	|Pro-Workers|	|Followings|	|Followers|
0boy	139	847	108,929
louiebaur	95	285	68,772
hasai	63	6,360	41,587
soshable	57	956	22,676
virtualmember	56	5,618	5,625
scarlettmadi	55	5,344	26,439
SocialPros	54	10,775	22,985
cqlivingston	54	6,377	28,556
huntergreene	49	27,390	25,207
TKCarsitesInc	48	1,015	18,661

message, so if many professional workers retweeted an origin's message, he will be a middleman. By using our approach, we found 575 potential middlemen, and one or two professional workers retweeted messages of 486 out of 575 potential middlemen. Because sometimes a few professional workers retweet the same user's message by chance, we considered the potential middlemen, whose messages are retweeted by at least 10 professional workers, the middlemen. Then there were 41 middlemen. Table 6.10 shows the top-10 middlemen whose messages are retweeted the most by the professional workers.

Interestingly, the top-10 middlemen have a large number of followers (5,625 $\sim$ 108,929), and most of the middlemen disclosed they are interested in social media strategy, social marketing and SEO on their profiles. *hasai* and *SocialPros* have the same homepage on their profiles which is http://hasai.com/ advertising social media marketing. Several middlemen opened their location as Orange County, CA. Some of these middlemen also often retweeted other middlemen's messages. These observations led us to conclude that some of these middlemen accounts are connected or controlled by the same people/organization.

Next, we measured which messages are most retweeted by professional workers,

and 10 most retweeted messages are shown in Table 6.11. Nine messages except the first one have promoting and advertising flavors. We conjecture that sometimes middlemen post regular messages like the first message in the table and professional workers retweet them so that they can pretend a regular user, avoid spam detection from Twitter Safety team and be alive longer on Twitter.

Table 6.11: 10 most retweeted messages by professional workers

Freq.	Message
29	RT @alexambroz: RT @Twitter for our beautiful planet #earth with #peace and #happiness to all people. One retweet could change Our world.
23	RT @viawomen: Check out the great pregnancy info on http://t.co/5NiVbh6v. Love the celeb parenting blog posts! #pregnancy #pregnancysy ...
22	RT @BidsanityDeals: Bid now! Auctions are ending. Get DVDs, gift cards, jewelry, IPad accessories, books, handmade goods, and much more! ...
20	RT @ik8sqi: Family Tracker free iPhone app lets you track friends and family, even shows past GPS locations as breadcrumbs on map http:/ ...
17	RT @0boy: Here's an interesting marketing idea with lots of #flavor http://t.co/EPl24WZ2 #BucaVIP
17	RT @JeremyReis: 7 Reasons to Sponsor a Child - http://t.co/weg0Tq0y #childsponsorship @food4thehungry
17	RT @louiebaur: StumbleUpon Paid Discovery Is Getting Massive http://t.co/OvYJv2ne via @0boy
16	RT @evaporizing: #ECigarette Save EXTRA @v2cigs today http://t.co/BNbJl1cX use V2 Cigs Coupon EVAPE15 or EVAPE10 plus 20% Memorial Day S ...
16	RT @evaporizing: The Best #ECIGARETTE Deal Of The YEAR Today Only @V2Cigs 4th July Sale + Coupon EVAPE15 40% OFF http://t.co/yrrhTYDy
15	RT @DoYouNeedaJob: Internet Millionaire Looking For Students! Give me 30 days & I will mold you into my next success story!...Visit http ...

In summary, by finding professional workers we can potentially find middlemen, and by finding the most retweeted messages from professional workers we can potentially find hidden workers who retweeted the messages many times.

6.4 Summary

In this chapter, we have presented a framework for "pulling back the curtain" on crowdturfers to reveal their underlying ecosystem. We have analyzed the types of malicious tasks and the properties of requesters and workers on Western crowdsourcing sites. By linking tasks and their workers on crowdsourcing site to social media, we have traced the activities of crowdturfers in social media and have identified three classes of crowdturfers – professional workers, casual workers, and middlemen – and their relationship structure connecting these workers in social media. We have revealed that these workers' profile, activity and linguistic characters are different from regular social media users.

7. CONCLUSIONS AND FUTURE WORK

In this chapter, we present the conclusion of this dissertation and future research opportunities.

7.1 Conclusion

In the recent years, large-scale social systems including Web-based social networks, online social media sites and Web-scale crowdsourcing systems have continued to grow and to become popular with the number of users who generate, share and consume information. But the openness and reliance of users caused malicious participants can easily participate in these systems and threaten information quality in these systems.

In this dissertation, we have developed algorithms and architectures to improve information quality for the reliable and secure use of these systems. We identified three classes of threats to these systems: (i) content pollution by social spammers, (ii) coordinated campaigns for strategic manipulation, and (iii) threats to collective attention. To combat these threats, we propose four inter-related methods for detecting evidence of these threats, mitigating their impact, and improving the quality of information in social systems. We augment this three-fold defense with an exploration of their origins in "crowdturfing" – a sinister counterpart to the enormous positive opportunities of crowdsourcing. In particular, this dissertation research made four unique contributions:

First, we have presented the design and real-world evaluation of a novel social honeypot-based approach to social spam detection. Our overall research goal is to investigate techniques and develop effective tools for automatically detecting and filtering spammers who target social systems. By focusing on two different commu-

nities, we have seen how the general principles of (i) social honeypot deployment, (ii) robust spam profile generation, and (iii) adaptive and ongoing spam detection can effectively harvest spam profiles and support the automatic generation of spam signatures for detecting new and unknown spam. Our empirical evaluation over both MySpace and Twitter has demonstrated the effectiveness and adaptability of the honeypot-based approach to social spam detection. In addition, we have conducted seven-month long study to automatically detect and profile content polluters on Twitter with the extension of the social honeypot approach. During the study we were able to lure approximately 36,000 abusive Twitter accounts into following our collection of social honeypots. We have seen how these content polluters reveal key distinguishing characteristics in their behavior, leading to the development of robust classifiers.

