UNIVERSITY OF CALIFORNIA
RIVERSIDE

Profiling, Modeling and Classifying On-Line Users' Behavior

A Dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

by

Tai Ching Li

September 2017

Dissertation Committee:

Prof. Michalis Faloutsos, Chairperson
Prof. Srikanth Krishnamurthy
Prof. Vagelis Hristidis
Prof. Vagelis Papalexakis

The Dissertation of Tai Ching Li is approved:

Committee Chairperson

University of California, Riverside

Acknowledgments

I express my deepest appreciation to my committee chair, Professor Michalis Faloutsos, who has taught me how to conduct research, how to write scientific paper, the art of presentation, and last but not least, the spirit of adventure. I would also like to extend my gratitude to my committee member, Professor Vagelis Papalexakis, whose work completed the last mile of my PhD degree. In addition, I thank Professor Vagelis Hristidis and Professor Srikanth Krishnamurthy for joining my committee and all my collaborators, without whose help, I would not have been here.

To my beloved parents and my sister for all the support.

ABSTRACT OF THE DISSERTATION

Profiling, Modeling and Classifying On-Line Users' Behavior

by

Tai Ching Li

Doctor of Philosophy, Graduate Program in Computer Science
University of California, Riverside, September 2017
Prof. Michalis Faloutsos, Chairperson

The online behavior of users is generating an unprecedented wealth of information, which is implicitly public. First, the amount of information is increasing because users spend more time and engage more through posts, comments etc. Second, the users volunteer the information from the most part, but what is less obvious is that the ability to quickly collect, aggregate and analyze that information can provide very involved information about the user. Individual piece of information is harmless, but once combined with others could lead to revelations that the user has not imagined.

The focus of our research could be captured in the following question: how much user-specific information can be extracted from users online behavior? We study this problem in two different domains. First, we examine commenting platforms, and examine how much information about users can be extracted from their posts. Commenting platforms are companies that facilitate the backend management of comments for a wide range of websites. Second, we study the third-party tracking which is an Internet-wide privacy issue that happen across multiple domains/websites. Third-party tracking described the phe-

nomenon when the web-browsing behavior of the user is revealed and collected by websites that the user has not explicitly visited.

Our contribution of this dissertation is focusing on (a) understanding how much user information can be extracted from commenting platforms, and (b) quantifying the extent of the third-party tracking. First, we develop a systematic way to profile users based on their commenting behavior. Our goal is to identify a small set of features that can profile users and reveal fundamental patterns and anomalies. Second, we propose a systematic method to detect and classify antisocial behavior in commenting platforms. We use the term antisocial behavior to describe activities like trolling, spamming, fanatic posting etc. The focus here is in developing a comprehensive and interpretable way to define and detect antisocial behaviors. Last, we quantify the extent of third-party tracking and develop a user-implementable counter-measure. We find that third-party tracking is much more pervasive than one could imagine and this could pose a threat to one's privacy.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Internet has become an integral part of our life, varieties of services (email, social networks and shopping) can be accessed anytime, anywhere from all kind of devices today. In the mean time, users interact with the Internet, information is created and then diseeminated and retained in the network. This information could be in the form of content provided voluntarily from users, such as Facebook posts, Tweets and comments toward news articles, or in the more salient form of a behavior trace, like one's browsing history. Considering the time a modern human spends on the Internet, one cas expect that the this online information, when aggregated, can reveal singificant information about the person including, habits, political preferences, gender, personality and location.

The overachieving problem of the dissertation is: how much user-specific information can be extracted from the online users behavior? We study this problem in two different domains, "commenting platforms" and "third-party tracker". First, we study commenting platforms, and examine how much information about users can be extracted from their

comments. Commenting platforms is a web service provided by third party companies to news websites typically, in order to facilitate the backend management of comments for a wide range of websites. Such platforms include Disqus [30], LiveFyre [45], and IntenseDebate [27]. Second, we study the third-party tracking which is an internet-wide privacy issue that happen across multiple domains/websites/browsers. Third-party tracking described the phenomenon when the web-browsing behavior of the user is revealed and collected by websites that the user has not explicitly visited.

The underlying motivation for this work is the potential privacy exposure that users have. Each piece of user-specific information extracted form their behavior is little but combining them together results in an uncovering of personal information. The most worrisome thing is that users actually "enjoy" such a situation when these information are used to provide user-targeted content. For example, buying a plane ticket to New York will make all advertisement from Google become "Booking your hotel in NYC!". The price of the convenience is exposing one's privacy to third-party. This leads to the following question: how much information can we extract from this online information and what can this tell us about the user?

In this dissertation, we present three pieces of work that are aimed toward answering the above grand question. In Chapter 2 and 3, we study users behavior from their commenting activities and use it to develop user profiles, inlcuding the detection of malicious users. In Chapter 4, we examine the extend of third-party tracking and we quantify that extent of exposure that a user's browsing history to third-party websites.

As our first contribution, we develop a profiling framework that we call **Com-**

mentProfiler (presented in Chapter 2). Our goal is to understandg users behavior on commenting platform and identify key features that best describe the behavior pattern. To conduct our study, we crawl users' data from DSIQUS, which is arguably the largest commenting platforms. The data include 1 million users and 19 millions comments spanning Nov-2007 to Jan-2015. We claim three main contributions in this work: (a) we study users behavior from 3 different dimensions includes engagement properties (user-article interaction), social properties (user-user interaction) and users' temporal properties, (b) we identify the key features of users' behavior of each dimensions and propose a visualization profile, and (c) we identify parasitic behaviors on the commenting platform and show that our profiles are able to recognize it.

These parasitic behaviors include collusion, spamming/trolling and chat-room usage: (a) We find two large cliques of sizes 29 and 34 with unusual social behavior that suggests collusion, users in each cluster are collaborating on more than 100 different articles with high intensity, (b) 38 users exhibit spamming or trolling behavior with a hit-n-away commenting pattern on all articles they participate and (c) 24 users use Disqus as a chat-room on 19 different websites.

As our second contribution, we develop an approach that we call **TrollSpot** (detailed in Chapter 3), which extends the work in **CommertProfiler**. In this work, we focus on developing a systematic approach to classify user behaviors leveraging our previous study. In more detail, the goal is to detect and classify misbehaving users in commenting platforms. The term misbehaving is used to describe activities like trolling, spamming, fanatic posting etc. We highlight our contributions in TrollSpot the following points below.

First, we develop two approaches to classify users behavior: (a) feature-based, and (b) behavior-based classifier. Both provide accurate result of identifying misbehaving users with a baseline of 80% precision and 79% recall. The behavior-based classifier uses a two-stage classification mapping features into behavior and then behaviors to the result. With a light trade-off of accuracy, this classifier provides a highly interpretable result.

Second, we use our classifiers to provide a fine-grained malicious role classification. We separate misbehaving users into three categories, trolls, spammers and fanatics. We show that our classifier can identify the difference between their behavior with an overall accuracy of 73.3%. To the best of our knowledge, this is the first work focus on classifying the role of malicious users on the commenting platforms in such a level of granularity.

The last contribution is that we use TrollSpot to study the existence of misbehaving users in 4 news media (ABC News, CNBC News, Bloomberg Vies and a DISQUS channel). We find a total 1,378 (0.86%) users are labeled as misbehaving includes 866 trolls, 165 spammers and 597 fanatics. CNBC has the most misbehaving users with 1,167 (1.48%) labeled, comparing to ABC News having 717 (0.56%) labeled and Bloomberg Views, finance-oriented news media, has only 131 (0.39%) of users labeled as misbehaving.

As our third contribution, we develop an approach to determine third-party tracking, which we call **TrackAdvisor** (presented in Chapter 4). In this work, we study the third-party tracking which is a phenomenon addressing users browsing history collected by third-party websites without informing users. The collected information will be re-built and complete a list of URL reflecting users interest and other personal information. We have three contribution in this work: we (a) develop a supervised classifier which can identify

HTTP request sent to a third-party tracker, (b) propose counter measures against third-party trackers, and (c) quantify the extend of third-party tracker on the Internet.

To maintain a better flow in the document and avoid duplication, we provide an overview of prior literature in the first section of each chapter, and a more extended section at the end of each chapter.

Chapter 2

CommentProfiler: Detecting trends and parasitic behaviors in on-line comments

Online commenting on articles has evolved into a highly-social, popular, and interactive activity over the last 10 years. Two interesting phenomena have contributed to the emergence of this phenomenon. First, there are a few companies that facilitate the backend management of comments for **a wide range of websites**. We use the term **commenting platform** to refer to such platforms, which include Disqus [30], LiveFyre [45], and IntenseDebate [27]. Second, many users exhibit intense commenting activity, such as spending many hours daily leaving comments. As a result of these two phenomena, one can obtain a comprehensive view of the commenting behavior of people across a large number of sites since all the user information is housed in one place. These platforms are becom-

ing more and more like social networks, since users can "follow" other users, and often comments are addressed to other users (e.g. "Hello john123! Long time no see.").

We begin by providing some terms and definitions. A **user** is defined by a platform account, which enables her to leave comments to articles on a website that uses the commenting platform. A user may leave more than one comment for an article, which leads us to define the **engagement** of a user for that article. An engagement has a time duration and intensity in terms of number of comments, with both metrics being an indirect indication of the user's interest in that article or topic in general.

Due to the lack of a better term, when two users comment on the same article, we say that they **collaborate** and we use the term **collaboration** to describe this joint activity. Users can leave comments for an article or respond to a previous comment, but we do not distinguish between these two types of comments in this work, despite some initial exploration. We use the term **collaboration intensity** to refer to the number of articles for which two users collaborate, with the intention of capturing the extent of the collaboration between users.

The overarching question in this chapter is how we can detect unusual and parasitic users in these commenting platform. We want to identify patterns and anomalies focusing mostly on the behaviors of the users. In fact, we use the content of the posts very lightly or as a validation of suspicious activity. Specifically, the input to the problem is the commenting information of the users. This includes: the author of the comment, the time it was posted, and information on the article it was posted for. The goal is to determine patterns of behavior, per user, and per user-article pair and then identify and explain surprising and

anomalous phenomena.

Beyond the pure scientific curiosity, the motivation of our work is the protection of the commenting platforms from abuse. Such protection is important for ensuring that these platforms serve their primary purpose, which is the honest exchange of opinions among readers. We want to empower commenting platforms with techniques and tools to identify parasitic behaviors. The goal is to shield users from malicious actors, who could try to push their opinions and agendas, and use it as a mechanism to bias the public opinion, and even intimidate and propagate hostility and hate in society. For example, an organized group could systematically and aggressively attack a product, an idea, a person, or an institution.

To the best of our knowledge, commenting platforms have attracted very little attention so far. By contrast, most social media studies focus on analyzing the posts of Online Social Networks (OSNs) like Facebook, Twitter and blogs. We group prior work into three broad areas: (a) inferring the geographic location of users based usually on the textual content; (b) detecting spam, abusive behaviors and malware in OSNs; and (c) inferring users' psychological state and personality traits. We will discuss more about related work in § 2.4.

In this chapter, we conduct an extensive study to identify patterns and anomalies among users. We collect our data by "crawling" Disqus, arguably the largest commenting platform today. In our effort to develop a systematic approach to behavior profiling, we identify three dimensions in the user behavior: (a) social interaction or user-user interaction, (b) engagement or user-article interaction, and (c) temporal features of the user behavior. Our work focuses on two thrusts: (a) we identify patterns and common behaviors, and

(b) we identify surprising behaviors. Crawling Disqus, we collect 19 million comments between Nov-2007 to Jan-2015, and study 109K unique user accounts who have more than 10 comments. We summarize our key results below.

a. Identifying behavioral patterns. We study the behavior of users along the aforementioned three dimensions, and we make the following observations.

i) Social behavior: We find that users have many ephemeral collaborations, but very few intense collaborations: 82% of users have more than 10 collaborators, but, 99% of these pair-wise collaborations have an intensity lower than 13, while we find only 0.03% collaborations with intensity larger than 100.

ii) User-article engagement: The engagement of users per article exhibit a skewed distribution with most engagements being short lived, and with few exceptionally long ones. Specifically, we find that 78.9% of engagements last less than ten minutes, while 4.2% engagements last more than one day and 0.54% of the engagements last more than one month.

iii) Temporal behavior. Going beyond the expected periodic behaviors, we focus on the downtime (think night-time) in the users' daily behavior and identify three major groups with 3, 9, 15 hours a day of engagement time. In other words, we observe users that spend 15 hours a day writing comments, which is very suspicious and could be helpful in detecting anomalous behaviors.

As expected, we also observe that most behaviors are highly variable and can be described with skewed statistical distributions. We also observed the skewed distribution of the behavior, the further investigation shows that the number of comments per: (a) user,

(b) article, and (c) website follow the (a) truncated power law, (b) log-normal, and (c) power law distributions respectively. For example, We find that 0.8% of users have more than 500 comments (average is 22), 2.9% of the articles have more than 500 comments (average is 91).

b. Identifying surprising behaviors. We identify abnormal behaviors and potentially malicious users by leveraging our metrics and observations. Our goal is showing: (a) our metrics are informative, and (b) some very surprising behaviors indeed exist.

i) Collusion: We find two large cliques of sizes 29 and 34 with unusual social behavior that suggests collusion: all clique members collaborate with each other with collaboration intensity of 100+ articles, and we also find that they support each other's opinions.

ii) Spamming and Trolling: We find 38 users that exhibit trolling and spamming behavior. Their engagement behavior and temporal patterns are unusual: their engagement intensity is very low, mostly one comment per article, and they do not show the periodic and predictable temporal pattern.

iii) Chatroom use: We find 24 users on 19 different websites where Disqus is used as a chat room, anchored by a fake empty article. We identify these chatrooms by finding the chatroom users, who are characterized by unusually intense engagements: more than 500 comments and engagement durations that exceed one month.

This chapter demonstrates the richness of information and interesting user behaviors that can be found in these much-less-studied online platforms, and we will fully explore it in the next chapter.

2.1 Basic Statistics and Data

Disqus is arguably the most widespread commenting platform as it reached one billion unique visitors a month and about roughly two and a half million site installs by May 2013 [29]. There are two main reasons for the success of commenting platform. First, it enables users to comment on multiple sites with a single sign-in, thus eliminating the need for multiple registration, passwords and logins. Second, the host website does not have to develop and manage its own commenting infrastructure, which save both resources and bandwidth.

Disqus Data Collection: We explain how we collect data from Disqus. Disqus exposes its database to its clients (the blogs and news services) through an Application Programming Interface, or API. In fact, if a **user ID** associated with a user profile is known, anyone can retrieve every comment that this user has made on any website that uses Disqus. Furthermore, the following pieces of information are also available:

- The timestamp of the comment (in UTC), and any URL of attached media (video or image).
- Basic metadata about the article which includes: (a) the title of the article, and (b) the URL of the article.

As mentioned above, we only need a set of user ID to retrieve all their comments from the user. The user ID happens to be number, which seems to correspond to the order with which the person joined Disqus. We decided to focus on ID c within $1,000,000 \leq c \leq 1,999,999$. This seemed to be the more densely populated range compared to the zero to

1M range of IDs that we tried initially. Given the throttling mechanisms of the site, it took us roughly three months to retrieve comments from every user in that range. We collected 19M comments spanning Nov-2007 to Jan-2015.

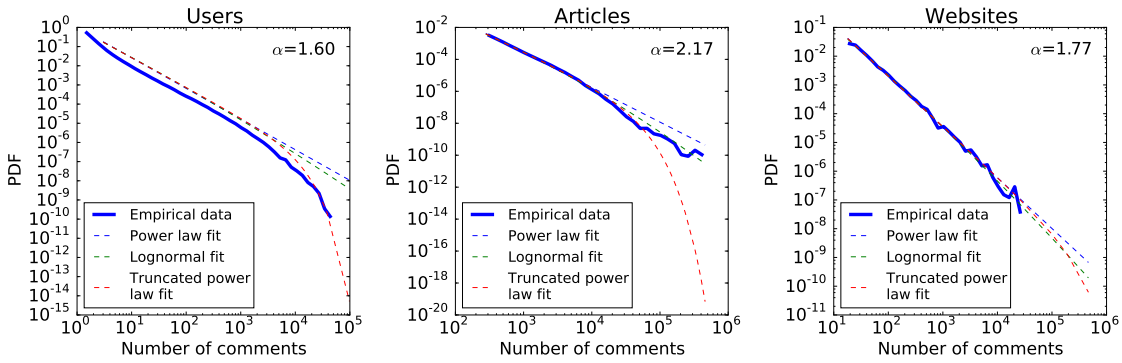
Our User-centric data set: $D_{\mathbf{Comm}}$. To create meaningful user profile, we need them to have non-trivial posting behavior. Thus, for this kind of analysis, we filter out users with fewer than 10 comments each. The dataset has 109,564 users, 19,121,250 comments, 3,474,360 articles and 91,878 **forums** as defined by Disqus. Disqus uses the term forum to indicate a website that provides articles, and we adopt their definition of a website in our work. For the rest of this chapter, the term **forum** and **website** are used interchangeably.

The daily behavior dataset: $D^{\boldsymbol{t}}_{\mathbf{Comm}}$. To study the daily pattern of users, we create a new data set, $D^{\boldsymbol{t}}_{\text{Comm}}$. For each user, we identify their 90-day window that has the maximum number of comments. We select users that have more than 100 comments in the window, and this leads to 26,009 unique users. The starting day of that window varies among users. This was necessary as some users are active for a while and then disappear for a long-time or forever. For each user, we aggregate their comments from that 90-day interval per hour of the day. These bins form a time series for each user as the one shown later in Figure 2.6(c).

