DETECTING AND REASONING COLLUSIVE ACTIVITIES IN ONLINE MEDIA

BY

HRIDOY SANKAR DUTTA

Under the supervision of

Tanmoy Chakraborty

COMPUTER SCIENCE AND ENGINEERING

INDRAPRASTHA INSTITUTE OF INFORMATION TECHNOLOGY DELHI

NEW DELHI– 110020

SEPTEMBER, 2021

DETECTING AND REASONING COLLUSIVE ACTIVITIES IN ONLINE MEDIA

BY

HRIDOY SANKAR DUTTA

A THESIS

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy

COMPUTER SCIENCE AND ENGINEERING

INDRAPRASTHA INSTITUTE OF INFORMATION TECHNOLOGY DELHI

NEW DELHI– 110020

SEPTEMBER, 2021

Certificate

This is to certify that the thesis titled *Detecting and reasoning collusive activities in online media* being submitted by *Hridoy Sankar Dutta* to the Indraprastha Institute of Information Technology Delhi, for the award of the degree of Doctor of Philosophy, is an original research work carried out by him under my supervision. In my opinion, the thesis has reached the standard fulfilling the requirements of the regulations relating to the degree.

The results contained in this thesis have not been submitted in part or full to any other university or institute for the award of any degree or diploma.

September, 2021

Dr. Tanmoy Chakraborty

Indraprastha Institute of Information Technology Delhi

New Delhi 110020

Abstract

Online media platforms have enabled users to connect with individuals, organizations and share their thoughts. Other than connectivity, these platforms also serve multiple purposes - education, promotion, updates, awareness, etc. Increasing the reputation of individuals in online media (*aka social growth*) is thus essential these days, particularly for business owners and event managers who are looking to improve their sales and reputation. The natural way of gaining social growth is a tedious task, which leads to the creation of unfair ways to boost the reputation of individuals artificially. We refer to such unfair ways of bolstering social reputation in online media as *collusion*.

This thesis covers various aspects of *collusion*: a large-scale analysis of collusive entities and designing state-of-the-art models for detection of collusive entities in multiple online media platforms. First, we design approaches using user's metadata properties to identify collusive users on Twitter who request for artificial retweets from the blackmarket services. Here, we also explore the differences between the working of various types of blackmarket services. Second, we extend our previous approaches to identify collusive Twitter users using user's network properties. Third, we consider another type of collusive Twitter appraisal (followers) and study the collusive entities present in another online media platform (YouTube). Fourth, we propose an approach to detect core users of the blackmarket services and show the differences in the working of core and non-core users. Finally, we release a multi-platform data repository of collusive entities collected from two blackmarket services.

To,
All my family and friends

Acknowledgements

Above all, I want to thank my advisor Dr. Tanmoy Chakraborty. He fostered my personal and professional growth with a lot of care and patience. All our interactions helped me to discard weak and seemingly interesting ideas and to concentrate my research efforts towards innovative and ambitious goals. I believe we share many principles and are a good match as an advisor and an advisee. I am grateful for having this wonderful experience. I must say I will miss working with him as his student.

I would like to say thank you to the faculties in my yearly progress committee members, namely Prof. Vikram Goyal, Dr. Sambuddho Chakravarty and Dr. Md Shad Akhtar. Each and every thoughtful feedback that I received from them stimulated me to better shape my ideas and encouraged me to improve my research work. I hope there will be opportunities in the future to discuss and address together with them new questions and challenges that this thesis raises.

Over the years, students and staffs in Laboratory for Computational Social Systems (LCS2) research group have been sharing with me their kindness and great insights on my work. The friendly and collaborative culture in our lab has helped me a lot throughout my time here. I must say you all have made IIITD a home for me. Over the course of my graduate career, I had met many knowledgeable researchers and friends at my institute and at various conferences/workshops. Many thanks to all of you. Thanks to Dr. Hemant Purohit for hosting my visit to George Mason University in Summer 2019. I very much enjoyed our collaboration and learned a great deal during my time there.

Last but not least, I am most thankful to my family. It is impossible to express the debt of gratitude that I owe them. I cannot thank them enough for their unconditional love and countless sacrifices that ensured my ability to pursue my own dreams. I love them dearly.

Publications

Journals

J1. Hridoy Sankar Dutta and Tanmoy Chakraborty. "Blackmarket-Driven Collusion Among Retweeters–Analysis, Detection, and Characterization." *IEEE Transactions on Information Forensics and Security (TIFS)* 15 (2019): 1935-1944.

J2. Udit Arora, Hridoy Sankar Dutta, Brihi Joshi, Aditya Chetan and Tanmoy Chakraborty. "Analyzing and Detecting Collusive Users Involved in Blackmarket Retweeting Activities." *ACM Transactions on Intelligent Systems and Technology (TIST)* 11, no. 3 (2020): 1-24.

J3. Hridoy Sankar Dutta, Vishal Raj Dutta, Aditya Adhikary, and Tanmoy Chakraborty. "HawkesEye: Detecting fake retweeters using Hawkes process and topic modeling." *IEEE Transactions on Information Forensics and Security (TIFS)* 15 (2020): 2667-2678.

J4. Hridoy Sankar Dutta, Mayank Jobanputra, Himani Negi, and Tanmoy Chakraborty. "Detecting and analyzing collusive entities on YouTube." In *ACM Transactions on Intelligent Systems and Technology (TIST)* (2021).