Second, we have investigated the problem of campaign detection in social media. We have proposed and evaluated an efficient content-driven graph-based framework for identifying and extracting campaigns from the massive scale of real-time social systems. We have found six campaign types (spam, promotion, template, celebrity, news and babble), and analyzed temporal behaviors of various campaign types.

Third, we have presented a dual study of the robustness of social systems to collective attention threats through both a *data-driven modeling* approach and *deployment over a real system trace*. We have explored the resilience of large-scale social systems to threats to collective attention, observing that relevance-based access methods are more robust than recency-based ones and that only slight increases in the fraction of spammers in a system can fundamentally disrupt the quality of information. We have identified two countermeasures – rule-based filtering and supervised classification – and demonstrated their effectiveness at filtering spam during the early development of a bursting phenomenon in a real system.

Finally, we have presented a framework for "pulling back the curtain" on crowdturfers to reveal their underlying ecosystem. We have analyzed the types of malicious tasks and the properties of requesters and workers on Western crowdsourcing sites. By linking tasks and their workers on crowdsourcing site to social media, we have traced the activities of crowdturfers in social media and have identified three classes of crowdturfers – professional workers, casual workers, and middlemen – and their relationship structure connecting these workers in social media. We have revealed that these workers' profile, activity and linguistic characters are different from regular social media users.

7.2 Future Research Opportunities

We have two future research plans:

- *New Types of Collective Attention Threats:* Recently, location based social networks (e.g., Foursquare and Google Latitude) have grown rapidly in terms of the number of users and check-ins, and have become very popular. They allow users to share their locations with friends, and post a comment on a venue. Foursquare in January 2013 has reached over 30 millions users who have created over 3 billion check-ins [39]. But, with the growth and popularity of location based social networks, there is an emerging threat to these networks – geo-spatial collective attention spam. In Chapter 5, we have observed that collective attention spam relies on users themselves to seek out the content where the spam will be encountered. In the direction of complementing our work on collective attention spam, we are interested in studying an investigation of alternative types of collective attention threats, including manipulations of the geo-spatial footprint and temporal dynamics of collective attention (e.g., to selectively "push" certain topics in particular locations or at specific times).

- *Adversarial Propaganda:* These days social systems have become one of the main targets for online marketers and government workers to propagate misleading information or persuade users regarding a product or a government. For example, people in an anti-US government may spread rumors and misinformation to US residents, some of whom may eventually believe that the misinformation is true. Understanding underlying ecosystem of adversarial propaganda and identifying it in these social systems has become important. In our previous work investigated in Chapter 4, we have found inorganic campaigns (e.g., spam and promotion). We are interested in studying an investigation of adversarial propaganda, extending our work in campaign detection to consider models and methods of mass interpersonal persuasion.

REFERENCES

[1] Omar Alonso, Daniel E. Rose, and Benjamin Stewart. Crowdsourcing for relevance evaluation. *SIGIR Forum*, 42(2):9–15, November 2008.

[2] E. Amitay, D. Carmel, A. Darlow, R. Lempel, and A. So. The connectivity sonar: Detecting site functionality by structural patterns. In *Proceedings of the 14th ACM Conference on Hypertext and Hypermedia*, pages 38–47. ACM, 2003.

[3] I. Androutsopoulos, J. Koutsias, K. V. Chandrinos, and C. D. Spyropoulos. An experimental comparison of naive bayesian and keyword-based anti-spam filtering with personal e-mail messages. In *Proceedings of the 23rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 160–167. ACM, 2000.

[4] Apache. Hadoop. `http://hadoop.apache.org/`, 2012.

[5] Luca Becchetti, Carlos Castillo, Debora Donato, Stefano Leonardi, and Ricardo Baeza-Yates. Link-based characterization and detection of web spam. In *Proceedings of the 2nd International Workshop on Adversarial Information Retrieval on the Web*. ACM, 2006.

[6] A. Benczur, K. Csalogany, T. Sarlos, and M. Uher. SpamRank - fully automatic link spam detection. In *Proceedings of the 1st International Workshop on Adversarial Information Retrieval on the Web*. ACM, 2005.

[7] Andra's A. Benczu'r, Ka'roly Csaloga'ny, and Tama's Sarlo's. Link-based similarity search to fight web spam. In *Proceedings of the 2nd International Workshop on Adversarial Information Retrieval on the Web*. ACM, 2006.

[8] Fabrício Benevenuto, Tiago Rodrigues, Virgílio Almeida, Jussara Almeida, and Marcos Gonçalves. Detecting spammers and content promoters in online video social networks. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 620–627. ACM, 2009.