Distributions of the number of comments. We model the relationships between the number of comments and each of the following entities:

1. User

2. Website



(a) (b) (c)

Figure 2.1: Modeling the relationship between the number of comments and (a) commenters, (b) article and (c) website

3. Article

During preliminary inspection on Figure 2.1, we observed that the three aforementioned relationships seem to follow some power law with different variations at the tails. We present our investigation in the context below and show how well each of the relationships fit with each of the following target distributions:

1. Power law

$$f(x) = x^{-\alpha}$$

2. Exponential

$$f(x) = e^{-\lambda x}$$

3. Log-Normal

$$f(x) = \frac{1}{x} e^{-\frac{(\ln x - \mu)^2}{2\sigma^2}}$$

4. Truncated power law, which has the power laws scaling behavior over some range but is truncated by an exponentially bounded tail. This distribution is expressed by the

equation:

$$f(x) = x^{-\alpha} e^{-\lambda x}$$

We use the data in D_{Comm} to model the number of comments and the tool that we use to model the Disqus data is called `powerlaw`, a module written for the Python language by Alstott et. al. [15], which in turn was built upon the findings of [23, 42].

Alstott et. al. stated in [15] that the question of "how well the data fits the power law" is frequently less important and relevant than whether "there exists a distribution that better describes the data". We follow their advice here and instead of trying to fit our data against only the power law distribution, we try to find one distribution among four (power law, exponential, log-normal, truncated power law) that fits the data best. Towards this end, we leverage the function `distribution_compare` of the `powerlaw` module to study how well the data we have fits against each of the following pairs :

1. ⟨Power law, Exponential⟩
2. ⟨Power law, Log-Normal⟩
3. ⟨Power law, Truncated power law⟩

Invoking the function `distribution_compare` (which is the method Alstott et. al. recommended to find the best-fit distributions among several) on each pairs yields a **Log-likelihood ratio** (R) and a corresponding significance (p). If R is positive, the data fits the *first* distribution in the pair better than it does the second and vice versa if R is negative. The large the absolute value of R is, the better the fit. In Table 2.1, we show the R and p values yielded by comparing how well the data fits the power law distribution

Comments per	Exponential		Log-Normal		Truncated power law	
	R	p	R	p	R	p
User	119.24	0.00	-30.56	0.00	**-44.04**	0.00
Website	6.13	0.00	-2.25	0.02	-1.17	0.01
Article	39.95	0.00	**-24.26**	0.00	-15.08	0.00

Table 2.1: Power Law fit versus (a) Exponential fit, (b) Log-Normal fit, and (c) Truncated power law fit. The R value in **bold** indicates which distribution provides the best fit.

versus each of the other three aforementioned distributions. As we can see from the table, it is very clear that:

Comments per user. The truncated power law distribution describes the number of comments per user better than the power law distribution as we explain below:

1. Against the exponential distribution, $R = +119.24$, indicating that the data favors **power law** more.

2. Against the log-normal distribution, $R = -30.56$, which means that the data favors the **log-normal** distribution *more than* the power law one.

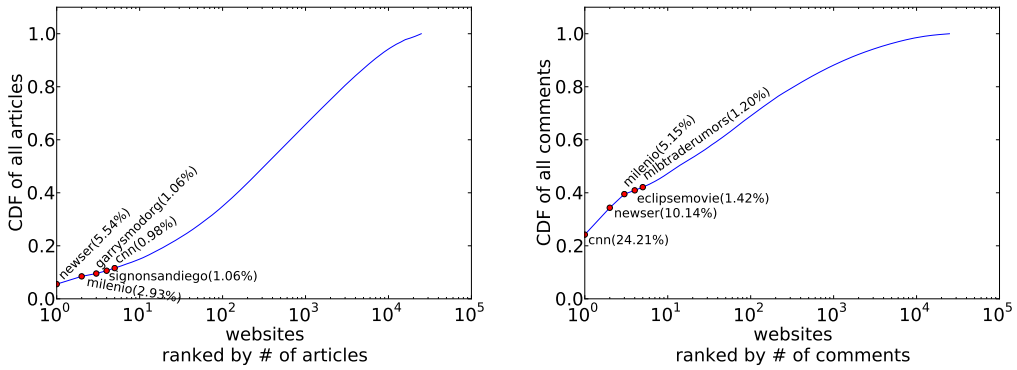
3. Against the truncated power law, $R = -44.04$, indicating that the data is *even more* strongly supported by the **truncated powerlaw** distribution than the power law.

Indeed, Figure 2.1(a) does show that the exponential cut-off at the tail end of the empirical data follows the downward bend of the truncated power law distribution. The estimated values for the parameters are: $\alpha = 1.554665$, $\lambda = 0.000152$.

Comments per article. The number of comments per article seems to be best

fitted by the log-normal distribution, by the same reasoning of the R values and by the closeness with which the empirical data follows the line representing the log-normal distribution in Figure 2.1(b). The estimated values for the parameters are: $\mu = 2.050046$, $\sigma = 0.000051$.

Comments per website. Describing the distribution of the number of comments per *website* is somewhat less clear, as the low values of R and p seem to indicate that all four distributions fit the data equally well. This is not uncommon, as Alstott et. al. [15] also describe one such case. In our case, we opt to favor the **power law** distribution as the simpler (fewer parameters) of the choices and has a value of $\alpha = 1.7660$ in our dataset.



(a) Distribution of the number of articles per website

(b) Distribution of the number of comments per website

Figure 2.2: Distribution of articles and comments per website

A website-centric view. We take a website-centric point of view, and we show the highly skewed distributions of the numbers of articles and comments per website in our D_{Comm} dataset in Figure 2.2. We find that 80% (1.52 million) of all comments are found on only 0.45% (116) websites and 80% of all articles (315K) belong to only 4 of websites.

In Figure 2.2(a), we show the top five websites with the highest number of articles and in Figure 2.2(b), we show the top five with the highest number of comments. It is interesting to see that even though Newser has a larger number of commented articles than CNN, CNN has more comments overall. In fact, the average number of comments on Newser article is 27 compared to 670 for a CNN article, considering articles that appear in our dataset meaning that they have at least one comment. CNN is the website with the highest number of comments, receiving 12,000 comments a day on average.

2.2 Modeling user behavior

We study the behavior of users on Disqus to identify major patterns and develop a frame of reference, which we leverage in detecting surprising behaviors in § 2.3.

2.2.1 The social properties of the user interactions

We model and study how users relate to each other in terms of commenting on the same articles and use the term collaboration to capture this relationship as we defined earlier. We define the **Social Graph** $SG = \langle V, E \rangle$ as an undirected weighted graph that:

1. V is the set of vertices and each vertex $v \in V$ is a user.
2. E is the set of edges, where the edge e_{ij} between nodes v_i and v_j exists, if and only if the users i and j collaborated on at least one article.
3. The weight $W(e_{ij})$ of the edge is the collaboration intensity (the number of articles that i and j have collaborated on).

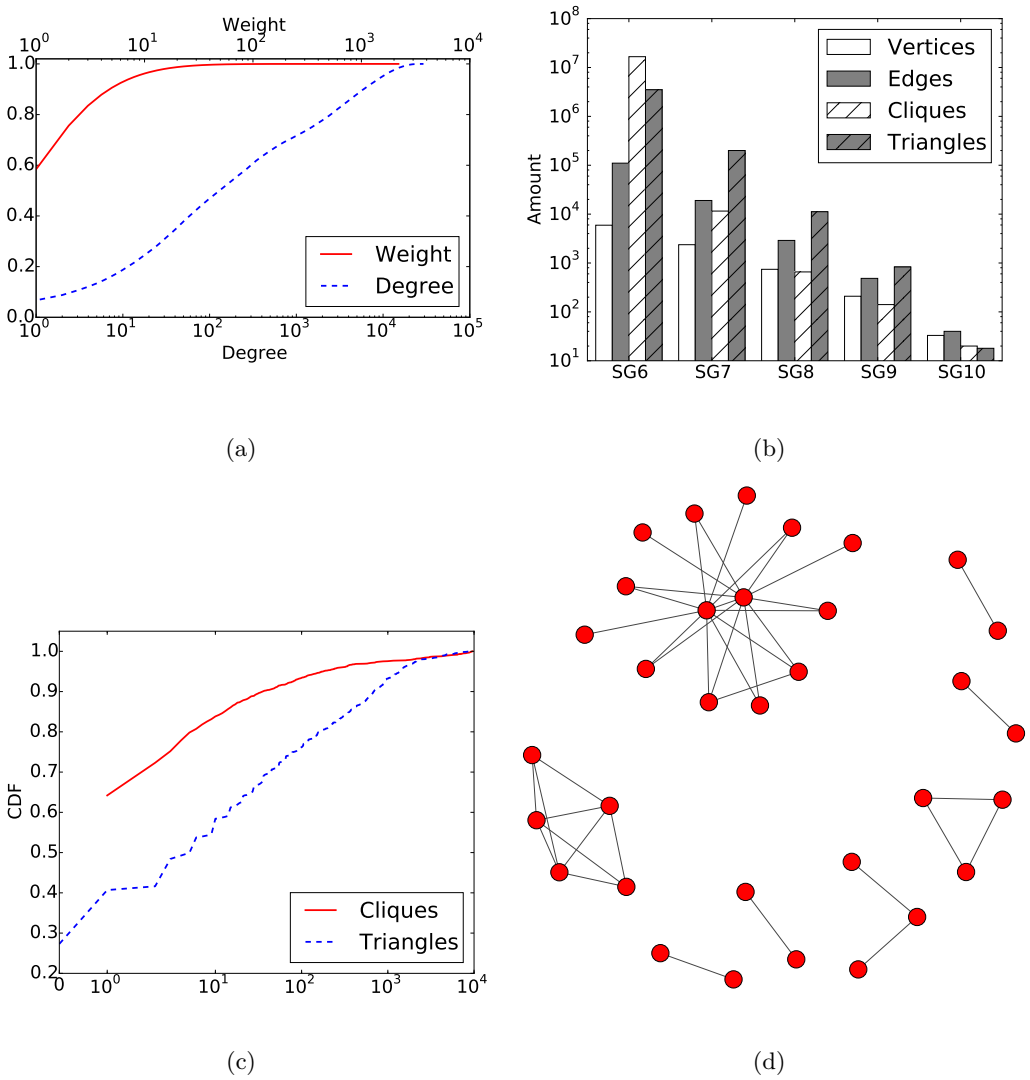


Figure 2.3: **Social properties.** (a) CDF of the node degrees and the edge weight distributions of the SG_0 graph. (b) The number of vertices, edges, cliques and triangles for various SG_n. (c) Distribution of cliques and triangles of users in SG_7. (d) A visual representation of SG_{10}.

The SG graph is quite dense with 102,028 vertices, 87,667,686 edges, an average degree of 1718.5 and a median degree of 130. We are interested in both the connectivity and the weight distribution of the graph. Figure 2.3(a) shows that 95% of the edges have weights lower than 10 and there are extreme cases: 19,018 edges (2.1%) have weights higher than

128, 485 edges higher than 512, and 40 higher than 1024. The highest value of collaboration intensity is 2,198 articles. Although 82% of users have no more than 10 collaborators in SG, there are few intense collaborations as we show below.

Studying the collaboration intensity: the SG_n graph sequence. To focus on highly collaborative user-pairs, we define a sequence of the subgraphs SG_n from SG by filtering out edges with low collaboration intensity. A subgraph SG_n is a subset $\langle V_n, E_n \rangle$ of $\langle V, E \rangle$ and $E_n = \{e_{ij} \in E \mid W(e_{ij}) \geq 2^n\}$. In other words, subgraph SG_n focus on edges with higher collaboration intensity as n increases. It is easy to see that SG_0 and SG are the same graph. Also, the lower n subgraphs include the higher n subgraphs: $SG_k \subseteq SG_l$ for $k > l$.

Focusing on highly collaborative user pairs. We find that for $n > 10$, we end up with the null graph, therefore SG_{10} represents the most collaborative pairs of nodes, which is shown in Figure 2.3(d). The graph includes 33 users, 40 edges, 20 maximal cliques (we always refer to maximal cliques in this work), and 8 connected components.

Finding indications of collusion. We can make several interesting observations by analyzing subgraph SG_{10}. First, each connected component in SG_{10} seems to be focusing on a particular website. Most of the jointly commented articles in each connected component belong to particular site, which is different than that of the other components. These sites are Newser, Foxnews, Milenio, RawStory, Fotbollskanalen, EscantonBlog, nonamedufus.blogspot, and stfueverybody.tumblr.com (which recently was shut down by its owner). Second, all the usernames of the users in the largest component of SG_{10} start with the prefix **newser-**. This shows that the users registered in Disqus through a link from

Newser, as this prefix is automatically given by Disqus. Looking more closely, we also find that all of these users registered on the same day. In fact, two of these users have collaborated on 2,198 articles, which is the highest collaboration intensity, and their registration times are only forty minutes apart. We continue the exploration of collusion in § 2.3.

Social density: studying the local connectivity. We study the local connectivity of the nodes using the number of maximal cliques and triangles for which a node is part of in a given SG_n graph.

In Figure 2.3(b), we show the total number of nodes, vertices, maximal cliques and triangles for subgraphs SG_n, for $6 \leq n \leq 10$. There are roughly three orders of magnitude in the drop of the number of maximal cliques from SG_6 to SG_7: there are more than 16 million in SG_6 and 11,546 maximal cliques in SG_7. Thus, we select SG_n for $n \geq 7$ as good places to look for social structure, since SG_6 is very large and dense. Note that finding the maximal cliques in subgraphs with $n \leq 5$ was computational expensive due to the size of these graphs. In addition, we are more interested in node pairs with high collaboration intensity anyway. Recall that there are no collaboration edges in the graphs with $n > 10$.

Most users (82%) participate in less than 10 cliques in SG_7. In Figure 2.3(c), we plot the Cumulative Distribution Function of cliques and triangles of user in SG_7. We see that only 18% of users have more than 10 cliques and there are 21 users that participate in more than 5,773 cliques, which is 50% of total cliques in SG_7. Note that the distribution of cliques starts from 1, but the distribution of triangles from 0: a pair of users each with a degree of one constitutes a trivial clique, but has zero triangles.

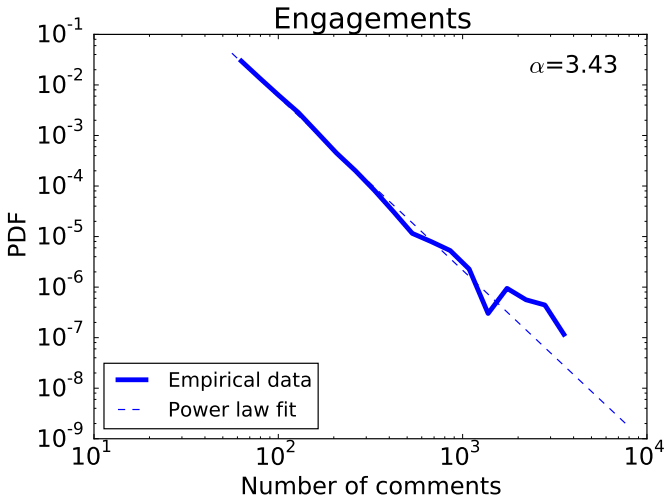


Figure 2.4: **Engagement properties.** Distribution of comments per engagement.

2.2.2 User-article engagement properties

An engagement refers to the interaction of a user with a given article for which she has left at least one comment. We consider two attributes for an engagement: (a) **engagement duration**, which is the amount of time interval between the first and last comments of the user, and (b) **engagement intensity**, which is the number of comments that the user left for that article.

Figure 2.4 shows the skewed distribution of comments per engagement, which seems to follow a powerlaw with high accuracy and a slope of 3.43. We observe that: a) 94% of the engagements contain fewer than 10 comments, b) only 0.2% of engagements contain more than 50 comments, and c) there are 20 engagements with more than *1000* comments which is a surprisingly high number.

Duration and intensity of engagement are not correlated. Intuitively,

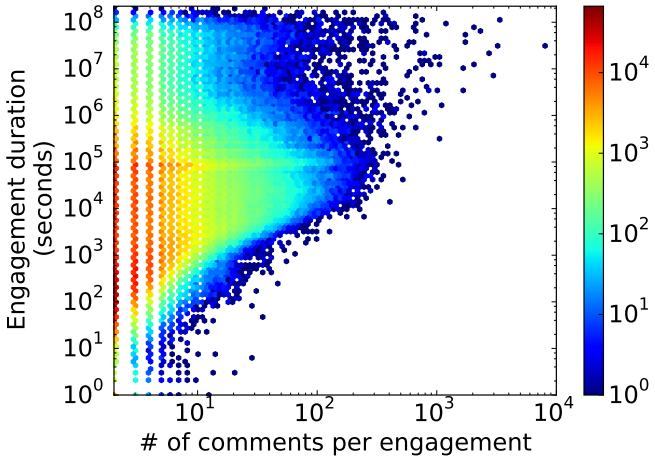
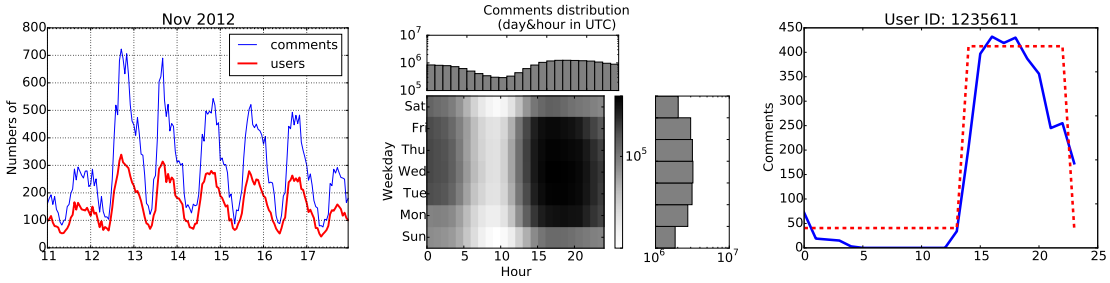


Figure 2.5: **Engagement properties (cont.).** The heatmap of the engagement duration versus intensity.

one would have expected that the duration and the number of comments per engagement should be correlated, but we find that this is not the case. Figure 2.5 depicts the heatmap that captures the relationship between the duration of an engagement and the number of comments posted by a single user in that engagement. We find that only 1.4% engagements which last more than an year have more than 100 comments while 74% have only less than 10 comments. Engagements are tend to be short lived, 78.9% of engagements end in 10 minutes and only 4.2% last more than one day.