[9] Michael S. Bernstein, Greg Little, Robert C. Miller, Björn Hartmann, Mark S. Ackerman, David R. Karger, David Crowell, and Katrina Panovich. Soylent: a word processor with a crowd inside. In *Proceedings of the 23rd Annual ACM Symposium on User Interface Software and Technology*, pages 313–322. ACM, 2010.

[10] Danah Boyd and Jeffrey Heer. Profiles as conversation: Networked identity performance on friendster. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences*. IEEE Computer Society, 2006.

[11] Daren C. Brabham. Crowdsourcing as a model for problem solving: An introduction and cases. *Convergence: The International Journal of Research into New Media Technologies*, 14(1):75–90, 2008.

[12] Andrej Bratko, Bogdan Filipič, Gordon V. Cormack, Thomas R. Lynam, and Blaž Zupan. Spam filtering using statistical data compression models. *J. Mach. Learn. Res.*, 7:2673–2698, 2006.

[13] Leo Breiman. Bagging predictors. *Mach. Learn.*, 24(2):123–140, 1996.

[14] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the 7th International Conference on World Wide Web*, pages 107–117. Elsevier Science Publishers B. V., 1998.

[15] Andrei Z. Broder, Steven C. Glassman, Mark S. Manasse, and Geoffrey Zweig. Syntactic clustering of the web. *Comput. Netw. ISDN Syst.*, 29(8-13):1157–1166, 1997.

[16] Garrett Brown, Travis Howe, Micheal Ihbe, Atul Prakash, and Kevin Borders. Social networks and context-aware spam. In *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work*, pages 403–412. ACM, 2008.

[17] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. Information credibility on twitter. In *Proceedings of the 20th International Conference on World Wide Web*, pages 675–684. ACM, 2011.

[18] James Caverlee and Ling Liu. Countering web spam with credibility-based link analysis. In *Proceedings of the 26th Annual ACM Symposium on Principles of Distributed Computing*, pages 157–166. ACM, 2007.

[19] James Caverlee, Ling Liu, and Steve Webb. Socialtrust: tamper-resilient trust establishment in online communities. In *Proceedings of the 8th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 104–114. ACM, 2008.

[20] James Caverlee and Steve Webb. A large-scale study of myspace: Observations and implications for online social networks. In *Proceedings of the 2nd International Conference on Weblogs and Social Media*. AAAI, 2008.

[21] CCTV. Uncovering online promotion. `http://news.cntv.cn/china/20101107/102619.shtml`, November 2010.

[22] Casey Chan. How a fake erotic fiction ebook hit the top 5 of itunes. `http://gizmodo.com/5933169/how-a-fake-crowdsourced-erotic-ebook-hit-the-top-5-of-itunes`, August 2012.

[23] Zhicong Cheng, Bin Gao, Congkai Sun, Yanbing Jiang, and Tie-Yan Liu. Let web spammers expose themselves. In *Proceedings of the 4th ACM International Conference on Web Search and Data Mining*, pages 525–534. ACM, 2011.

[24] Zhiyuan Cheng, James Caverlee, and Kyumin Lee. You are where you tweet: a content-based approach to geo-locating twitter users. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, pages 759–768. ACM, 2010.

[25] Abdur Chowdhury, Ophir Frieder, David Grossman, and Mary Catherine McCabe. Collection statistics for fast duplicate document detection. *ACM Trans. Inf. Syst.*, 20(2):171–191, 2002.

[26] W. W. Cohen. Learning rules that classify e-mail. In *Proceedings of 1996 AAAI Spring Symposium on Machine Learning and Information Access*, pages 18–25. AAAI Press, 1996.

[27] Gordon V. Cormack. Email spam filtering: A systematic review. *Found. Trends Inf. Retr.*, 1(4):335–455, 2008.

[28] Microsoft Corporation. Exchange intelligent message filter. `http://technet.microsoft.com/en-us/exchange/bb288484.aspx`, 2003.

[29] Nilesh Dalvi, Pedro Domingos, Mausam, Sumit Sanghai, and Deepak Verma. Adversarial classification. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 99–108. ACM, 2004.

[30] Anirban Dasgupta, Kunal Punera, Justin M. Rao, and Xuanhui Wang. Impact of spam exposure on user engagement. In *Proceedings of the 21st USENIX Conference on Security Symposium*. USENIX Association, 2012.

[31] Brian D. Davison. Recognizing nepotistic links on the Web. In *Proceedings of the AAAI-2000 Workshop on AI for Web Search*, pages 23–28. AAAI, 2000.

[32] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. In *Proceedings of the 6th Symposium on Operating System Design and Implementation*, pages 137–150. USENIX Association, 2004.

[33] A.P. Dempster, N.M. Laird, D.B. Rubin, et al. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[34] Isabel Drost and Tobias Scheffer. Thwarting the nigritude ultramarine: Learning to identify link spam. In *Proceedings of the 16th European Conference on Machine Learning*, pages 96–107. Springer-Verlag, 2005.

[35] Facebook. Key facts. `http://newsroom.fb.com/Key-Facts`, December 2012.

[36] Adrienne Felt and David Evans. Privacy protection for social networking platforms. In *Web 2.0 Security and Privacy 2008 in Conjunction with 2008 IEEE Symposium on Security and Privacy*, 2008.