2.2.3 Temporal properties of user behavior

Users exhibit persistent behavior in their daily and weekly behavior. We observe a strong periodicity in the temporal distribution of the number of comments as well as the number of users that make comments on Disqus. This suggests that the behaviors of a user exhibit reasonably stable temporal patterns. From the D_{Comm} data set,



(a) Hourly activity over a week

(b) Day-time plot

(c) Example of downtime (in UTC)

Figure 2.6: **Temporal properties.** (a) The hourly number of comments of Disqus starting on Sunday Nov 11-th. (b) Date-time plot for comments of Disqus shown as a heatmap. The color of each square represents the number of comments for a given time (x-axis) fora given day of the week (y-axis). (c) The aggregate number of comments per time of day for a single user and showing the downtime as calculated by our algorithm.

we plot the number of comments and the number of unique users (identified by their IDs) at two different levels: (a) hourly activity over a week, and (b) the heatmap of daily and hourly activity in Figure 2.6. We observe a strong periodicity in both plot. Although we only show plots generated from slices of 2012 data, we have observed the same patterns in the data for other years (2009 to 2014) in our D_{Comm} data set.

Interestingly, Figure 2.6(a) shows that the commenting activities begin to rise at the start of the week, peak in the middle, and drop sharply afterwards. This suggests that for a typical Disqus user, they post the most content toward the middle of a typical *workweek*. In Figure 2.6(b), we show the histogram of all comments in the D_{Comm} data set in terms of hour-of-day (X-axis) and day-of-week (Y-axis) as a heatmap. We see that daily commenting activity is more intense on weekdays, compared to weekends. Interestingly, the drop in weekends is also observed in the posting on **Facebook** [69] and **Twitter** [65].

Detecting the "downtimes" of users. A key temporal property is the time

that the user spends at the platform, namely the uptime and downtime of the user. Knowing the downtime helps us develop user profiles in § 2.3.

We describe, DownTimeFinder, the algorithm we use to detect a user's downtime period. Defining downtime for a user is a challenge as there is noise and variations among users. We are exploiting two facts from our previous analysis: (a) some users interact heavily with the platform, and (b) most users have periodic and predictable behaviors. Intuitively, we define a person's downtime as the *largest* period of time (a few hours in length at least) during the day when the person's commenting activity is significantly lower compared to the rest of the activities on that day. An example can be seen in Figure 2.6(c), which shows the commenting activity of one user expressed as a time series. In this figure, the time interval from hour 0 to hour 13 can potentially be the user's downtime.

DownTimeFinder Algorithm: The DownTimeFinder algorithm takes in a time series for a user as input and outputs a time range when the user is the least active. The algorithm assumes that the downtime of a user is a straight sequence of hours without any gap. With this assumption, activity time series of user can be modeled by a square wave with exactly one *high* and one *low* segment. However, even after this simplifying assumption, the algorithm needs to find the starting time and the duration of the low segment of the square wave.

A naive algorithm to find the best starting time and duration is to try every combination of the two quantities and pick the best combination that fits the corresponding square wave against the time series. We use the simplest and the most effective goodness-of-fit measure: *sum of squared error*. However, this approach is computationally expensive

Algorithm 1 *DownTimeFinder*(TS)

Require: $TS \leftarrow$ an aggregate time series of daily activity

Ensure: Output downtime, a range of ours of inactivity

1: $TS \leftarrow$ append TS with TS

2: **for** $d \leftarrow 1$ to 20 hours **do**

3: $q \leftarrow$ a square wave of d hours of low segment and 24-d hours of high segment

4: Search the best fit of q in TS

5: **if** error of the best fit $<$ global-best **then**

6: Update the global-best

7: **return** the global best

for millions of users that a commenting service hosts.

To speed-up the process, we exploit the state-of-the-art similarity search technique for time series data [55] to reduce the computation. We formulate the problem as a similarity search problem as shown in the Algorithm 1. The algorithm start with self appending the time series. It then loops over possible durations. For each duration, the algorithm the similarity search function [55] as a subroutine to identify the best starting time. The similarity search algorithm exploits the overlap between successive slides of the square wave and finds the best match in only $O(n \log n)$ time where n is the length of the time series. The Algorithm 1 is fast enough to process 26K users in just few minutes. By contrast, the naive approach would take roughly six hours, several orders of magnitude slower.

A straightforward implementation of the algorithm can be computationally expensive for large numbers of users, considering the fact that the algorithm enumerates thousands

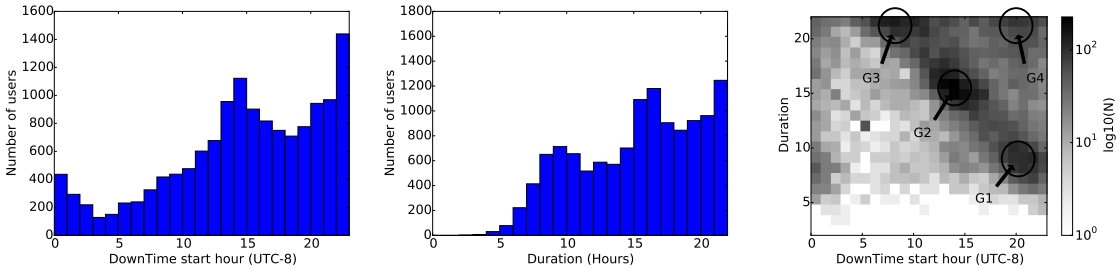
of combinations of start time and duration for each of them. We use a fast similarity search algorithm for time series data to reduce computation. The algorithm searches for the best start time for every duration. To evaluate all start time, the algorithm concatenates the daily time series with itself to allow for bathtubs in other time zones. The algorithm performs a similarity search under Euclidean distance to spot the most similar occurrence of the square wave in the concatenated time series [55]. The similarity search algorithm is $O(n \log n)$ in time complexity, which is sufficient enough to perform our Bathtub algorithm for 29K users. Note that, the naive implementation did not finish in four days with our dataset when we test it.

For reproducibility, an important detail is that the similarity search algorithm ignores the offset and scale of the time series by z-normalizing the activity time series for every user. Such normalization helps us compare different fits by the normalized error. A perfect fit of a normalized square wave ensures that the difference between activities in the downtime and uptime is statistically significant.

We also consider using more than one down times. This introduces two additional loops before line 2 in the Algorithm 1. One to iterate over the duration of the newly added downtime and the other is to model the gap between the two downtimes. Clearly, the model is more complex and is expected to reduce errors from the simpler fits. We test this complex model in the next section before building more complex models with variable activity levels in the two downtimes as well as in the uptimes.

We have also modified the algorithm to fit more than one downtime, but we find that the improvement in the approximation accuracy does not justify the added complexity.

In more detail, the process is is very similar but the complexity is increased compared to fitting for a single downtime. Each downtime period has its own start time and duration, in addition, we will need two factors to define the relative up-level and down-level of the two downtimes. Fitting two downtimes can reduce the sum of squared errors for only a few users, which is not sufficiently significant.



(a) (b) (c)

Figure 2.7: **DownTimeFinder**. Distributions of (a) Downtime start hours, (b) Downtime durations, and (c) both start housr and durations expressed as a heatmap (in grayscale to show the distinct "peaks")

Observations. Using the DownTimeFinder algorithm, we extract: (i) downtime start hour and (ii) downtime duration from each of the time series in $D^{\boldsymbol{t}}_{\text{Comm}}$. Figures 2.7(a) and 2.7(b) show the distribution for (i) and (ii) respectively while Figure 2.7(c) represents the joint distribution of both (i) and (ii) in the form of a heatmap. Even though the original data is in UTC, we shifted the data to UTC-8 (Pacific Time Zone) for this section.

From Figure 2.7(b), we see there are three dominant peaks, so we identify three groups of users, and we attempt to interpret their behavior assuming that they are US-based. In our analysis, we actually verify that this is the case for the majority of these users. Clearly for non US-based users, our interpretation will need to be corrected.

a) One group with roughly 3 hours of "up time" (21 hours of downtime). These users seem to post their comments during their breaks at work.

b) One group with roughly 9 hours of "up time" (15 hours of downtime). These users seem to post their comments throughout their day at work and stop afterwards.

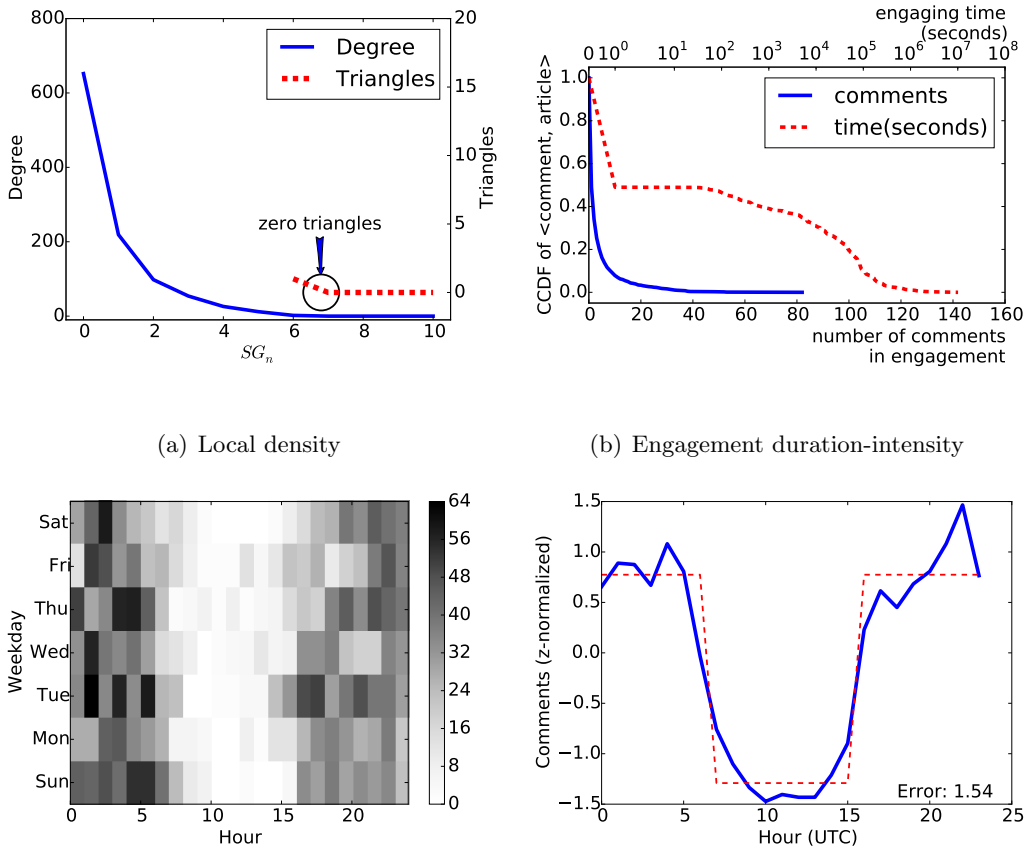
c) One group with roughly 15 hours of "up time" (9 hours of downtime). These users are highly active and therefore also highly suspicious, as they seem to spend the majority of their time making comments.

Furthermore, Figure 2.7(a) shows that there are two dominant peaks for downtime start hours at hours 14 and 23 (UTC-8), which would translates respectively to 2pm and 11pm on the United States West Coast. When we cross-reference the two peaks with the heatmap in Figure 2.7(c) (the dark spots labeled G2 and G1 in the Figure), we find that they are associated with the durations of 15 and 9 hours of downtime (or 9 and 15 hours of "up time") respectively.

2.3 Identifying parasitic behaviors

We develop user profiles using the features and observations in the previous section. We focus on features that capture social, engagement, and temporal properties of users as shown in figure 2.8. Our work here is the first step towards techniques that can automatically search for suspicious users, which we discuss in the next chapter. Here we provide a useful set of features and the intuition that a detection mechanism could leverage.

Profile of a benign user: features and plots. We selected and created these



(a) Local density

(b) Engagement duration-intensity

(c) Date-time plot

(d) Downtime plot

Figure 2.8: An example of a typical user profile. (ID:1164515)

four plots to visually capture key properties and we show how we can identify unusual users, as explained below.

First, we capture the user-user interactions with the **local density plot**, shown in Figure 2.8(a). The plot shows the number of collaborators of a user and the number of triangles it forms with collaborators in each SG_n. The higher values of n, the higher the collaboration intensity between a user and its neighbors. A typical user exhibits moderate local density, and SG_7 is the first index for which SG_n has zero triangles, as highlighted

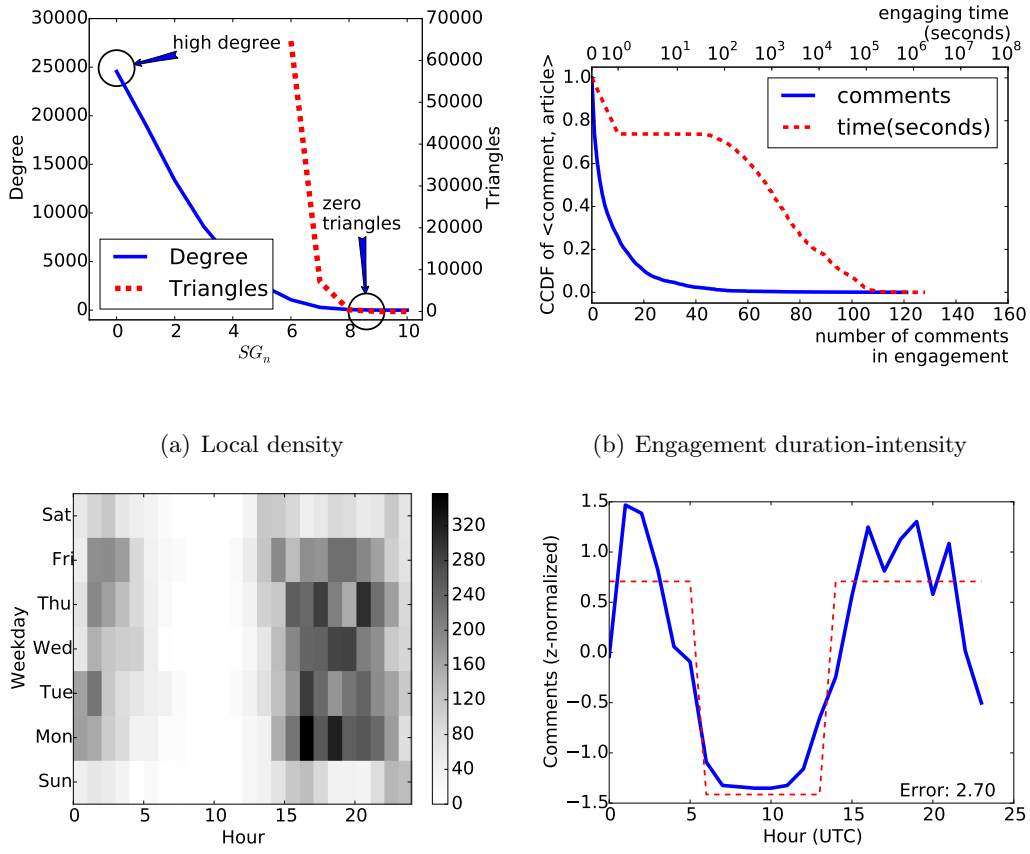
in the figure. Here, we study the social density of a node for each of the subgraphs SG_n starting from $n = 10$ and decrease n until SG_n becomes too large to analyze.

Second, the **engagement duration-intensity plot**, shown in Figure 2.8(b), depicts the distribution of the number of comments (x-axis bottom scale and solid line) and the distribution of the duration times (x-axis top scale and dotted line). For a typical user, the engagements tend to involve few comments per engagement, whose distribution follows a smooth exponential distribution, as seen in Figure 2.8(b).

Finally, we capture temporal properties using the **Date-time plot**, shown in Figure 2.8(c) and the **Downtime plot** of the user, shown in Figure 2.8(d), as discussed earlier. A typical user has a well-defined downtime shape, and pronounced regularity of commenting per day-of-week and hour-of-day.

a. "Colluding" users: (Profile: high local density). In Figure 2.9(a), we plot a colluding user: the user is tightly collaborating with a group with the most likely goal of promoting a particular idea. The tell-tale signs are captured in the social density: high degree of connections, and high local density as indicated by the number of triangles in its neighborhood. We discuss how we select this node below.