[37] Dennis Fetterly, Mark Manasse, and Marc Najork. Spam, damn spam, and statistics. In *Proceedings of the 7th International Workshop on the Web and Databases*, pages 1–6, 2004.

[38] Nick Fielding and Ian Cobain. Revealed: US spy operation that manipulates social media. `http://www.guardian.co.uk/technology/2011/mar/17/us-spy-operation-social-networks`, March 2011.

[39] Foursquare. About foursquare. `https://foursquare.com/about/`, January 2013.

[40] Michael J. Franklin, Donald Kossmann, Tim Kraska, Sukriti Ramesh, and Reynold Xin. Crowddb: answering queries with crowdsourcing. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*, pages 61–72. ACM, 2011.

[41] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, 1997.

[42] Liz Gannes. Estimate: 20% of web videos are spam. `http://gigaom.com/2009/06/10/estimate-20-of-web-videos-are-spam/`, June 2009.

[43] Hongyu Gao, Yan Chen, Kathy Lee, Diana Palsetia, and Alok Choudhary. Towards online spam filtering in social networks. In *Proceedings of the 19th Network & Distributed System Security Symposium*. The Internet Society, 2012.

[44] Hongyu Gao, Jun Hu, Christo Wilson, Zhichun Li, Yan Chen, and Ben Y. Zhao. Detecting and characterizing social spam campaigns. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, pages 35–47. ACM, 2010.

[45] Saptarshi Ghosh, Bimal Viswanath, Farshad Kooti, Naveen Kumar Sharma, Korlam Gautam, Fabrício Benevenuto, Niloy Ganguly, and Krishna P. Gummadi. Understanding and combating link farming in the twitter social network. In *Proceedings of the 21st International Conference on World Wide Web*, pages 61–70. ACM, 2012.

[46] David Gibson, Ravi Kumar, and Andrew Tomkins. Discovering large dense subgraphs in massive graphs. In *Proceedings of the 31st International Conference on Very Large Data Bases*, pages 721–732. VLDB Endowment, 2005.

[47] Michaela Goetz, Jure Leskovec, Mary McGlohon, and Christos Faloutsos. Modeling blog dynamics. In *Proceedings of the 3rd International Conference on Weblogs and Social Media*. AAAI, 2009.

[48] B. Gonçalves, M. Conover, and F. Menczer. Abuse of social media and political manipulation. In Markus Jakobsson, editor, *The Death of The Internet*. Wiley, 2012.

[49] Chris Grier, Kurt Thomas, Vern Paxson, and Michael Zhang. @spam: the underground on 140 characters or less. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, pages 27–37. ACM, 2010.

[50] Zoltán Gyöngyi, Hector Garcia-Molina, and Jan Pedersen. Combating web spam with trustrank. In *Proceedings of the 30th International Conference on Very Large Data Bases*, pages 576–587. VLDB Endowment, 2004.

[51] Zolt´n Gyöngyi, Pavel Berkhin, Hector Garcia-Molina, and Jan Pedersen. Link spam detection based on mass estimation. In *Proceedings of the 32nd International Conference on Very Large Data Bases*, pages 439–450. VLDB Endowment, 2006.

[52] E. Harris. The next step in the spam control war: greylisting. `http://projects.puremagic.com/greylisting/`, 2003.

[53] Paul Heymann, Georgia Koutrika, and Hector Garcia-Molina. Fighting spam on social web sites: A survey of approaches and future challenges. *IEEE Internet Computing*, 11(6):36–45, 2007.

[54] Liangjie Hong, Ovidiu Dan, and Brian D. Davison. Predicting popular messages in twitter. In *Proceedings of the 20th International Conference Companion on World Wide Web*, pages 57–58. ACM, 2011.

[55] Haiyan Hu, Xifeng Yan, Yu Huang, Jiawei Han, and Xianghong J. Zhou. Mining coherent dense subgraphs across massive biological networks for functional discovery. *Bioinformatics*, 21(1):213–221, 2005.

[56] Markus Huber, Martin Mulazzani, Edgar Weippl, Gerhard Kitzler, and Sigrun Goluch. Exploiting social networking sites for spam. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, pages 693–695. ACM, 2010.

[57] Neil J. Hurley, Michael P. O'Mahony, and Guenole C. M. Silvestre. Attacking Recommender Systems: A Cost-Benefit Analysis. *IEEE Intelligent Systems*, 22(3):64–68, 2007.

[58] Panagiotis G. Ipeirotis. Analyzing the amazon mechanical turk marketplace. *XRDS*, 17(2):16–21, December 2010.

[59] D. Irani, S. Webb, C. Pu, and K. Li. Study of trend-stuffing on twitter through text classification. In *Proceedings of the 7th Annual Collaboration, Electronic Messaging, Anti-Abuse and Spam Conference*, 2010.

[60] Danesh Irani, Marco Balduzzi, Davide Balzarotti, Engin Kirda, and Calton Pu. Reverse social engineering attacks in online social networks. In *Proceedings of the 8th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 55–74. Springer-Verlag, 2011.

[61] Jelena Isacenkova and Davide Balzarotti. Measurement and evaluation of a real world deployment of a challenge-response spam filter. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement*, pages 413–426. ACM, 2011.

[62] M. Iwanaga, T. Tabata, and K. Sakurai. Evaluation of anti-spam methods combining bayesian filtering and strong challenge and response. In *Proceedings of IASTED International Conference on Communication, Network, and Information Security*, pages 214–219. IASTED, 2003.

[63] Tom N. Jagatic, Nathaniel A. Johnson, Markus Jakobsson, and Filippo Menczer. Social phishing. *Commun. ACM*, 50(10):94–100, 2007.

[64] Kalervo Järvelin and Jaana Kekäläinen. Ir evaluation methods for retrieving highly relevant documents. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 41–48. ACM, 2000.

[65] Chris Kanich, Nicholas Weavery, Damon McCoy, Tristan Halvorson, Christian Kreibichy, Kirill Levchenko, Vern Paxson, Geoffrey M. Voelker, and Stefan Savage. Show me the money: characterizing spam-advertised revenue. In *Proceedings of the 20th USENIX Conference on Security*. USENIX Association, 2011.

[66] Aniket Kittur, Ed H. Chi, and Bongwon Suh. Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 453–456. ACM, 2008.

[67] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, September 1999.

[68] Georgia Koutrika, Frans Adjie Effendi, Zolt´n Gyöngyi, Paul Heymann, and Hector Garcia-Molina. Combating spam in tagging systems: An evaluation. *ACM Trans. Web*, 2(4):1–34, 2008.

[69] R. E. Kraut, S. Sunder, J. Morris, R. Telang, D. Filer, and M. Cronin. Markets for attention: will postage for email help? In *Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work*, pages 206–215. ACM, 2002.

[70] Christian Kreibich and Jon Crowcroft. Honeycomb: creating intrusion detection signatures using honeypots. *SIGCOMM Comput. Commun. Rev.*, 34(1):51–56, 2004.

[71] Balachander Krishnamurthy, Phillipa Gill, and Martin Arlitt. A few chirps about twitter. In *Proceedings of the 1st Workshop on Online Social Networks*, pages 19–24. ACM, 2008.

[72] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th International Conference on World Wide Web*, pages 591–600. ACM, 2010.

[73] Shyong K. Lam and John Riedl. Shilling recommender systems for fun and profit. In *Proceedings of the 13th International Conference on World Wide Web*, pages 393–402. ACM, 2004.

[74] Marty Lamb. Tarproxy: Lessons learned and what's ahead. In *Proceedings of the MIT 2004 Spam Conference*, 2004.

[75] Kyumin Lee, James Caverlee, Krishna Y. Kamath, and Zhiyuan Cheng. Detecting collective attention spam. In *Proceedings of the 2nd Joint WICOW/AIRWeb Workshop on Web Quality*, pages 48–55. ACM, 2012.

[76] Kyumin Lee, James Caverlee, and Steve Webb. Uncovering social spammers: Social honeypots + machine learning. In *Proceedings of the 33rd Interna-*

tional ACM SIGIR Conference on Research and Development in Information Retrieval, pages 435–442. ACM, 2010.

[77] Kyumin Lee, Brian David Eoff, and James Caverlee. Seven months with the devils: A long-term study of content polluters on twitter. In *Proceedings of the 5th International Conference on Weblogs and Social Media*. AAAI, 2011.

[78] Janette Lehmann, Bruno Gonçalves, José J. Ramasco, and Ciro Cattuto. Dynamical classes of collective attention in twitter. In *Proceedings of the 21st International Conference on World Wide Web*, pages 251–260. ACM, 2012.

[79] Kristina Lerman and Rumi Ghosh. Information contagion: An empirical study of the spread of news on digg and twitter social networks. In *Proceedings of the 4th International Conference on Weblogs and Social Media*. AAAI, 2010.

[80] Kirill Levchenko, Andreas Pitsillidis, Neha Chachra, Brandon Enright, Márk Félegyházi, Chris Grier, Tristan Halvorson, Chris Kanich, Christian Kreibich, He Liu, Damon McCoy, Nicholas Weaver, Vern Paxson, Geoffrey M. Voelker, and Stefan Savage. Click trajectories: End-to-end analysis of the spam value chain. In *Proceedings of the 2011 IEEE Symposium on Security and Privacyy*, pages 431–446. IEEE Computer Society, 2011.

[81] VI Levenshtein. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

[82] Ee P. Lim, Viet A. Nguyen, Nitin Jindal, Bing Liu, and Hady W. Lauw. Detecting product review spammers using rating behaviors. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, pages 939–948. ACM, 2010.

[83] Yu-Ru Lin, Hari Sundaram, Yun Chi, Junichi Tatemura, and Belle L. Tseng. Splog detection using self-similarity analysis on blog temporal dynamics. In *Proceedings of the 3rd International Workshop on Adversarial Information Retrieval on the Web*, pages 1–8. ACM, 2007.

[84] Cristian Lumezanu and Nick Feamster. Observing common spam in tweets and email. In *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, pages 461–466. ACM, 2012.

[85] Ingrid Lunden. Analyst: Twitter passed 500m users in june 2012, 140m of them in us; jakarta 'biggest tweeting' city. `http://techcrunch.com/2012/07/30/analyst-twitter-passed-500m-users-in-june-2012-140m-of-them-in-us-jakarta-biggest-tweeting-city/`, July 2012.