In § 2.2, we presented circumstantial evidence that implies that some users in Figure 2.3(d) are organized and work together for a shared purpose. For example, users joined the Disqus network on the same day, and comment mostly on a particular news site. Recall that in the SG_{10}, we have 8 connected components with collaborations intensity of more than 2^{10}. To investigate further, we study how the size of their cliques from SG_{10} to SG_7, which increases for lower values of n. We find two large cliques of interest: one consists



(a) Local density

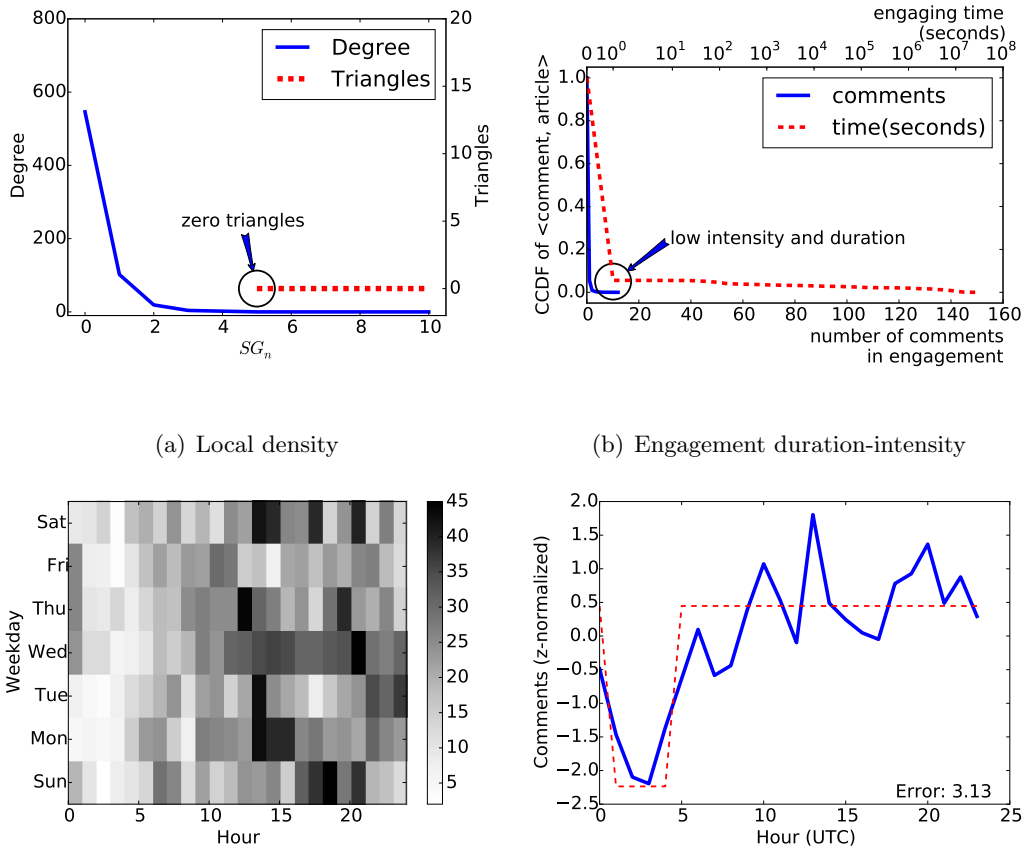
(b) Engagement duration-intensity

(c) Date-time plot

(d) Downtime plot

Figure 2.9: "Colluding" user: profile of an account that engages in "Colluding" activities. (ID:1742861)

of 34 users whose usernames start with **cnn-**, which was not present in SG_{10}, and second one consists of 29 users whose usernames start **newser-**. We randomly sample 500 pairs of comments from each clique which their users made on the same article. We manually validate that *at least* 86% of pairs from the **newser** clique, and *at least* 73% of the pairs in **cnn** clique are supportive of each other. The percentage could be higher, as we only report the pairs that the agreement was beyond doubt. Since most of users in **cnn** clique have very similar profiles, we randomly select one and show in Figure 2.9.



(a) Local density

(b) Engagement duration-intensity

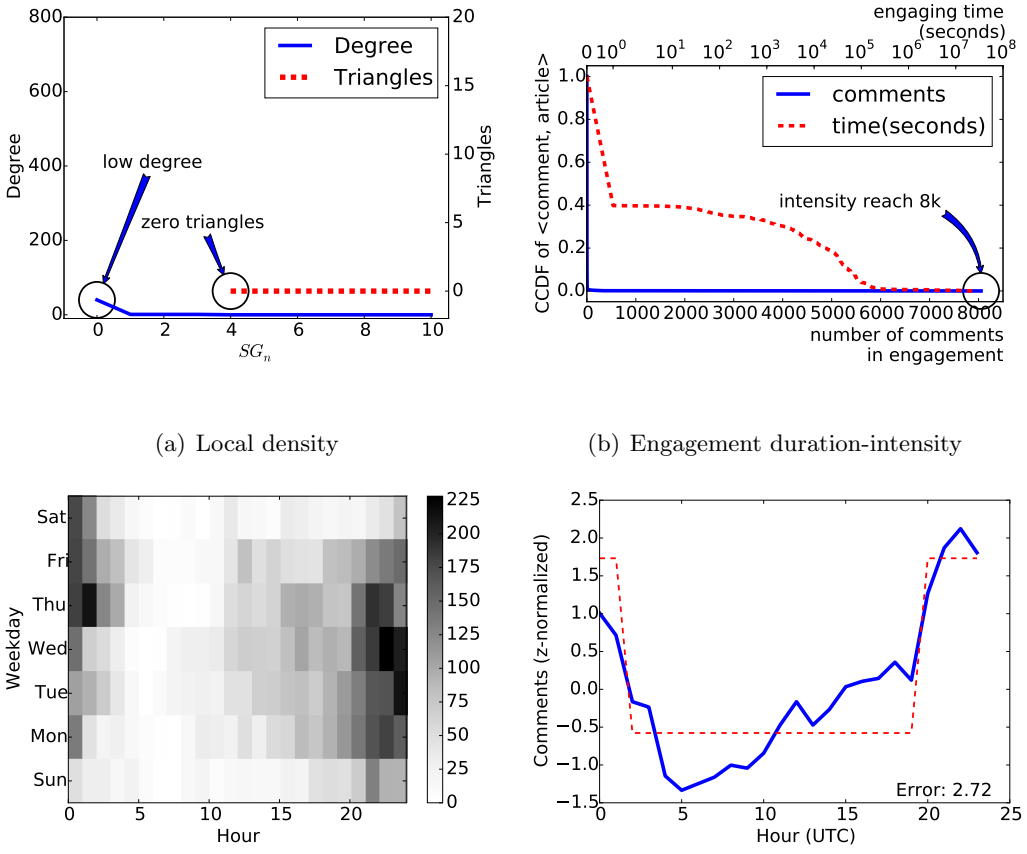
(c) Date-time plot

(d) Downtime plot

Figure 2.10: "Hit-n-away" user: profile of a user that engages in trolling and spamming. (ID:1026110)

"Hit-n-away" users, spamming and trolling: (Profile: low intensity of engagement, less regular temporal patterns). We identify users who post a large number of comments, but each comment is at a *different* article. We discover these users by focusing on engagements which have low intensity and then we examine the top 100 users with the highest number of such engagements. We find 27 users that are engaged in spamming: they keep posting the same comment or URL. We also find 11 users that exhibit behavior consistent with "trolling". Trolls attempts to incite others through insults, nonsensical,

or outrageous comments. These users (i) do not have any lengthy engagements, as shown in Figure 2.10(d), and (ii) tend to have less regular day and time patterns, as shown in Figure 2.10(c). Interestingly, two of those users posted thousands of (offensive) comments and each one was never more than two or three words long.



(a) Local density

(b) Engagement duration-intensity

(c) Date-time plot

(d) Downtime plot

Figure 2.11: "Chat room" user: profile of an account that engages in "chat room" activities. (ID:1204188)

"Chat room" users: (Profile: extremely intense and long lived engagement). In Figure 2.5, we see that there are outliers with engagements that can last for months and containing upwards of 500 comments. This is unusual, since 90% of all engagements last

less than one day and contain fewer than 10 comments. Upon further investigation, we find a total of 53 engagements that exhibit extremely high intensity (at least 500 comments each) and longevity (lasting at least one month). These engagements are conducted by 24 unique user accounts on 19 different websites. It turns out that these 19 websites seem to be using Disqus as their chat room. These websites create an "empty" article with no content and its sole purpose is to facilitate the discussion between a group of people, say an organization. Though not illegal, this is a surprising use of Disqus. For example, the user in Figure 2.11 spends almost all their nearly 8,000 comments discussing video games.

The profile of one of the 24 suspicious users is shown in Figure 2.11. We see in Figure 2.11(a) that their local density is low and the degree has a sharp decrease from SG_0 to SG_1 followed by a straight line to SG_{10}. Besides, their engagement intensity is not typical: it visually looks like a straight line and it extends to 8,000 comments over a period of slightly more than one year.

2.4 Related Work

We group previous work into five areas of research, which we summarize below.

Detecting malicious behaviors. Many studies detect anti-social behavior [20] [19], bots [22] [54] or cyber-bullying [38] but they focus on OSNs. A recent work [20] analyzes the comments on three websites using Disqus, but the focus is to predict if a user will be banned due to misbehavior, which is somewhat different from ours: (a) some misbehaviors are not necessarily reported, as we saw, and (b) we have a fine-grained definition of misbehavior.

Modeling and understanding online user behavior. Several works study temporal and behavioral patterns of OSN users [32] [28] [37] and personal traits [43]. Some recent studies also use mobile phones and OSNs to profile users' psychological states [68] [26].

Geolocation. Many works focus on inferring the geographic locations of users of OSNs by analyzing their posts [34] [35] [21], their behavioral activity over time [47] [46] or their social/spatial proximity to other users [17] [25] [40].

Content analysis. Many studies leverage textual analysis to detect spammers on OSNs [60] and blogs [52] or how to achieve the same objective by analyzing the users's behavioral patterns [64]. We focus on behavioral properties for discovering unusual user.

Identifying users' activities pattern. Some other works have developed techniques to estimate the downtime and uptime of users [32], and we intend to compare our approach with those in the future.

Chapter 3

TrollSpot: Detecting misbehavior in commenting platforms

Any successful medium eventually attracts abusive behaviors, and commenting platforms is no exception. Over the last decade, commenting on news articles has emerged as a new form of highly social interaction. First, a small number of companies facilitate the backend management of comments for a wide range of websites. We use the term **commenting platform** to refer to such platforms, which include Disqus [30], LiveFyre [45], and IntenseDebate [27]. Second, commenting is an intense activity for many users, who can spend many hours daily at it.

We list a set of definitions that we use in this chapter. A **user** is defined by a platform account, which enables her to leave comments to articles on a website that uses the commenting platform. A user may leave more than one comment for an article, which leads us to define the **engagement** of a user for that article. An engagement has a time

duration and intensity in terms of number of comments. In lack of a better term, when two users comment on the same article, we say that they **collaborate** and we use the term **collaboration** to describe this activity. We use the term **collaboration intensity** to refer to the number of articles for which two users collaborate.

The key question in this chapter is: *Can we automatically detect malicious users in these commenting platform?* Specifically, we frame the problem as follows. The input is the commenting information of the users. This includes: the author of the comment, the time it was posted, and information on the article it was posted for. The goal is to identify malicious behaviors and users. Detecting parasitic and abusive behaviors is a critical building block for ensuring that these platforms serve their primary purpose, which is the honest and safe exchange of opinions among readers.

Commenting platforms have attracted little attention so far with only few exceptions [20] [19]. Most work on modeling and misbehavior detection focuses on Online Social Networks (OSNs), and blogs. In § 3.5, we survey the key related areas including: (a) detecting abusive behaviors and malware propagation in OSNs [22] [54]; (b) modeling online user behavior [32] [28] [37]; and (c) analyzing text of online users [60] [52] [64].

We propose a systematic comprehensive methodology to identify malicious users on commenting platforms to enable: (a) interpretable, and (b) fine-grained classification of malicious behavior. We claim four key novelties in our work.

a. A behavior-based classification. We propose two classification methods, one of which introduces a two-stage classification approach. In this method, we map: (a) observable features into behaviors, and (b) behaviors into user roles, using unsupervised

and supervised learning respectively.

b. A comprehensive multidimensional feature set. We combine 73 features from four different dimensions of user interactions: (a) social interaction or user-user interaction, (b) engagement or user-article interaction, (c) temporal features, and (d) linguistic (text-based) features.

c. Fine-grained malicious role identification. Our approach goes beyond a good versus bad determination to a more fined-grained classification of misbehaving roles. Here, we focus on three roles: (a) spammers, (b) trolls, and (c) fanatics, which are defined in the next section. However, it is easy to introduce more roles as long as appropriate ground truth is available.

d. Interpretable classification. The results of our approach are interpretable, since they are behavior-centric. A system administrator can better understand or tweak the definition of, say, a spammer, by looking at the behaviors that constitute its definition (e.g. high number of repeated text across articles, low engagement per article).

Our study is grounded on nearly 7 million comments from nearly 200K users over 9 months from Discus, which is arguably the largest commenting platform. We highlight some interesting result from our study.

a. Identifying misbehaving users with 0.904 AUC. Our classifier can efficiently identify misbehaving users and it outperforms the baseline we compare with.

b. The role classification result achieves 80.8% overall accuracy.

c. **We find patterns of misbehaving users from their interpretable latent behavior.** An example latent behavior indicates that making longer comments and using

capital letters more frequently is a sign of misbehaving users, and the later the active hour of a users, the more likely they are to misbehave.

d. Large scale study. About 0.9% of total users in our data-set are misbehaving users, and CNBC has the largest percentage of misbehaving users which is 1.48%. Only 15% of the misbehaving users exhibit a cross-website behavior.

3.1 Data collection and definitions

Disqus is one of the most widely-used commenting service provider today with a billion unique visitors a month and installed by more than two and a half million sites, including ABC News, Rolling Stone, IGN, Bloomberg and more.

Data Collection. We collected data from Disqus through its Application Programming Interface (API). The API can be used to collect all the comments for a given article, and for all the articles of a given website. However, the functionality of collecting all the comments for a given user would have been useful for our study, but it became unavailable in 2014 for privacy reasons.

Using the website-centric API, we collect data from four popular websites: (a) CNBC News, (b) ABC News, (c) Bloomberg Views and (d) Breaking News - a Disqus channel. The first three are well-known news websites and the last one is the most popular channel on Disqus. A channel is similar to a newsfeed, whose articles are selected by the users that participate in that channel. We collect all comments posted at articles published on these four sources in between November 1st 2015 and July 31st 2016. The whole dataset consists of: (a) 286,275 articles, (b) 6,994,693 comments, (c) 201,112 unique users, and (d)

1,705,667 engagements.

Establishing the ground truth of misbehaving users. A key challenge in our work is the need for ground truth of misbehaving users. Fortunately, Disqus enables users to report misbehaving comments and the number of reports a comment received is provided by Disqus API. In our 7M comments, we have 66.4k comments reported as "bad/inappropriate" by the community. We will further discuss this in § 3.3 and show how we leverage this to identify misbehaving users.

Roles of misbehaving users and ground truth. We identify and attempt to detect three different roles of misbehaving users: trolls, spammers and fanatics. These are inherently difficult to define, so we resort to human feedback.

We define three malicious roles below. We show here the definitions that we gave our evaluators, as we explain in § 3.3.

1. **Trolls.** Users who make inflammatory or inappropriate comments for the sole purpose of upsetting other users and provoking a response.

2. **Spammers.** Users who repeatedly make similar comments in the same or multiple articles.

3. **Fanatics.** Users who exhibits an extreme and uncritical enthusiasm in religion or politics.

Table 3.1: The overview of the 73 features along each Dimension

Dimension	Count	Features
Engagement	7	number of engagements, engagement duration, engagement intensity
Social	17	degrees, number of maximal cliques, number of triangles (in different level of collaboration intensity)
Temporal	25	number of comments made in 24-hour slots, highly-active hour
Linguistic	24	number of words, number of sentences, percentage of capital letters, readability metrics, number of URLs

3.2 Features and user behavior

In this section, we identify and study features that relate to user behavior on the Disqus platform. Although studying each of these features in depth is of independent interest, the goal is to identify meaningful features that can help us detect misbehavior. In Table 3.1, we outline the features that we use in § 3.3.

We study the behavior of users along four dimensions: **(a)** engagement behavior (user-article interaction), **(b)** social behavior (user-user interaction), **(c)** temporal behavior, and **(d)** linguistic properties. The first three dimensions aim on capturing users commenting behavior and the last one focus on the text they use.

3.2.1 Engagement behavior

We quantify the engagement (user - article interaction) with 7 different features. Six of them are derived from two major properties: The **engagement duration** is the time

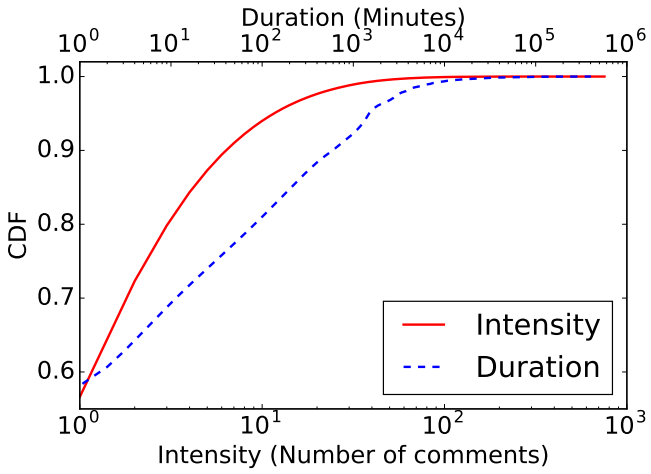


Figure 3.1: **Engagement behavior:** Distribution of intensity and duration of engagements.

interval between the first comment and the last comment user makes on the article. If the user leaves only one comment, we consider this as zero length interval. The **engagement intensity** is the total number of comments user makes on the article.

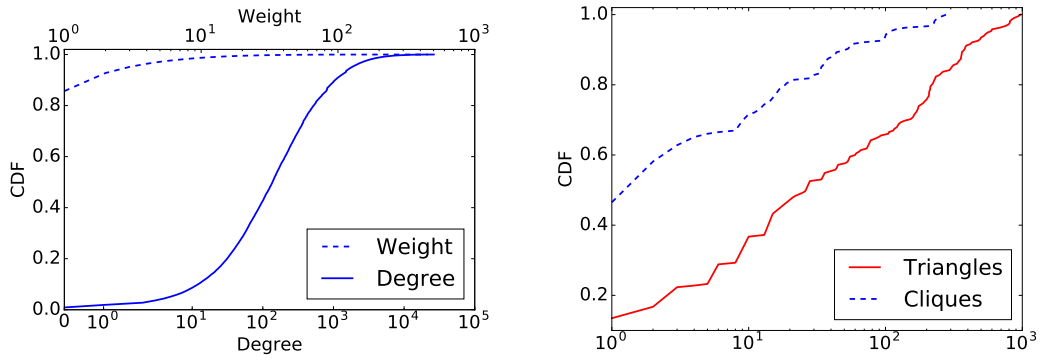
Engagement duration: 90% last for less than 10 hours, but some can be as long as half an year. In Figure 3.1, we plot the Cumulative Distribution Function (CDF) of the duration (top x-axis) for all 1.7M engagements. We find that 90% of engagements last less than 10 hours and have less than 7 comments. Interestingly, we find 106 engagements which last for more than half year!

Engagement intensity: 90% have less than 7 comments, but 0.06% have more than 100 comments. In Figure 3.1, we plot the CDF of the intensity (bottom x-axis) for all engagements. We find that 90% of them have less than 7 comments, while 1,151 (0.06%) of them have more than 100 comments.

High intensity is correlated with misbehavior. The observations above

raise the question: is unusually high engagement intensity associated with misbehavior? Preliminary manual inspection suggests that these engagements contain many reported (by the community) comments. We quantify the correlation between engagement intensity and the total number of reported comments in the engagement and we find a Pearson correlation coefficient $\rho = 0.53$. This suggests that engagement intensity should be a good metric in our classification.

3.2.2 Social behavior



(a) (b)

Figure 3.2: **Social behavior:** (a) the CDF of the user degrees (bottom x-axis) and the edge weight distributions (top x-axis), (b) CDF of the number of cliques and triangles of user in the G_{128} collaboration graph.

We propose 17 features to model the social interaction of the users, which we define as the posting of comments to the same articles.

Single-article collaboration Threshold: θ comments. We say that two users collaborate in one article, if they each post at least θ comments on that article. For $\theta = 1$, the graph become very dense, and the analysis is both slow and less informative. In the

remaining of this work, we use a threshold $\theta = 2$.

User-user collaboration intensity and threshold: λ articles. The collaboration intensity is the number of articles that two users collaborate for a given threshold θ. To study collaborations at difference levels of intensity, we introduce the **collaboration intensity threshold** λ, which we use below.

We define the undirected weighted **collaboration graph** $G_\lambda = \langle V_\lambda, E_\lambda \rangle$ of collaboration intensity λ where:

1. V_λ is a set nodes v, representing users.

2. E_λ is the set of edges, where edge e_{ij} between nodes v_i and v_j exists, if and only if the collaboration intensity of the users exceeds the **threshold of λ articles**. The edge weight $w(e_{ij})$ is set to the collaboration intensity.

The collaboration graph (for $\theta = 2$ and $\lambda = 0$) has 95,527 users and roughly 21 millions edges, an average degree of 440.7 and a median degree of 137. Note that we do not include users with zero degree in this graph.

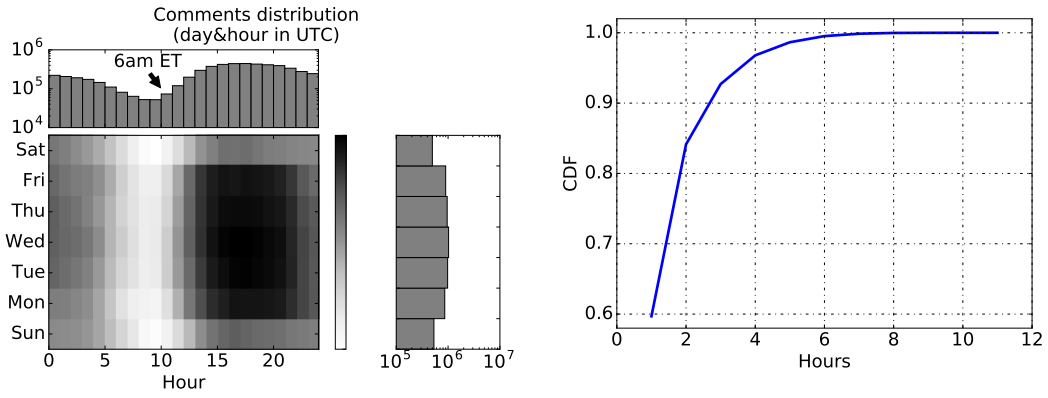
In Figure 3.2(a), we plot the CDF of the user degrees (bottom x-axis) and the edge weight distributions (top x-axis). We see that 90% of the users have degree lower than 1,054, the max degree goes to 26,318: this user collaborates with more than 27% of users in the graph! The figure also shows that 90% of the edges have weights less than 2. Although it is not easy to gauge from the plot, we find 12,374 edges with weight over 64, 1,779 edges with weight over 128 and 65 edges with weight over 256. In other words, there are 65 pairs of users who have collaborated on more than 256 articles.

Capturing the collaboration groups: triangles and cliques. We quantify

the local connectivity of the users using the number of maximal cliques and triangles in which a user participates. Both these metrics capture how densely connected the neighbors of that user are. Figure 3.2(b) shows the distribution of triangles and maximal cliques of user in graph, for λ equals to 128. We do not count "trivial" cliques of size two. We see that 50% of users have less than 27 triangles on the neighborhood, 90% of have less than 376, but there are 1% with more than 868 triangles. We also see that 90% of users have less than 51 maximal cliques and there are 8 users, who participate in more than 186 cliques which is more than 50% of all maximal cliques in the graph. These highly collaborative users are very suspicious and this observation encouraged us to consider both these features for misbehavior detection.

3.2.3 Temporal behavior

Most users exhibit persistent behavior: daily and weekly. We use the **time and day of the week** plot to understand the temporal behavior of the users. The plot shows the number of comments in each hour-of-day and day-of-week as a heat-map. Figure 3.3(a) shows this behavior for all the users, but individual users exhibit similar daily and weekly behavior, as we will see below. On the x-axis (hour-of-day), we see that the commenting activities increase at the beginning of day in the east coast as annotated in the plot, the peak hours cross the noon period of both east and west coasts then drop sharply after that. Given the news websites we are studying, it is reasonable to assume a US-centric user majority. In y-axis (day-of-week), the activity increases at the start of the week, peaks in the middle, and drops before the weekend. Combining the two observations, we see that Disqus users post most of their comments during the common work-hours in the week.



(a) (b)

Figure 3.3: **Temporal behavior.** (a) Time and day of the week plot. (b) The distribution of highly-active hours of users.

Note that the same pattern is also observed on other social networks, like Facebook [69] and Twitter [65].

The plot lead us to the **Hypothesis:** A typical user makes most of her comments in a period of 3-4 hours in a day.

The highly-active hours of users commenting behavior. To assess the validity of our hypothesis, we would like to answer the question: how many hours a day does a user spend commenting? Determining this hides subtleties as: (a) patterns can vary from day to day, and (b) it could be that many hours have non-zero activity due to averaging over a long period. Thus, we opt to capture how "focused" or "spread" is the commenting activity of the user. We define **highly-active hours** to be the minimum number of hours, during which the user makes more than 50% of their total comments during a day on average.

Highly-active hours is less than 4 hours for 96.7% of the users. The

distribution of the users' highly-active hours is shown in Figure 3.3(b). The plot shows that 96.7% of users have highly-active hours less than four hours, which matches our hypothesis. Interestingly, we find 368 users who have more than 8 highly-active hours, a significantly wider spread. Upon inspection, many of these users have non-trivial activity over 14 hours in a day. This wide range of behaviors suggests that highly-active hours can be a useful metric for our classification.

3.2.4 Linguistic properties

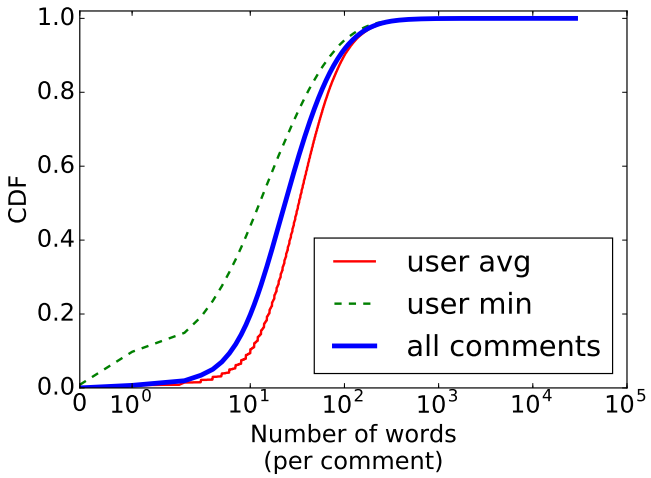


Figure 3.4: **Linguistic property**: number of words in a comment

We identify and study 24 linguistic features from the text of the posts, such as the length of the comment, several metrics readability, number of URLs, but we only discuss two interesting metrics here.

Only 1.9% of comments contain URLs. We naturally consider the existence of URLs in a comment as a feature in our classification: their presence can indicate ad-

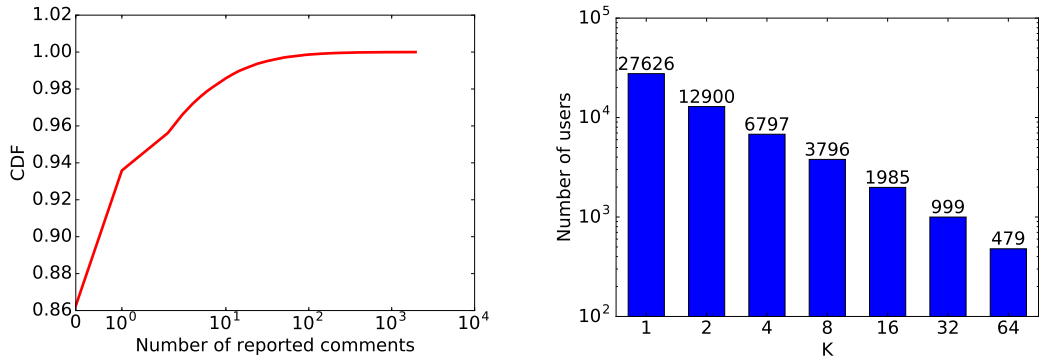
oriented spamming. We find that only 1.9% of all comments contain one or more URLs. We also find that only 1.7% of the users have ever posted more than 3 comments containing URLs. In fact, our interaction with the data suggests that often users will use URLs as references in support of their opinions.

Length: 91% of comments have less than 100 words. The length of a comment is a good metric for the interest and the time a user is willing to spend expressing her opinion about the article. In Figure 3.4, we plot the distribution of the number of words in a comment in three different ways: **(a)** across all comments, **(b)** the average comment length per user, and **(c)** minimum comment length per user. We see that 91% of comments have less than 100 words, while roughly 20% of the comments with less than 10 words. Interestingly, we also find 14,838 (0.002%) comments with more than 500 words, which is roughly equivalent to 1.5 pages (per wordstopages.com with Times New Roman, 12 font-size, single-spacing): the comment is the size of small article!

Lengthy comments are more likely to be spam. We examine the lengthy comments (over 500 words) and find out that 47% of them are verbatim copies of at least one other comment from the same user in our dataset. This is an indication of spamming especially in its broader definition that we adopt here. Such phenomenon is more pronounced for higher word counts: 63% of comments have verbatim copies for length over 1,000 words , and 83% of comments with length 2,000 words or longer. This suggests that comment length is a helpful feature in detecting misbehavior.

3.3 Feature-based misbehavior identification

Having identified interesting features, we develop a method to identify misbehaving users and their effectiveness. For convenience, we outline the features in Table 3.1.



(a) (b)

Figure 3.5: Study the "proxy-signal": (a) Distribution of reported comments of a user. (b) Number of users having k reported comments.

A. Establishing the ground truth. To overcome the lack of absolute truth, we rely on "proxy signals". Here, we use the community's own opinion: any user can report (a.k.a. flag) a comment as "inappropriate". Below, we explain how we use this community feedback to construct the ground truth, as the process hides several subtleties.

Ground-truth: Reportings per malicious comment. To increase our confidence, we set a minimum threshold of reports, ϕ, that a comment must have to labeled malicious. The rationale is that a single reporting can be created even accidentally (the authors have regrettably done this once). After analysis and deliberation omitted here, we settled on $\phi = 3$ reportings. We use the term **reported comment** to refer to a comment with more than ϕ reports.

Ground-truth: Reported comments per malicious user. In the same vain, we want to be careful in labeling a user as malicious based on the number of reported comments. We use the **reported comments threshold**, r, to control tune the definition of malicious user.

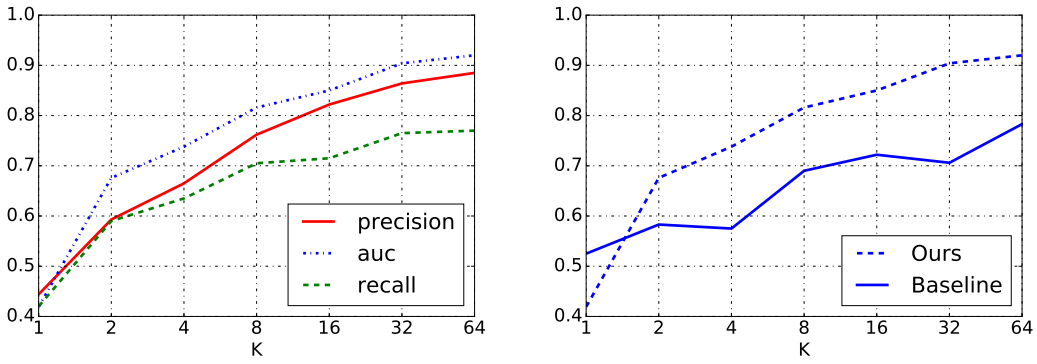
Roughly 86% of the users have no reported comments. We plot the distribution of the total number of reported comments for each user in Figure 3.5(a). We find that 86.2% of users have no reported comments, and only 1.6% of users have more than 10 reported comments.

We consider users with zero reported comments as **benign** for the purpose of establishing the ground truth.

Building the ground truth datasets: D_r. We create a set of labeled datasets as follows. First, we distinguish reported users into groups, $R_{r(i)}$, with $r(i) = 2^i, for\ \ i = 0, 1, ...$ A user is in group $R_{r(i)}$, if the number of her reported comments are greater or equal $r(i)$. It turns out that no user has more than 128 reported comments. The numbers of users in each group with threshold $r(i)$ are shown in Figure 3.5(b). Second, we create datasets $D_{r(i)}$ by randomly selecting 200 reported users from $R_{r(i)}$, and combining them with 200 benign users (zero reported comment).

Benign user selection: Minimizing the effect of the number of comments of a user. When selecting the 200 benign users to build the datasets, we follow the insightful approach of the earlier in this space [20]. Namely, we find benign users that "match" the number of comments of reported users. The goal is to minimize the dominant role that the number of comments of a user can play. First, the number of comments and

number of reported comments of a user are strongly positively correlated with a Pearson Correlation Coefficient of 0.73. Second, most users have few comments given the skewness of the distribution. Thus, a purely random selection of benign users would have probably created a low-activity benign set in terms of user comments, and a highly-active reported users. The classification would have been very accurate by relying heavily on the number of posts.



(a) (b)

Figure 3.6: The result of (a) Classification results as a function of the number of reported comments that "incriminate" a user. (b) AUC of our approach and the baseline.

B. The benign-malicious classification. For the classification, we use the Random Forest classifier provided by Weka [71], which gave the best results among many that we tried. We perform ten-fold cross validation and report the precision, recall, and **ROC curve (AUC)** of each dataset in Figure 3.6(a). The plot shows that our features can identify reported users with more than 80% precision when threshold $r \geq 16$.

Selecting reported comments threshold $r = 32$. We manually examined reported users in D_{16}, D_{32} and D_{64} by sampling 20 users from each group. We find that

users with more than 16 reported comments exhibit a persistent misbehavior throughout their lifetime. The 40 reported users sampled from D_{32} and D_{64} are 100% labeled as misbehaving users by our independent human evaluators. This gives us confidence to claim that users with 32 or more reported comments are misbehaving users. Thus, we use dataset D_{32} as reference below.

The baseline classifier. We adapt and use a previously proposed algorithm as a reference in our study [20]. That work focuses on predicting whether a user will get banned in the future, which is a somewhat different goal from our work.

We faithfully reproduce the reference method [20], which also uses the Random Forest technique like we do. In their work, they use 35 features, however 6 are not publicly available, namely the website moderator features, such as the number of up-votes given to others (similar to a Facebook "Like"). Unfortunately, we were not able to access those features through the public API, therefore we do not use them in our experiments.

The accuracy of our method exhibits 90% AUC. For clarity of presentation, we only show the comparison of AUC of the two methods in Figure 3.6(b). Our method performs better than the baseline classifier, with the caveat that both approaches are restricted to the publicly available features. The most interesting conclusion is that our features have good discriminatory power in classifying misbehaving and benign users.

C. The fine-grained malicious role classification. The novelty of the work relies partly on going beyond the "malicious" label to the role of the misbehaving user.

Ground truth for the roles of misbehaving users. Here, we resort to manual labeling to create our reference data: we examine 200 misbehaving users in D_{32} and

Table 3.2: Role classification result

Role	**Precision**	**Recall**
Trolls	86.4%	73.1%
Spammers	58.6%	81%
Fanatics	86.1%	73.3%
Benign	81%	87.5%
Overall accuracy	80.8%	

categorize them into three different roles: trolls, spammers and fanatics. To the best of our knowledge, this is the first work that does such a fine-grained classification of user behaviors, especially in commenting platforms.

Each user is labeled by three evaluators who were presented with the definitions in § 3.1. In absence of unanimity, we set their role by taking the majority label. Out of 200 misbehaving users, 104 are labeled as trolls, 21 as spammers and 75 as fanatics.

Role classification: Our approach has 80.8% overall accuracy. Having ground truth, we apply 10-fold cross validation with the same classifier and the same 73 features that we used to identify misbehaving users. Our result shows that our model can effectively classify the role of misbehaving users with an overall accuracy of 80.8% as shown in Table 3.2. For all the classes the recall is above 73% and the precision above 81% except the Spammers.

The community shows tolerance to non-provocative spammers. Intrigued

by the low precision for spammers, we find that spam comments without provocative language, swear words and sarcasm often do not get reported by the community. Recalling § 3.2.4, the unusually long comments are more likely to be spam, due to verbatim repetitions. However, this behaviors seems to either escape detection or be met with tolerance.