[86] Alec Lynch. Crowdsourcing is booming in asia. `http://techcrunch.com/2012/12/08/asias-secret-crowdsourcing-boom/`, December 2012.

[87] Justin Ma, Lawrence K. Saul, Stefan Savage, and Geoffrey M. Voelker. Learning to detect malicious urls. *ACM Trans. Intell. Syst. Technol.*, 2(3):30:1–30:24, May 2011.

[88] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[89] Christopher D. Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT Press, 1999.

[90] Benjamin Markines, Ciro Cattuto, and Filippo Menczer. Social spam detection. In *Proceedings of the 5th International Workshop on Adversarial Information Retrieval on the Web*, pages 41–48. ACM, 2009.

[91] Bhaskar Mehta. Unsupervised shilling detection for collaborative filtering. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, pages 1402–1407. AAAI Press, 2007.

[92] Bhaskar Mehta, Thomas Hofmann, and Peter Fankhauser. Lies and propaganda: detecting spam users in collaborative filtering. In *Proceedings of the 12th International Conference on Intelligent User Interfaces*, pages 14–21. ACM, 2007.

[93] Panagiotis T. Metaxas and Joseph DeStefano. Web spam, propaganda and trust. In *Proceedings of the 1st International Workshop on Adversarial Information Retrieval on the Web*. ACM, 2005.

[94] Alan Mislove, Massimiliano Marcon, Krishna P. Gummadi, Peter Druschel, and Bobby Bhattacharjee. Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, pages 29–42. ACM, 2007.

[95] Marti Motoyama, Damon McCoy, Kirill Levchenko, Stefan Savage, and Geoffrey M. Voelker. Dirty jobs: The role of freelance labor in web service abuse. In *Proceedings of the 20th USENIX Conference on Security*. USENIX Association, 2011.

[96] Arjun Mukherjee, Bing Liu, and Natalie Glance. Spotting fake reviewer groups in consumer reviews. In *Proceedings of the 21st International Conference on World Wide Web*, pages 191–200. ACM, 2012.

[97] Arjun Mukherjee, Bing Liu, Junhui Wang, Natalie Glance, and Nitin Jindal. Detecting group review spam. In *Proceedings of the 20th International Conference Companion on World Wide Web*, pages 93–94. ACM, 2011.

[98] Vit Niennattrakul and Chotirat Ann Ratanamahatana. Inaccuracies of shape averaging method using dynamic time warping for time series data. In *Proceedings of the 7th International Conference on Computational Science*, pages 513–520. Springer-Verlag, 2007.

[99] Alexandros Ntoulas, Marc Najork, Mark Manasse, and Dennis Fetterly. Detecting spam web pages through content analysis. In *Proceedings of the 15th International Conference on World Wide Web*, pages 83–92. ACM, 2006.

[100] Michael P. O'Mahony, Neil Hurley, and Guenole C. M. Silvestre. Promoting recommendations: An attack on collaborative filtering. In *Proceedings of the 13th International Conference on Database and Expert Systems Applications*, pages 494–503. Springer-Verlag, 2002.

[101] Myle Ott, Claire Cardie, and Jeff Hancock. Estimating the prevalence of deception in online review communities. In *Proceedings of the 21st International Conference on World Wide Web*, pages 201–210. ACM, 2012.

[102] J.W. Pennebaker, M.E. Francis, and R.J. Booth. *Linguistic Inquiry and Word Count*. Erlbaum Publishers, 2001.

[103] François Petitjean, Alain Ketterlin, and Pierre Gançarski. A global averaging method for dynamic time warping, with applications to clustering. *Pattern Recogn.*, 44:678–693, March 2011.

[104] Nga Pham. Vietnam admits deploying bloggers to support government. `http://www.bbc.co.uk/news/world-asia-20982985`, January 2013.

[105] Andreas Pitsillidis, Chris Kanich, Geoffrey M. Voelker, Kirill Levchenko, and Stefan Savage. Taster's choice: A comparative analysis of spam feeds. In

Proceedings of the 2012 ACM Conference on Internet Measurement Conference, pages 427–440. ACM, 2012.

[106] Andreas Pitsillidis, Kirill Levchenko, Christian Kreibich, Chris Kanich, Geoffrey M. Voelker, Vern Paxson, Nicholas Weaver, and Stefan Savage. Botnet judo: Fighting spam with itself. In *Proceedings of the 17th Network & Distributed System Security Symposium*. The Internet Society, 2010.

[107] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[108] V. V. Prakash. Vipul's razor. `http://razor.sourceforge.net/`, 2004.

[109] Matthew B. Prince, Benjamin M. Dahl, Lee Holloway, Arthur M. Keller, and Eric Langheinrich. Understanding how spammers steal your e-mail address: An analysis of the first six months of data from project honey pot. In *Proceedings of the 2nd Conference on Email and Anti-Spam*, 2005.

[110] Emil Protalinski. Facebook: 8.7 percent are fake users. `http://news.cnet.com/8301-1023_3-57484991-93/facebook-8.7-percent-are-fake-users/`, August 2012.

[111] Zhiyun Qian, Z. Morley Mao, Yinglian Xie, and Fang Yu. Investigation of triangular spamming: A stealthy and efficient spamming technique. In *Proceedings of the 2010 IEEE Symposium on Security and Privacy*, pages 207–222. IEEE Computer Society, 2010.