Our method identifies un-reported spammers. We examine the 12 false positive in the spam category: our spam label is not corroborated by the community. We actually find that at least one of these users exhibits clear spamming behavior, as she repeats the exact same comment 3 times in one article and 5 times in another. This investigation suggests that our approach could be catching parasitic behaviors that the community could miss. As a result, the accuracy of our algorithm could be better than reported here, especially for spammers.

D. Studying the misbehaving users in the wild. We apply the classier to the whole dataset to understand how misbehaving users distributed in the wild. For identifying misbehaving users, the classier labeled 1,738 (0.86%) users as misbehaving with confidence over 80%. The role classification results are: 866 users are labeled as trolls, 165 users are labeled as spammers and 597 users are labeled as fanatics. We also study how misbehaving users distributed in each websites, the result is shown in Table 3.3. It shows that CNBC has the largest number of misbehaving users with 1,167 which is 1.48% of total users active on the website while the rest of websites are at most 0.56% of users are misbehaving users. Again, we only label misbehaving users with confidence over 80%.

Table 3.3: Misbehaving users in websites

Website	**Total users**	**Misbehaving users**
ABC News	129,053	717 (0.56%)
CNBC	78,618	1167 (1.48%)
Bloomberg Views	33,223	131 (0.39%)
Breaking-News (Disqus channel)	11,607	48 (0.41%)

3.4 Interpretable two-stage classification

In this section, we introduce a novel, two-stage approach in classifying misbehaving users. In the first step we identify tightly-knit groups of users who exhibit very similar behavior across a certain subset of the features we discussed in the previous section. Each such group of users and their associated features defines a **latent user behavior**. Note that we do not define those latent user behaviors a-priori, they rather emerge from the data automatically. Subsequently, we use these latent user behaviors as our new features towards classifying misbehaving users. The advantage of this two-stage approach is that the latent user behaviors offer interpret-ability of the results and allow for in-depth analysis of different user (mis-)behaviors.

3.4.1 Identifying latent behaviors: Co-clustering

The first stage of our two-stage approach can be exactly mapped to an instance of *co-clustering*. In a nutshell, co-clustering is the simultaneous clustering of all data modalities. In our case, consider a representation of a user in the vector space defined by the

entire set of their features. Hence, the modalities of our data are two: "user" and "feature". Co-clustering of a users × features data matrix essentially yields different groups (or co-clusters), where each one contains a subset of the users and a subset of the features. In other words, when grouping users and features together, co-clustering allows for the flexibility of finding a set of users that are highly similar *only for a subset* of their features. This subset of features is key to our method. Essentially, the subset of features of a co-cluster defines a **latent user behavior**.

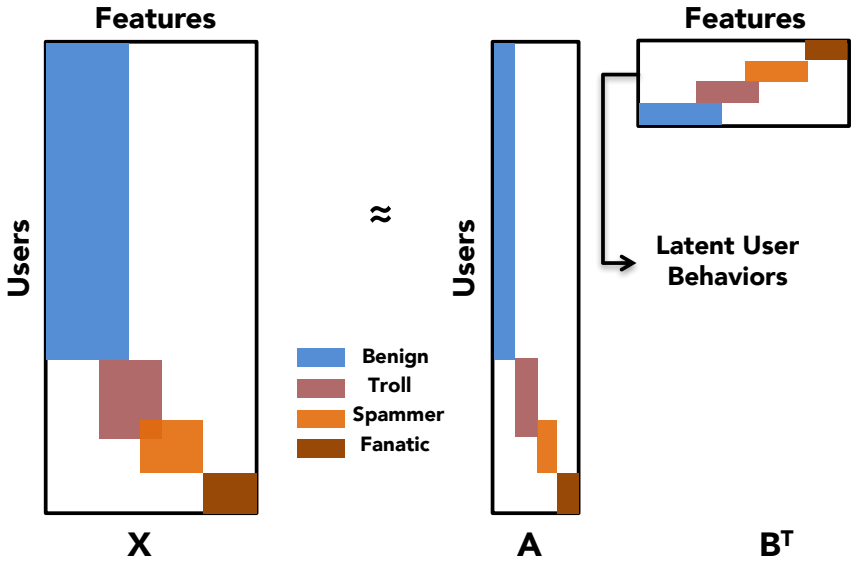


Figure 3.7: Co-clustering discovers latent user behaviors and the assignment of users and features to those behaviors.

An illustrative example is shown in Figure 3.7, where we have four distinct latent user behaviors: "Benign", "Troll", "Spammer", and "Fanatic". After properly rearranging the users and the features, in this simple example we see that each latent user behavior is forming a block within the matrix. Notice that blocks can overlap since: (a) in some

cases a user can be classified with different types of behavior, and (b) some features may be associated with multiple types of behavior. Mathematically, each block (formally co-cluster) in the data can be seen as (approximately) a rank-one matrix. By definition, a rank-one matrix can be written as the outer product of two vectors, i.e., $\mathbf{ab}^T$. We, thus, can use the following formulation for co-clustering, which was originally proposed in [57]:

$$\min_{\mathbf{A}\geq\mathbf{0},\mathbf{B}\geq\mathbf{0}} \|\mathbf{X} - \mathbf{A}\mathbf{B}^T\| + \lambda \sum_{i,r} |\mathbf{A}(i,r)| + \lambda \sum_{j,r} |\mathbf{B}(j,r)| \tag{3.1}$$

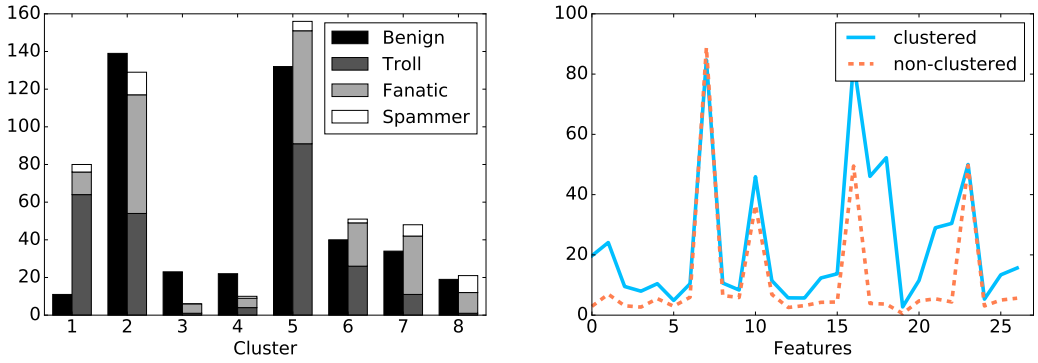
The above equation decomposes the data matrix $\mathbf{X}$ into a sum of rank-one matrices (each one being approximately a co-cluster) and the regularization penalties on the ℓ_1 norms of matrices $\mathbf{A}, \mathbf{B}$ (whose columns hold the indicator vector for each co-cluster, as shown in Fig. 3.7) ensure that the solution is sparse, so that only the users and features that belong to a particular co-cluster have a non-zero (and positive) value in our indicator vectors. Using this type of overlapping co-clustering, a user may be assigned to multiple latent behaviors at the same time, with varying degrees of participation. This enhances interpretability because it can give further insight on whether e.g., a user has more than one roles or their account has been hijacked and some of their comments are benign but some are malicious. Finally, the above co-clustering formulation is *lossy*, which means that it does not require all users to belong to at least one co-cluster. In other words, the above formulation focuses on identifying the most dominant latent behaviors and the users that engage in them, and leaves out users who do not clearly belong in one (or more) of those latent behaviors. We, henceforth, refer to those users as "non-clustered".

3.4.2 Methodology

We apply co-clustering to the same dataset we used for role classification. This dataset contains 200 benign users and 200 misbehaving users, which include 104 trolls, 21 spammers and 75 fanatics. After fine tuning the parameters of co-clustering, we find that the best number of co-clusters in the data is 8. An indication for this number is that the algorithm returned empty clusters for larger numbers of clusters. We show the distribution of users in each clusters in Figure 3.8(a).

The size of clusters varies from 29 to 268, indicating that co-clustering could not only capture the common behaviors but also the obscure ones (small clusters). In each cluster, the clustered users will exhibit a different level of value on multiple features comparing to non-clustered users as we present in Figure 3.8(b). The X-axis demonstrates the 27 features used to capture the cluster and the Y-axis is the mean of features value. Clustered users have a much higher value on feature 18 and 21 which are the percentage of capital letters and number of sentences. This explains users in this cluster exhibits a "latent behavior" of using more capital letters and more sentences in their comments.

Using latent behaviors to identify misbehaving users: 0.77 AUC. In order to show that the latent behaviors can be used to identify misbehaving users, we take **A** matrix from the co-clustering result. Each row in the matrix is a user and each column corresponds to a cluster, with the (i, j) value being the "membership" of user i to the cluster j. We use these values as features to train a classifier. Note that for each set of features we use the classifier that yielded the highest performance. In the case of raw features this was Random Forest, but for the case of the latent behavior features it was Support Vector



(a) (b)

Figure 3.8: **Co-clustering.** (a) Distribution of users in each cluster. (b) Example of latent behavior from cluster 1.

Machine (SVM) [1]. We thus report classification results using SVM. Our result shows that we can still accurately identify misbehaving users with an AUC of 0.77 with 86.3% precision and 63.5% recall.

Using latent behaviors to identify misbehaving roles: 73.3% overall accuracy. Using 10-fold cross validation, we assess the ability of our method to identify misbehaving roles, and we show the results in Table 3.4.

The overall classification accuracy when using the latent behaviors is lower than using the raw features, albeit still very good. However, we argue that the reduction in accuracy comes with an increase in the **interpretability** of the classification using latent behaviors, as we present below.

Interpreting latent behaviors. Here, we focus on the latent behaviors that help us understand misbehaving users.

[1]This behavior is expected since the latent behavior features, contrary to the raw features, are embeddings of the users in a vector space, thus SVM should work better than Random Forest.

1. Latent behavior 1: Users who exhibit such behavior are, on average, making more comments to articles, are highly active between 7-13 UTC (midnight in US), use more capital letters and also more characters and sentences on average in their comments.

2. Latent behavior 3: Users are highly active between 13-26 UTC, having ˜60% more comments and degrees of collaboration than the average.

3. Latent behavior 4: Users are active between 3-8 UTC, making slightly more (24%) comments than the average.

4. Latent behavior 6: Users are active between 6-14 UTC, making comments which have better readability.

Figure 3.8(a) shows that 87.9% of users who exhibit latent behavior 1 are misbehaving users. This is in contrast to users of latent behavior 6, which are active at similar periods of time but only 56% of them are misbehaving users. This explains that using more capital letters and making longer comments are the characteristic of misbehaving users.

Another interesting observation here is that, the active period could also imply whether a user is misbehaving or not. We see that there is an active time shift from latent behavior 3, latent behavior 4 to latent behavior 6, and the percentage of misbehaving users are 20.7%, 31.3% and 56% respectively. If we assume that most of the users are active in the timezone between ET and PT, then we may further deduce that the later at night a user is active, the more likely they are to be misbehaving.

Table 3.4: Role classification result (co-clustering)

Role	**Precision**	**Recall**
Trolls	78.7%	71.2%
Spammers	60.7%	81%
Fanatics	76.4%	73.3%
Benign	71.4%	73.5%
Overall accuracy	73.3%	

3.5 Related Work

Detecting malicious behavior. Many studies detect anti-social behavior [20] [19], bots [22] [54] or cyber-bullying [38] but they focus on OSNs. A recent work [20] analyzes the comments on three websites using Disqus, but the focus is to predict if a user will be banned due to misbehavior, which is somewhat different from ours: (a) some misbehaviors are not necessarily reported, as we saw, and (b) we have a fine-grained definition of misbehavior.

Detecting malicious behavior. Many studies detect anti-social behavior [20] [19], bots [22] [54] or cyber-bullying [38] but they focus on OSNs. A recent work [20] analyzes the comments on three websites using Disqus, but the focus is to predict if a user will be banned due to misbehavior, which is somewhat different from ours: (a) some misbehaviors are not necessarily reported, as we saw, and (b) we have a fine-grained definition of misbehavior.

Analyzing text of online users. Many studies leverage textual analysis to detect spammers on OSNs [60] and blogs [52] or how to achieve the same objective by analyzing the behavioral patterns [64].

Modeling and understanding online user behavior. Several works study temporal and behavioral patterns of OSN users [32] [28] [37]. Some recent studies also use mobile phones and OSNs to profile users' psychological states [68] [26].

Chapter 4

TrackAdvisor: Taking back browsing privacy from Third-Party Trackers

Would you feel that your privacy is violated if someone knew which websites you visited last night? Most people would feel uneasy and want to ensure their personal browsing information is not revealed to anyone else but the opposite is exactly what has been happening thanks to a phenomenon called **third-party tracking**. As a user visits a website of interest, third-party websites linked to that website become aware of the user's browsing activities and due to the ubiquitous use of cookies, these third-parties can uniquely identify the user[1]. Although this can be appalling for privacy-sensitive users, there is no violation

[1]In general, it is more accurate to say that third party tracking can track and identify web-browsers and not end users. In the rest of this document, we will use the term "tracking a user" to imply tracking the browser that is being used.

of laws. The third-party tracker is legitimately contacted by the user's browser, because it hosts resources required by the website that the user wants to visit.

It is natural to ask why the third-party tracking phenomenon is occurring and how. The answer to the "why" question is money, marketing, and advertising. It is easy to see that knowing how many users watch golf scores and search for luxury cars can help one place ads more effectively. With third-party tracking, ads on a website can be customized based on the user's visits to other websites. If you searched for yachts on one site, you could be shown yacht insurance ads on another site. The answer to the "how" question is the widespread use of: (a) embedded links on a webpage (think Facebook "Like" or Google+ "+1" button) or content being pulled from another site, and (b) cookies. Cookies turn any browser into a silent accomplice as the browser voluntarily provides cookies to the third-party websites. These cookies could have been obtained from a tracking website at an earlier time (e.g. when we logged in to Facebook). The obvious solution would not work: not sending cookies at all will often degrade the user experience or even "break" the interaction with websites.

In this chapter, we want to answer two main questions: (a) *How can we identify cookie-based third-party tracking accurately?* and (b) *How widespread is the phenomenon of third-party tracking?* To address both questions, we need a method that, when given a website and the HTTP interactions between users and that website, can identify third-party trackers. The challenge lies in identifying features of cookies and of the user interaction in general that can accurately identify third-party trackers. This is non-trivial and there exists no such method in the literature, as we discuss below. For the remainder of this chapter, we

use the term **privacy** to refer to the right of a web-browsing user to not have a third-party website become aware of websites that the user visits. We focus on cookie-based tracking, because it is still the most prevalent form of tracking, as we discuss in § 4.6.

There has been very little attention on measuring the pervasiveness of third-party tracking activities, which is our focus here. To the best of our knowledge, the most widely-used approaches to combat the third-party tracking problem rely on black lists of third-party trackers, which are maintained by corporations or communities. Microsoft's **Tracking Protection Lists (TPL)** [10] is one such prominent black list, which aggregates many others. As we show later, these efforts are far from perfect, as they are geared towards blocking the more well-known third-party trackers. We discuss related and complementary research efforts in § 4.6.

The contribution of this work is a systematic study of the third-party tracking phenomenon and its extent. We also briefly discuss practical countermeasures to enable users to protect their web-browsing privacy. First, we propose TrackAdvisor, an effective method to detect third-party trackers that surpasses existing third-party tracking lists in terms of both accuracy and detection. Second, we use TrackAdvisor to study the prevalence of third-party tracking among Alexa's Global Top 10K websites. We outline our key contributions and results below.

a. We develop TrackAdvisor, a supervised learning approach that identifies third-party trackers with high accuracy. A key novelty of our approach is that it does not rely on a blacklist of websites; TrackAdvisor focuses on the collective statistics of all cookies inside an outgoing third-party HTTP request to infer whether the third-party

website that receives those cookies is tracking the user. Using Machine Learning techniques and carefully selected features, our method exhibits 99.4% precision and 100% recall.

b. We evaluate the accuracy and completeness of TPL and show it yields a relatively low Recall of 72.2%. Microsoft's Tracking Protection Lists (TPL), which combines many existing blacklists, achieves a Recall of 72.2% although with a high Precision of 96.3%. TPL is incorporated in Internet Explorer and can therefore be thought of as the protection that is readily available to Windows users. As a result, its low Recall is somewhat disconcerting.

c. We show that third-party tracking is prevalent: 46% of Alexa's Global top 10K sites being tracked. We find that close to 46% of the *home pages* of the websites in Alexa's Top 10,000 websites have at least one third-party tracker and on average, one out of every three HTTP requests sent to third-party websites is sent to a third-party tracker. More worrisomely, Google is monitoring 25% of the Alexa sites as a third party tracker through its ad and analytics services. As expected, Facebook and Twitter are also prominent third-party tracking, as Facebook "Like" and Twitter's "Tweet" widgets have become very common, especially on blogs and news-related websites. Interestingly, two lesser known companies, Scorecard Research and QuantServe, are among the top five third-party trackers in our dataset.

4.1 Background

A. Cookies. In the context of the HTTP protocol and web browsing, a cookie is a small, **local** file (about 4KB in size) that helps a website identify a user and their

preferences and it is intended to quickly provide the remote website with information such as language (for rendering the content in the correct language) or geographic location (maybe for nearest store location). Cookies are created by the website and stored on the device by the browser the first time the user visits the website. During every subsequent visit, the browser volunteers the saved information to the website.

There are two main components to the structure of a cookie.

1. A **Name** and **Value** pair, which is explicitly set by the website. The pair can be used to save a user's language preference or geographic location. In the case of a third-party tracker, the value portion will be assigned a string that represents a user's unique ID.

2. **Attributes**, which tell the browser how to handle the cookies. The most common attributes of the cookies are: (a) the **domain** that instructs the browser which cookies to send to which websites upon visit and (b) the **expiration**, which is a timestamp specifying to the browser when to a cookie is to be discarded.

B. Third-party tracking. There are three parties involved in a user's visiting a website: the target website w (the first party) the user wants to visit, the user u (the second party), and the entities (the third party) hosting content external to the website w. Third parties, in this case, are generally transparent to the users and not all of them are third-party trackers.

As the browser needs to download third-party content, it must send an HTTP request to each of the third parties. We call the ones that collect information about the user at this stage **third-party trackers**.

Tracking mechanism: Although HTTP cookies are not the only means with which third-party trackers keep track of users, they are the most popular. There are three reasons to this. Firstly, all browsers can accept and send cookies. Secondly, other non-HTTP cookies exist and can be used for tracking, but they are inefficient or will create legal issues for the entities who utilize them. Finally, even though third-party websites can track a user by their browser fingerprint [31], this method incurs a much higher overhead, thus is unlikely to adopted widely. We will discuss browser fingerprinting and other tracking mechanisms in more details in § 4.6.

4.2 Methodology

In this section we will: 1) discuss characteristics of HTTP requests going to trackers and 2) provide an overview of our solution for the problem of detecting third-party trackers.

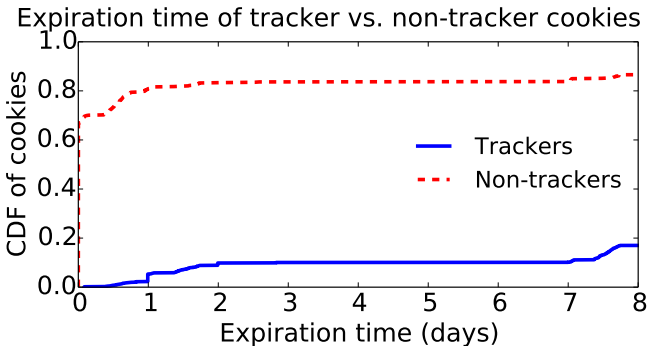
A. HTTP Requests going to third-party trackers. The key question to ask is whether there are characteristics that differentiate between: (1) HTTP requests carrying information to third-party trackers that can uniquely identify the user, and (2) HTTP requests that carry no such information.

We answer this question positively. The requests going to trackers contain **tracker cookies**, which we define as a cookie that contains a name-value pair that can uniquely identify a user. One such cookie, for instance, may have the name-value pair: `UID=163fkcs65bz` where the value is simply a unique identifier given to the browser by the website. In contrast, there are **non-tracker cookies**, which are used capture user preferences (e.g. display language, timezone), and the browser provides to them to the website in each visit. Because

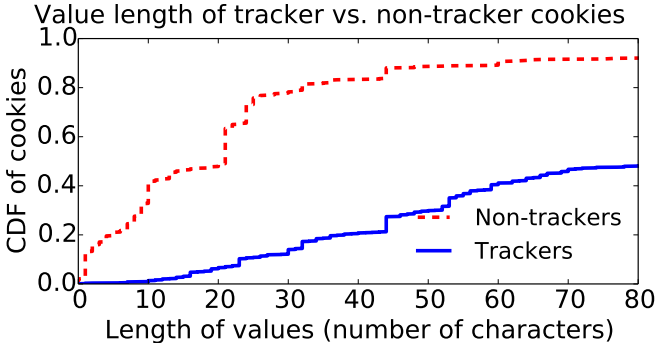
tracker cookies are meant to identify a user, they bear the following characteristics:

1. Their Lifetimes tends to be much longer than non-tracker cookies. A cookie's Lifetime is the time between its creation time and its expiration time.

2. The value part of the name-value pair inside each cookie (recall that each cookie contains only one such pair) must have sufficient **length** to be able to distinguish one user from many others.



(a) Expiration time



(b) Length of value

Figure 4.1: Difference between tracker and non-tracker cookies

In Figure 4.1(a), we show the difference in the lifetime values between tracker

cookies and non-tracker cookies that we collected and manually labeled (see § 4.3 for more details on data collection). We can see that while less than 10% of tracker cookies have a lifetime of a single day or less, at least 80% of non-tracker cookies have such short lifetime. Furthermore, Figure 4.1(b) shows that the length of the value is at least 35 characters for 80% of the tracker cookies, while 80% of the non-tracker cookies have values that are shorter than 35 characters.

The next important question to answer is, then, how we can exploit these characteristics in an effort to correctly classify HTTP requests as either going to third-party trackers and carrying user-identifiable information or purely harmless and carrying no sensitive information.

B. TrackAdvisor: Identifying trackers, one HTTP request at a time. We present TrackAdvisor, our solution for the problem of identifying third-party trackers. TrackAdvisor looks at *all of the cookies* carried by each outgoing HTTP request, extract collective statistics, and performs classification to determine whether the request is heading for a tracker.

TrackAdvisor is a supervised Machine Learning-based application that we envision to reside inside the browser, where it can inspect each outgoing HTTP request and inform the user if the HTTP request carries information that may be able to uniquely identify the user. TrackAdvisor takes as input the cookies exchanged between the browser and the remote websites and identifies the websites that are third-party trackers.

Feature selection: First, we define CookieJar(A, B) as the group of all third-party cookies exchanged between the host A and the remote website B. Note that we

exclude the Session cookies because Session cookies are created during a browsing session and are destroyed once the browser is closed. Because of their short-lived nature, Session cookies are unlikely to be used as a tracking mechanism.

Instead of looking at each cookie in CookieJar(A, B) individually, TrackAdvisor looks at CookieJar(A, B) in its entirety, extracts relevant statistics from cookie's property, and performs classification.

We started with considering a large number of features, including maximum Lifetime, minimum ValueLength, mean ValueLength, maximum ValueLength, as well as others. This set of features is then reduced to only three by the Recursive Feature Elimination (RFE) functionality of WEKA [71] which, at a high level, recommends a subset of features that achieves the best accuracy. In our case, the final three features are:

(a) Minimum lifetime: $L^{\min}_{A,B} = \min_c [\text{Lifetime}(c)]$. This feature is selected because trackers, as discussed earlier, tracker cookies tend to live longer than non-tracker cookies.

(b) Number of third-party cookies in CookieJar(A, B): $N_{A,B}$. This feature is selected because of the trackers' tendency to utilize more cookies than benign third-parties in order to record as much information about the user as possible.

(c) Augmented Lifetime: $L^{\text{aug}}_{A,B} = \sum_c [\text{ValueLength}(c) \times \text{Lifetime}(c)]$. The Augmented Lifetime captures at once captures two important characteristics of tracker cookies: long Lifetime and long ValueLength, and it is also crucial to future-proofing TrackAdvisor's performance against two possible evasive tactics from third-party trackers: **cookie chunking** and **lifetime reduction**. We will discuss the two techniques, as well as how robust TrackAdvisor is against them at the end of § 4.3.

We present the process flow that TrackAdvisor executes in the following:

1. Retain only third-party HTTP requests from the browser. A third-party HTTP request is one that is sent toward an URL that does **not** share the same hostname as the website the user intentionally visits. TrackAdvisor achieves this by looking at the referrer of the request and ignoring requests where the hostnames in the referrer and URL fields are the same.

2. For each CookieJar(A, B) representing an HTTP request sent by host A to website B, TrackAdvisor calculates three features of CookieJar(A, B), that we described above: (a) $L^{\min}_{A,B}$, (b) $N_{A,B}$, and (c) $L^{\text{aug}}_{A,B}$.

3. Use a binary classifier to classify the tuple $\langle L^{\text{aug}}_{A,B}, L^{\min}_{A,B}, N_{A,B} \rangle$.

   A positive output from the classifier means that the tuple belongs in an interaction with a third-party tracker and a negative otherwise. We will discuss how to create the classifier from training data in § 4.3.

4. If the module returns a positive value, we label B as a third-party tracker and add it to a list that will be presented to the user later.

4.3 Experiments and Evaluation

In this section we will (a) describe our data collection and preliminary labeling processes and (b) compare the performance of Microsoft's Tracking Protection Lists against that of TrackAdvisor.

A. Data Collection. Our dataset is created by visiting the *landing pages* Alexa's

Top 10K Global list [2] during the month of July of 2012. We collected our data using **FourthParty** [5], a Firefox extension that collects data in the background as the user browses the Web. The data that we collected are: a) the header of each HTTP request, b) the header of each HTTP response, and c) the cookie log associated with each request and response. We used the automation framework Selenium [61] with FourthParty installed to collect 563,031 HTTP requests and 99,397 cookies. Of all 563,031 requests, 202,556 were sent to third-party websites and 78,213 contain cookies. Out of 99,397 cookies, 22,270 cookies were sent to third-party websites.

B. Creating training and testing data sets. From the set of all HTTP requests to third-party websites, we created a training and a testing data-set as follows:

- D_{train}: includes 500 randomly chosen requests such that roughly half of them were dispatched to third-party trackers and half were meant to retrieve third-party content and containing no tracking information.

- D_{test}: includes 500 HTTP requests that were randomly chosen in a similar fashion to the ones in D_{train}.

D_{train} and D_{test} are mutually exclusive. The former is used to train TrackAdvisor and the latter will be used for testing both TrackAdvisor and Tracking Protection Lists.

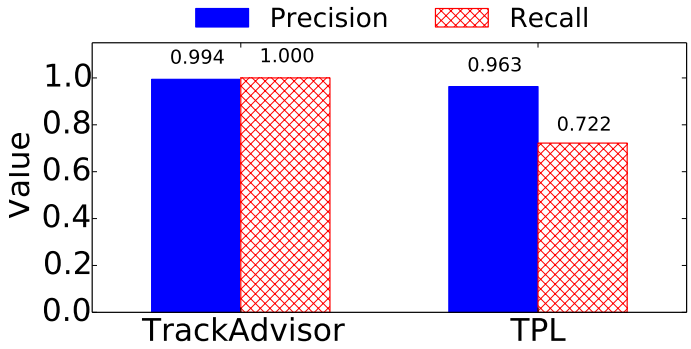
To establish the ground truth, we label the websites in D_{train} and D_{test} (1,000 in total) as either third-party trackers or benign third-party websites using extensive and careful manual evaluation. In our evaluation, we label a website as a third-party tracker by combining the information gained from the three following processes: (a) a manual inspection the website, (b) a consultation with multiple black lists specifically created for

third-party tracker, and (c) a careful inspection of cookie properties. To label something as athird-party tracker, we require significant supporting evidence to that effect. We argue that this method is essentially the same used by the contributors to third-party tracking lists. For transparency, we will make our two labelled sets available to the research community.

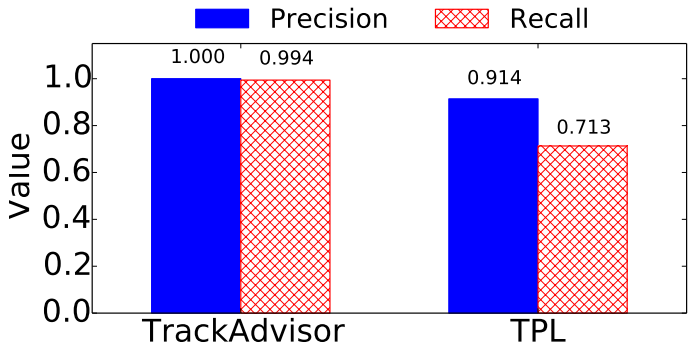
C. Reference: Microsoft's Tracking Protection List. We compare our approach against Tracking Protection Lists, which is a black list-based component that is used in Microsoft's Internet Explorer. We selected Tracking Protection Lists because: a) it uses the same popular black lists (FanBoy, EasyList, EasyPrivacy, etc.) that empower AdBlock Plus and b) it has been shown that the a combination of the popular black lists achieved comparable performance to Ghostery's [49].

D. Creating a classifier for TrackAdvisor from D_{train}. Recall from the beginning of this section that we have constructed a training dataset and a testing dataset called D_{train} and D_{test}. Also recall that each request in D_{train} is represented by a tuple $\langle L^{\text{aug}}_{A,B},\ L^{\text{min}}_{A,B},\ N_{A,B}\rangle$. Since each tuple is labeled, we are able to use the WEKA Machine Learning suite [71] to build classifiers. The algorithm that we picked from the suite is Support Vector Machine because it offers the best performance in terms of **Pr**ecision and **Re**call, where Pr = TP / (TP + FP) and Re = TP / (TP + FN). TP is the number of True Positives, FP the number of False Positives, and FN the number of False Negatives.

Before we start the testing, we examine the sensitivity of our approach to the training input by performing a ten-fold cross-validation on D_{train}. The assessment yields a combined Precision of 0.998 and Recall of 0.998 (one FN and one FP). We conclude that our approach is robust to the training data.



(a)



(b)

Figure 4.2: Classification results for HTTP requests (a) and domains (b)

E. Evaluation of classification on D_{test}. First, we check the URLs of D_{test} against Tracking Protection Lists. As shown in Figure 4.2(a), TPL achieves a Precision and Recall of 96.3% and 72.2% respectively (13 FPs and 134 FNs). In contrast, TrackAdvisor achieves perfect Recall and nearly perfect Precision (0 FPs and 2 FNs).

One possible reason why TPL has so many False Negatives could be that TPL is better tuned to recognize the trackers more relatively well-known to the community, as it relies significantly on user reports to populate the list.

F. Possible evasive tactics from third-party trackers: An inquisitive reader

may ask why we simply did not use only ValueLength and Lifetime as features for the classifier even though as we have shown in Figure 4.1 that the ValueLengths and Lifetimes of non-tracker cookies are different from those of tracker cookies. The reason is that a classifier built from only ValueLength and Lifetime is ineffective against two possible evasive tactics from third-party trackers:

T1. Cookie Chunking: Instead of using a single cookie that contains an identifier, third-party trackers can chop it into multiple cookies with different *names* that will be combined later when the HTTP requests are processed at the server. This way, they can reduce the lengths of the cookies and help them avoid detection.

T2. Lifetime Reduction: Instead of setting a large value for the expiration of the cookies, trackers can use smaller values depending on their own *popularity*. For example, a very popular website like Google can set their cookie lifetime to a month or even a week instead of a year because Google knows people visit the site frequently.

We have conducted extensive experiments on the robustness of TrackAdvisor against **T1** and **T2** where we (a) identify every tracker cookie in each HTTP request (in both D_{train} and D_{test}) that we manually label as going to third-party trackers, (b) either split them up according to **T1** or reduce their lifetimes according to **T2**, and (c) re-train our classifier on D_{train} and re-test on D_{test}.

4.4 The pervasiveness of third-party trackers

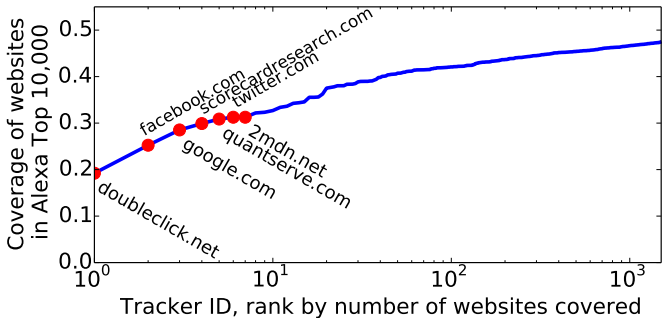
In this section, we quantify the extent of third-party tracking by analyzing the Alexa Top 10K websites. Overall, we find a significant presence of third-party tracking that

would be disconcerting to privacy advocates.

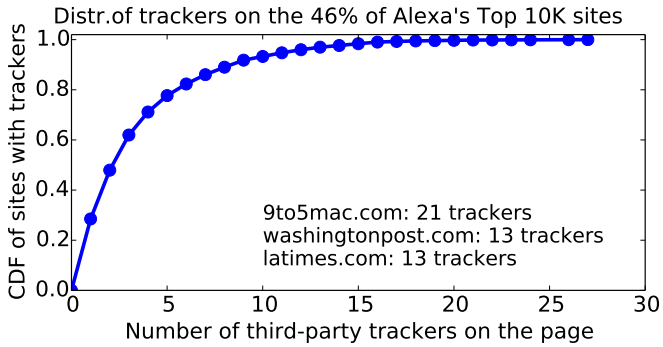
A) 46% of the Alexa Top 10K websites have at least one third-party tracker on them. By applying TrackAdvisor on our entire dataset, we found that 46% of the Alexa Top 10K websites had at least one third-party tracker on them. We use the term "target website" to refer to the Alexa website that was explicitly visited by the user in each request as we explained earlier. We plot the cumulative coverage in terms of unique target sites as a function of the number third-party trackers in the order of decreasing activity in Figure 4.3(a). In more detail, for each third-party tracker t, let S_t be the set of websites in our dataset that are tracked by t. On the x-axis, we order the trackers in decreasing order in terms of the number of sites on which they appear: $|S_{t_i}|$. The y-axis is the cumulative coverage (C_{t_i}) of the first i trackers in that order. $C_{t_i} = |\cup_{k=1}^{i} S_{t_k}|/N$ where $N = 10,000$ is the total number of target websites. We can see from Figure 4.3(a) that:

- 46% of the Alexa Top 10,000 websites have at least one tracker on them.
- The top 5 most common trackers cover 30% of the top 10,000 sites.
- Google alone (`doubleclick.com` and `google.com`) covers 25% of the sites. The `doubleclick.com` domain is responsible for advertisements and `google.com` is where other websites download widgets and libraries.

B. The majority of tracked sites are tracked by more than one tracker. Equally interesting is the fact that a website that has third-party tracking is likely to contain multiple trackers. In Figure 4.3(b), we plot the CDF of the distribution of third-party trackers on the Alexa websites that have at least one tracker. For example, we see



(a)



(b)

Figure 4.3: Study the third-party trackers in real world: (a) Cumulative coverage of top 10K Alexa sites as a function of third-party trackers in the order of decreasing tracking presence in our dataset, and (b) The distribution of the number of trackers on the Alexa top 10K sites.

in the plot that 28% of websites have one tracker, which means that there are at least two trackers present on each of the remaining websites (72%).

We also find that 29% of the websites that are tracked by at least five third-party trackers. For a visitor that means that five different entities become aware of her web-surfing preferences. It is equally worrisome to see that some popular websites such as `latimes.com` and `washing-tonpost.com` have upwards of 10 third-party trackers.