[112] Zhiyun Qian, Zhuoqing Morley Mao, Yinglian Xie, and Fang Yu. On network-level clusters for spam detection. In *Proceedings of the 17th Network & Distributed System Security Symposium*. The Internet Society, 2010.

[113] Anirudh Ramachandran and Nick Feamster. Understanding the network-level behavior of spammers. In *Proceedings of the 2006 Conference on Applica-*

tions, Technologies, Architectures, and Protocols for Computer Communications, pages 291–302. ACM, 2006.

[114] Anirudh Ramachandran, Nick Feamster, and Santosh Vempala. Filtering spam with behavioral blacklisting. In *Proceedings of the 14th ACM Conference on Computer and Communications Security*, pages 342–351. ACM, 2007.

[115] Jacob Ratkiewicz, Michael Conover, Mark Meiss, Bruno Gonçalves, Alessandro Flammini, and Filippo Menczer. Detecting and tracking political abuse in social media. In *Proceedings of the 5th International Conference on Weblogs and Social Media*. AAAI, 2011.

[116] Sanjog Ray and Ambuj Mahanti. Privacy, security, and trust in kdd. chapter Strategies for Effective Shilling Attacks against Recommender Systems, pages 111–125. Springer-Verlag, 2009.

[117] Dan Reid. Is twitter winning the war on spam? our stats do not support this assertion. `http://thedustpan.com/2010/03/is-twitter-winning-the-war-on-spam-our-stats-do-not-support-this-assertion/`, March 2010.

[118] Daniel M. Romero, Brendan Meeder, and Jon Kleinberg. Differences in the mechanics of information diffusion across topics: idioms, political hashtags, and complex contagion on twitter. In *Proceedings of the 20th International Conference on World Wide Web*, pages 695–704. ACM, 2011.

[119] Joel Ross, Lilly Irani, M. Six Silberman, Andrew Zaldivar, and Bill Tomlinson. Who are the crowdworkers?: shifting demographics in mechanical turk. In *CHI '10 Extended Abstracts on Human Factors in Computing Systems*, pages 2863–2872. ACM, 2010.

[120] Mehran Sahami, Susan Dumais, David Heckerman, and Eric Horvitz. A bayesian approach to filtering junk E-mail. In *AAAI Workshop on Learning for Text Categorization*. AAAI Technical Report WS-98-05, 1998.

[121] Paul Sawers. 8.7% of facebook accounts are fake? still, that leaves 872m bona fide users. `http://thenextweb.com/facebook/2012/08/02/8-7-of-facebook-accounts-are-fake-still-that-leaves-872m-bona-fide-users/`, August 2012.

[122] High Scalability. Mollom architecture – killing over 373 million spams at 100 requests per second. `http://highscalability.com/blog/2011/2/8/mollom-architecture-killing-over-373-million-spams-at-100-re.html`, February 2011.

[123] Sushant Sinha, Michael Bailey, and Farnam Jahanian. Improving spam blacklisting through dynamic thresholding and speculative aggregation. In *Proceedings of the 17th Network & Distributed System Security Symposium*. The Internet Society, 2010.

[124] Lance Spitzner. The honeynet project: Trapping the hackers. *IEEE Security and Privacy*, 1(2):15–23, 2003.

[125] Bruce Sterling. The chinese online 'water army'. `http://www.wired.com/beyond_the_beyond/2010/06/the-chinese-online-water-army/`, June 2010.

[126] Gianluca Stringhini, Manuel Egele, Apostolis Zarras, Thorsten Holz, Christopher Kruegel, and Giovanni Vigna. B@bel: leveraging email delivery for spam mitigation. In *Proceedings of the 21st USENIX Conference on Security Symposium*. USENIX Association, 2012.

[127] Gianluca Stringhini, Thorsten Holz, Brett Stone-Gross, Christopher Kruegel, and Giovanni Vigna. Botmagnifier: Locating spambots on the internet. In *Proceedings of the 20th USENIX Conference on Security*. USENIX Association, 2011.

[128] Xue-Feng Su, Hua-Jun Zeng, and Zheng Chen. Finding group shilling in recommendation system. In *Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, pages 960–961. ACM, 2005.

[129] SpamAssassin Development Team. The apache spamassassin project. `http://spamassassin.apache.org/`, 2004.

[130] Martin Theobald, Jonathan Siddharth, and Andreas Paepcke. Spotsigs: robust and efficient near duplicate detection in large web collections. In *Proceedings of the 31st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 563–570. ACM, 2008.

[131] Kurt Thomas, Chris Grier, Justin Ma, Vern Paxson, and Dawn Song. Design and evaluation of a real-time url spam filtering service. In *Proceedings of the 2011 IEEE Symposium on Security and Privacy*, pages 447–462. IEEE Computer Society, 2011.

[132] Kurt Thomas, Chris Grier, Dawn Song, and Vern Paxson. Suspended accounts in retrospect: an analysis of twitter spam. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, pages 243–258. ACM, 2011.

[133] Etsuji Tomita, Akira Tanaka, and Haruhisa Takahashi. The worst-case time complexity for generating all maximal cliques and computational experiments. *Theor. Comput. Sci.*, 363(1):28–42, 2006.