The well-known Google Analytics is not on the list in Figure 4.3(b), because by

contract, Google Analytics provides statistics only to the first-party websites and the cookies set by Google Analytics are always associated with the domains of the first-party websites and therefore are not third-party cookies. Furthermore, the same user who visits different websites monitored by Google Analytics will likely receive different IDs, which makes tracking him or her non-trivial.

C. Third-party interactions: 37% tracking versus 63% benign. Recall from § 4.3 that our dataset contains a total of 202,556 third-party HTTP requests, which includes both third-party tracking and benign third-party interactions. Using our approach, we identify 75,849 (37%) of them as third-party tracking interactions. This is of interest in considering counter-measures to third-party tracking, since there is a large number of interactions with benign third-party websites, as we discuss in the next section.

4.5 Possible solutions against third-party trackers

Here, we discuss some potential solutions that can be implemented in a browser fairly easily to block third-party trackers from collecting user information.

A. Blocking all third-party cookies. One can consider labeling as trackers all third-party websites that exchange cookies with the user's computer. On the one hand, this type would allow a user to block 100% of the trackers with a false positive rate of 12.6%. On the other hand, that comes at the expense of the degraded browsing experience. There are websites that refuse to display their content unless the user's browser accepts third-party cookies. More specifically, with third-party cookies disabled, iFrames, widely used in third-party games and apps on social networks, cannot read their own cookies [13]

and cannot work. As we saw in § 4.4, the majority (63%) of requests to third-party websites is benign. A complete blocking solution would have unnecessarily blocked them.

B. Removing/Anonymizing the referrer fields in HTTP requests. Apart from the cookies that can uniquely identify users, the values of the referrer fields of the HTTP requests are important to the third-party websites' ability to partially construct a user's browsing history. Therefore, using TrackAdvisor to identify HTTP requests carrying identifying information and then either removing the referrer information or replacing it with bogus values is one way to protect the user's privacy. To the best of our knowledge, third-party websites have tried to withhold content from the users only in the case where the browsers would not accept the cookies and no efforts at all have been invested in validating the referrers as a condition to provide content.

Here we only provide suggestions for possible defense methods against third-party trackers. The full evaluation of the two methods is, however, beyond the scope of this work and may be tackled in a future work.

4.6 Related Work

Although much attention has been devoted to studying the phenomenon of third-party trackers using cookies to track users [44, 50, 70], there exists no practical solution that leverages cookies as a means to detect third-party tracking. To the best of our knowledge, all existing practical solutions such as AdBlock Plus [1], Microsoft's Tracking Protection Lists [10], Collusion [3], and Ghostery [7] rely on corporate- and community-maintained black lists (sometimes called block lists) to block HTTP requests to well-known third-party

trackers. AdBlock Plus is an improvement to the original AdBlock that also blocks third-party trackers in addition to advertisements. Ghostery and TPL are completely focus on blocking trackers instead.

All other related work have been focused on uncovering other types of cookies (aside from the standard HTTP ones) that could be used to track users but did not propose countermeasures like we did. In [16, 51], the authors documented the use of Flash cookies, which are Locally Shared Objects similar to cookies. Advertisers can create a pair of cookies, an HTTP one and a Flash one, with identical content, where the latter can "re-spawn" the former even after the former has been deleted. Fortunately, the practice of using Flash cookies have been on the decline because there have been lawsuits against the advertisers, who essentially re-spawned the HTTP cookies against the users' will.

There is a form of cookie-less tracking, which is cache-based and utilizes ETags [9, 16]. An ETag, assigned by the website and unique for each user, is associated with an object on a web page (like an image) that can tell the server if the object in the browser cache is the same as the one on the server. An advertiser then can have exactly the same objects on many websites and track the users just like they would with cookies. This method is not popular, as users can just clear the browser caches frequently.

Most modern browsers offer a "Do Not Track" option which is nothing more than a request and the websites can ignore it if they choose to. The most recent high-profile website that decided to not honor "Do Not Track" is Yahoo [14]. The Electronic Frontier Foundation then responded by releasing Privacy Badger [12], a browser add-on that detects third-party trackers. It keeps tracks of all cookies as the user visits websites and blocks

cookies that are *previously seen*. This is a promising development, but, given that this was released only in May 2014, there are no reports yet as to how well Privacy Badger works, if it degrades user experience, and how much overhead it may add in terms of memory due to the large number of cookies that need to be tracked.

Finally, there exists a form of tracking using the *fingerprint* [31] of the browsers. This form of tracking relies on the information that the browser sends to the remote website (such as IP address, User-Agent, System fonts, screen resolution etc.). The remote website then can use all of this information to uniquely identify the browser that the request comes from. However, because the overhead that incurs is very high for browser fingerprinting, we would make the argument that third-party trackers are unlikely to adopt it as a means to track the browsing behaviors of users.

Chapter 5

Conclusions

In this dissertation, we show how we can use Machine Learning, Graph Analytics and Natural Language Processing to tackle three challenging problems: (a) modeling users' on-line behavior on commenting platforms, (b) detecting and classifying users misbehavior and (c) detecting third-party trackers and studying the extent of such phenomenon.

First, we analyze users on-line behavior along three dimensions: (a) The social behavior, (b) the user-article engagement, (c) the temporal properties of users. We ultimately propose a way to profile users relying on features along these dimensions. We have collected information from 1 million users between Nov-2007 to Jan- 2015, and study 109K users, who have more than 10 comments in the platform. Using our profiles, we show how we can detect anomalous users and how the profiles gives us a glimpse of their unusual behavior. In other words, our profiles provide a first step of forensics analysis: we identify profiles for colluding users, spammers and trolls.

Second, we extend our work above by adding one extra dimension, the linguistic

property of users on-line behavior, and then we develop a systematic and comprehensive methodology to identify malicious users on commenting platforms with fine-grained classification of malicious behavior in four major news media websites. From an algorithmic point of view, our work uses a comprehensive set of 73 features across four different types of information, and a novel two-stage classification. The overall classification accuracy of our approach is 80.8% for the fine-grained 4-class problem. This work is a first step towards safeguarding commenting platforms from malicious users.

Finally, We present TrackAdvisor, a Machine Learning-based method designed to detect third-party trackers and argue that this could be the basis for protecting the users' privacy from third-party trackers. TrackAdvisor's novelty is its focus on the interactions between the browsers and the remote websites to detect when the user's browsing privacy is being leaked instead of relying on black lists. TrackAdvisor exhibits high Precision (99.4%) and Recall (100%) in contrast with a Recall of 72.2% by Microsoft's Tracking Protection Lists, which is a black list-based component in the widely used Internet Explorer. Our study shows that 46% of the websites on Alexa's Global Top 10,000 list contain at least one tracker each and 25% of the 10,000 are tracked by a single entity: Google.

In conclusion, this dissertation provides significant steps towards deploying Machine Learning and Data Mining to practical problems in understanding and protecting users behavior online.

Bibliography

[1] Adblock plus. https://adblockplus.org.

[2] Alexa, the web information company. http://www.alexa.com.

[3] Collusion, browser extension. https://chrome.google.com/webstore/detail/collusion-for-chrome/ganlifbpkcplnldliibcbegplfmcfigp.

[4] Firefox on disabling third-party cookies. https://support.mozilla.org/en-US/kb/disable-third-party-cookies.

[5] Fourthparty firefox extension. http://fourthparty.info.

[6] The free web debugging proxy for any browser, system or platform.

[7] Ghostery. https://www.ghostery.com/.

[8] Google saved an estimated 887 million by paying adblock plus to show its ads. http://www.businessinsider.com/google-saved-an-estimated-887-million-by-paying-adblock-plus-to-show-its-ads-2013-8.

[9] Http etags. http://en.wikipedia.org/wiki/HTTP_ETag.

[10] Microsoft's tracking protection lists. http://ie.microsoft.com/testdrive/Browser/p3p/Default.html.

[11] Preview problem and third-party cookies. http://en.support.wordpress.com/third-party-cookies/.

[12] Privacy badger. http://www.theregister.co.uk/2014/05/02/eff_privacy_badger/.

[13] Third-party iframes can no longer read their own cookies when "block third-party cookies and site data" is enabled. https://code.google.com/p/chromium/issues/detail?id=113401.

[14] Yahoo declines to honor "do not track". http://yahoopolicy.tumblr.com/post/84363620568/yahoos-default-a-personalized-experience.

[15] Jeff Alstott, Ed Bullmore, and Dietmar Plenz. powerlaw: a python package for analysis of heavy-tailed distributions. *PloS one*, 9(1):e85777, 2014.

[16] Mika Ayenson, Dietrich Wambach, Ashkan Soltani, Nathan Good, and Chris Hoofnagle. Flash cookies and privacy ii: Now with html5 and etag respawning. *Social Science Research Networks*, 2011.

[17] Lars Backstrom, Eric Sun, and Cameron Marlow. Find me if you can: improving geographical prediction with social and spatial proximity. In *WWW*, pages 61–70. ACM, 2010.

[18] Arzu Baloglu and Mehmet S Aktas. Blogminer: Web blog mining application for classification of movie reviews. In *ICIW*, pages 77–84. IEEE, 2010.

[19] Justin Cheng, Michael Bernstein, Cristian Danescu-Niculescu-Mizil, and Jure Leskovec. Anyone can become a troll: Causes of trolling behavior in online discussions. In *CSCW*, 2017.

[20] Justin Cheng, Cristian Danescu-Niculescu-Mizil, and Jure Leskovec. Antisocial behavior in online discussion communities. In *ICWSM*, pages 61–70, 2015.

[21] 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.

[22] Zi Chu, Steven Gianvecchio, Haining Wang, and Sushil Jajodia. Detecting automation of twitter accounts: Are you a human, bot, or cyborg? *IEEE Transactions on Dependable and Secure Computing*, 9(6):811–824, 2012.

[23] Aaron Clauset, Cosma Rohilla Shalizi, and Mark EJ Newman. Power-law distributions in empirical data. *SIAM review*, 51(4):661–703, 2009.

[24] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[25] Clodoveu A Davis Jr, Gisele L Pappa, Diogo Rennó Rocha de Oliveira, and Filipe de L Arcanjo. Inferring the location of twitter messages based on user relationships. *Transactions in GIS*, 15(6):735–751, 2011.

[26] Munmun De Choudhury, Scott Counts, Eric J Horvitz, and Aaron Hoff. Characterizing and predicting postpartum depression from shared facebook data. In *CSCW*, pages 626–638. ACM, 2014.

[27] Intense Debate. Intense debate: Imagine better comments. `http://intensedebate.com`.

[28] Pravallika Devineni, Danai Koutra, Michalis Faloutsos, and Christos Faloutsos. If walls could talk: Patterns and anomalies in facebook wallposts. In *ASONAM*, pages 367–374. ACM, 2015.

[29] Disqus. What's cooler than a billion monthly uniques? `http://blog.disqus.com/post/50374065365/whats-cooler-than-a-billion-monthly-uniques`, 05 2013.

[30] Disqus. Disqus: Blog-comment hosting service. `https://disqus.com`, 2015.

[31] Peter Eckersley. How unique is your web browser? In *Privacy Enhancing Technologies*, pages 1–18. Springer, 2010.

[32] Alceu Ferraz Costa, Yuto Yamaguchi, Agma Juci Machado Traina, Caetano Traina Jr, and Christos Faloutsos. Rsc: Mining and modeling temporal activity in social media. In *SIGKDD*, pages 269–278. ACM, 2015.

[33] Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23–37. Springer, 1995.

[34] Judith Gelernter and Shilpa Balaji. An algorithm for local geoparsing of microtext. *GeoInformatica*, 17(4):635–667, 2013.

[35] Judith Gelernter and Wei Zhang. Cross-lingual geo-parsing for non-structured data. In *Proceedings of the 7th Workshop on Geographic Information Retrieval*, pages 64–71. ACM, 2013.

[36] Vicenç Gómez, Andreas Kaltenbrunner, and Vicente López. Statistical analysis of the social network and discussion threads in slashdot. In *Proceedings of the 17th international conference on World Wide Web*, pages 645–654. ACM, 2008.

[37] Xiaotao Gu, Hong Yang, Jie Tang, and Jing Zhang. Web user profiling using data redundancy. In *ASONAM*, pages 358–365. IEEE, 2016.

[38] Homa Hosseinmardi, Rahat Ibn Rafiq, Richard Han, Qin Lv, and Shivakant Mishra. Prediction of cyberbullying incidents in a media-based social network. In *ASONAM*, pages 186–192. IEEE, 2016.

[39] IMDB. Imdb: Internet movie database. `http://www.imdb.com/`.

[40] David Jurgens. That's what friends are for: Inferring location in online social media platforms based on social relationships. *ICWSM*, 13:273–282, 2013.

[41] Samy Kamkar. Evercookie–never forget. *New York Times*, 2010.

[42] Andreas Klaus, Shan Yu, and Dietmar Plenz. Statistical analyses support power law distributions found in neuronal avalanches. *PloS one*, 6(5):e19779, 2011.

[43] Michal Kosinski, David Stillwell, and Thore Graepel. Private traits and attributes are predictable from digital records of human behavior. *Proceedings of the National Academy of Sciences*, 110(15):5802–5805, 2013.

[44] Pedro Leon, Blase Ur, Richard Shay, Yang Wang, Rebecca Balebako, and Lorrie Cranor. Why johnny can't opt out: A usability evaluation of tools to limit online behavioral advertising. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 589–598. ACM, 2012.

[45] LiveFyre. Livefyre: Real-time content marketing and engagement. `http://web.livefyre.com`.

[46] Jalal Mahmud, Jeffrey Nichols, and Clemens Drews. Where is this tweet from? inferring home locations of twitter users. *ICWSM*, 12:511–514, 2012.

[47] Jalal Mahmud, Jeffrey Nichols, and Clemens Drews. Home location identification of twitter users. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 5(3):47, 2014.

[48] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, 2014.

[49] Jonathan Mayer. Tracking the trackers: Self-help tools. `http://cyberlaw.stanford.edu/node/6730`.

[50] Jonathan R Mayer and John C Mitchell. Third-party web tracking: Policy and technology. In *Security and Privacy (SP), 2012 IEEE Symposium on*, pages 413–427. IEEE, 2012.

[51] Aleecia M McDonald and Lorrie Faith Cranor. A survey of the use of adobe flash local shared objects to respawn http cookies. *A Journal of Law and Policy for the Information Society*, 7:639–721, 2012.

[52] Gilad Mishne, David Carmel, and Ronny Lempel. Blocking blog spam with language model disagreement. In *AIRWeb*, volume 5, pages 1–6, 2005.

[53] Gilad Mishne and Natalie Glance. Leave a reply: An analysis of weblog comments. In *Third annual workshop on the Weblogging ecosystem*. Edinburgh, Scotland, 2006.

[54] Fred Morstatter, Liang Wu, Tahora H Nazer, Kathleen M Carley, and Huan Liu. A new approach to bot detection: striking the balance between precision and recall. In *ASONAM*, pages 533–540. IEEE, 2016.

[55] Abdullah Mueen, Krishnamurthy Viswanathan, Chetan Gupta, and Eamonn Keogh. The fastest similarity search algorithm for time series subsequences under euclidean distance. `http://www.cs.unm.edu/~mueen/FastestSimilaritySearch.html`, August 2015.

[56] Part of Speech Tagging. Parts-of-speech tagging. `http://bioie.ldc.upenn.edu/wiki/index.php/Part-of-Speech`.

[57] Evangelos E Papalexakis, Nicholas D Sidiropoulos, and Rasmus Bro. From k-means to higher-way co-clustering: Multilinear decomposition with sparse latent factors. *IEEE transactions on signal processing*, 61(2):493–506, 2013.

[58] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, 1988.

[59] Md Sazzadur Rahman, Ting-Kai Huang, Harsha V Madhyastha, and Michalis Faloutsos. Efficient and scalable socware detection in online social networks. In *USENIX Security Symposium*, pages 663–678, 2012.

[60] D Sculley and Gabriel M Wachman. Relaxed online svms for spam filtering. In *SIGIR*, pages 415–422. ACM, 2007.

[61] Selenium. Selenium, web browser automation. `http://docs.seleniumhq.org/`.

[62] Slashdot. Slashdot. `http://slashdot.org`.

[63] Sara Owsley Sood, Elizabeth F Churchill, and Judd Antin. Automatic identification of personal insults on social news sites. *Journal of the American Society for Information Science and Technology*, 63(2):270–285, 2012.

[64] Ashish Sureka. Mining user comment activity for detecting forum spammers in youtube. *arXiv preprint arXiv:1103.5044*, 2011.

[65] Sysomos. Inside twitter: An in-depth look inside the twitter world. `http://sysomos.com/sites/default/files/Inside-Twitter-BySysomos.pdf`, April 2014.

[66] Topia TermExtract. Topia.termextract: Content term extraction using pos tagging. `https://pypi.python.org/pypi/topia.termextract/`.

[67] Bruce Upbin. What are the best times to share on facebook and twitter? `http://www.forbes.com/sites/bruceupbin/2012/05/09/when-to-make-stuff-go-viral-online/`, 05 2012.

[68] Rui Wang et al. Studentlife: assessing mental health, academic performance and behavioral trends of college students using smartphones. In *UbiComp*, pages 3–14. ACM, 2014.

[69] Christina Warren. When are facebook users most active? `http://mashable.com/2010/10/28/facebook-activity-study/#XGYDkhqSmkqQ`, 10 2010.

[70] Zachary Weinberg, Eric Yawei Chen, Pavithra Ramesh Jayaraman, and Collin Jackson. I still know what you visited last summer: Leaking browsing history via user interaction and side channel attacks. In *Security and Privacy (SP), 2011 IEEE Symposium on*, pages 147–161. IEEE, 2011.

[71] Ian H Witten, Eibe Frank, Leonard E Trigg, Mark A Hall, Geoffrey Holmes, and Sally Jo Cunningham. Weka: Practical machine learning tools and techniques with java implementations. 1999.