[134] TradingEconomics. Gni per capita; atlas method (us dollar) in bangladesh. http://www.tradingeconomics.com/bangladesh/gni-per-capita-atlas-method-us-dollar-wb-data.html, 2011.

[135] TREC. Terabyte track. http://www-nlpir.nist.gov/projects/terabyte/, 2004.

[136] TREC. Spam track. http://plg.uwaterloo.ca/~gvcormac/treccorpus07/, 2007.

[137] Twitter. State of twitter spam. http://blog.twitter.com/2010/03/state-of-twitter-spam.html, March 2010.

[138] Twitter. The engineering behind twitter's new search experience. http://engineering.twitter.com/2011/05/engineering-behind-twitters-new-search.html, May 2011.

[139] Twitter. The twitter rules. http://support.twitter.com/articles/18311-the-twitter-rules, 2012.

[140] Twitter. Twitter turns six. http://blog.twitter.com/2012/03/twitter-turns-six.html, March 2012.

[141] Petros Venetis and Hector Garcia-Molina. Quality control for comparison microtasks. In *Proceedings of the 1st International Workshop on Crowdsourcing and Data Mining*, pages 15–21. ACM, 2012.

[142] Ellen M. Voorhees and Hoa Trang Dang. Overview of the trec 2005 question answering track. In *Proceedings of the 14th Text REtrieval Conference*, 2005.

[143] Gang Wang, Christo Wilson, Xiaohan Zhao, Yibo Zhu, Manish Mohanlal, Haitao Zheng, and Ben Y. Zhao. Serf and turf: Crowdturfing for fun and

profit. In *Proceedings of the 21st International Conference on World Wide Web*, pages 679–688. ACM, 2012.

[144] Nan Wang, Srinivasan Parthasarathy, Kian-Lee Tan, and Anthony K. H. Tung. Csv: visualizing and mining cohesive subgraphs. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 445–458. ACM, 2008.

[145] Steve Webb, James Caverlee, and Calton Pu. Introducing the webb spam corpus: Using email spam to identify web spam automatically. In *Proceedings of the 3rd Conference on Email and Anti-Spam*, 2006.

[146] Steve Webb, James Caverlee, and Calton Pu. Social honeypots: Making friends with a spammer near you. In *Proceedings of the 5th Conference on Email and Anti-Spam*, 2008.

[147] Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. Twitterrank: finding topic-sensitive influential twitterers. In *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*, pages 261–270. ACM, 2010.

[148] Wikipedia. Internet water army. `http://en.wikipedia.org/wiki/Internet_Water_Army`, February 2013.

[149] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques (Second Edition)*. Morgan Kaufmann, June 2005.

[150] Gilbert Wondracek, Thorsten Holz, Engin Kirda, and Christopher Kruegel. A practical attack to de-anonymize social network users. In *Proceedings of the 2010 IEEE Symposium on Security and Privacy*, pages 223–238. IEEE Computer Society, 2010.

[151] WPBeginner. Akismet 101 guide for all wordpress users. http://www.wpbeginner.com/beginners-guide/akismet-101-guide-for-all-wordpress-users/, June 2012.

[152] Baoning Wu and Brian D. Davison. Identifying link farm spam pages. In *Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, pages 820–829. ACM, 2005.

[153] Baoning Wu, Vinay Goel, and Brian D. Davison. Topical trustrank: using topicality to combat web spam. In *Proceedings of the 15th International Conference on World Wide Web*, pages 63–72. ACM, 2006.

[154] Bin Wu, Shengqi Yang, Haizhou Zhao, and Bai Wang. A distributed algorithm to enumerate all maximal cliques in mapreduce. In *Proceedings of the 2009 4th International Conference on Frontier of Computer Science and Technology*, pages 45–51. IEEE Computer Society, 2009.

[155] Fang Wu and Bernardo A. Huberman. Novelty and collective attention. *Proceedings of the National Academy of Sciences*, 104(45):17599–17601, November 2007.

[156] Guangyu Wu, Derek Greene, Barry Smyth, and Pádraig Cunningham. Distortion as a validation criterion in the identification of suspicious reviews. In *Proceedings of the 1st Workshop on Social Media Analytics*, pages 10–13. ACM, 2010.

[157] Tao Xia, Chuang Zhang, Jingjing Xie, and Tai Li. Real-time quality control for crowdsourcing relevance evaluation. In *Proceedings of the 3rd IEEE International Conference on Network Infrastructure and Digital Content*, pages 535–539, 2012.

[158] Chao Yang, Robert Harkreader, Jialong Zhang, Seungwon Shin, and Guofei Gu. Analyzing spammers' social networks for fun and profit: a case study of cyber criminal ecosystem on twitter. In *Proceedings of the 21st International Conference on World Wide Web*, pages 71–80. ACM, 2012.

[159] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, pages 412–420. Morgan Kaufmann Publishers Inc., 1997.

[160] YouTube. Statistics. `http://www.youtube.com/yt/press/statistics.html`, March 2013.

[161] Qi Zhang, Yue Zhang, Haomin Yu, and Xuanjing Huang. Efficient partial-duplicate detection based on sequence matching. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 675–682. ACM, 2010.

[162] Aaron Zinman and Judith S. Donath. Is britney spears spam? In *Proceedings of the 4th Conference on Email and Anti-Spam*, 2007.