J5. Hridoy Sankar Dutta and Tanmoy Chakraborty. "Collusion on the web: A Survey." In *ACM/IMS Transactions on Data Science (TDS)*. (Minor Revision)

Conferences

C1. Hridoy Sankar Dutta, Aditya Chetan, Brihi Joshi, and Tanmoy Chakraborty. "Retweet us, we will retweet you: Spotting collusive retweeters involved in blackmarket services." In *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 242-249. IEEE, 2018.

C2. Aditya Chetan, Brihi Joshi, Hridoy Sankar Dutta, and Tanmoy Chakraborty. "Corerank: Ranking to detect users involved in blackmarket-based collusive retweeting activities." In Proceedings of *the Twelfth ACM International Conference on Web Search and Data Mining (WSDM)*, pp. 330-338. 2019.

C3. Hridoy Sankar Dutta, Udit Arora and Tanmoy Chakraborty. "ABOME: A Multi-platform Data Repository of Artificially Boosted Online Media Entities." In *Proceedings of the International AAAI Conference on Web and Social Media (ICWSM)*, 15(1), 1000-1008, 2021.

C4. Shreyash Arya and Hridoy Sankar Dutta. "Revealing the Blackmarket Retweet Game: A Hybrid Approach." In *AAAI Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situation (CONSTRAINT@AAAI)*, pp. 30-41. 2021.

C5. Hridoy Sankar Dutta, Kartik Aggarwal and Tanmoy Chakraborty. "DECIFE: Detecting Collusive Users Involved in Blackmarket Following Services on Twitter." In *Proceedings of the 32nd ACM Conference on Hypertext and Social Media*, pp. 91-100. 2021.

C6. Hridoy Sankar Dutta, Nirav Diwan and Tanmoy Chakraborty. "Weakening the Inner Strength: Spotting Core Collusive Users in YouTube Blackmarket Network." In *International AAAI Conference on Web and Social Media (ICWSM)*. (Under Revision)

Contents

List of Tables

List of Figures

1. Introduction

1.1 Thesis overview and statement

The prosperity of online media has attracted people and organizations to join the platform and use it for several purposes - create a network among commonalities, build and broaden their business, promote/demote e-commerce products, etc. This has led users choose artificial ways of gaining social growth to get benefits in a short time. The main reason behind choosing artificial boosting is that the legitimate efforts of gaining appraisals (followers, retweets, likes, shares, etc.) take a significant amount of time and may not meet the actual needs of users. Such activities impose a significant threat to social media platforms as these are mostly against the Terms of Service (see Section 1.2.1.2 for more details). Such artificial boosting of social reputation/growth is often known as "*collusion*" in online media [3, 4].

According to a recent survey by HitSearch[1], 98% of content creators admitted to having spotted collusive followers among online influencers on Instagram. The adversarial impact of the collusive entities poses a massive threat to online media. These entities create an atmosphere where people start trusting their information due to the popularity they receive. For example, in the 2019 UK general election, politicians approached the blackmarket services[2] for online political campaigning in order to reach out to their potential voters. A study conducted by LawSuit[3], a law management firm in the United States, reported that around 15% of the Twitter users are non-human accounts driving more than 66% links published to the site. It is also reported that most of the politicians currently in contention or conversation for the 2020 US presidential election have a very high percentage or volume of non-human followers linked to their Twitter account. The above examples show how collusive entities boost the believability of information during events. Moreover, the limitation of humans' potential to distinguish between the collusive and genuine entities due to the vast amount of available information is an important concern that motivates to design methods for automatic identification of these entities. Collusive entities not only deceive people but also pollute the entire social space. Using the blackmarket services, collusive entities can perform appraisals such as improving the credibility of rumors, propagate fake news, inflate/deflate the ratings of products in e-commerce platforms, gain popularity in video-sharing platforms, etc. Our studies tackling the problem of collusive entity detection state that it is harder to discern these entities as they express a mixture of organic and inorganic activities [3, 5]. Moreover, the problem is relatively new as compared to its related studies, such as bot/spam detection and fake account detection and recently has gained significant attention from diverse research communities. In this thesis, we address the following challenges: How can we identify collusive users on Twitter based on metadata properties? How can we additionally leverage the network properties of Twitter users to improve our collusive user detection task? How can we identify collusive users requesting other Twitter appraisals on blackmarket services and how collusion happens on other online media platforms? Furthermore, how can we detect core users of the

[1] https://1tnr.short.gy/HitSearch
[2] https://www.ics-digital.com/general-election-2019-mps-fake-twitter-followers/
[3] https://lawsuit.org/politics-and-fake-social-media-followers/

blackmarket services? Summarily, we provide a large-scale analysis and detection of collusive entities in multiple online media platforms. We also develop state-of-the-art models and create benchmark datasets for collusive entity detection. Below is the thesis statement:

> Modeling multifaceted features of online media entities such as metadata, temporal, textual and network information to detect collusive behavior.

1.2 Background

1.2.1 Overview of online media

1.2.1.1 Online media and its platforms

Online media refers to the technologies present on the Internet that connects people or organizations to exchange information. The reason online media became popular is due to the abundant sources of information offered by Internet i.e. a user can get access to the same news from several places. To keep the user engaging, the online media platforms also enable the users to share their opinion. In today's world, social media is considered to be the fast, inexpensive, and effective way to reach a target audience. The history of online media started at the end of the 19th century with the arrival of Arpanet, email, blogging, and bulletin boards. Prior to that, services such as Telegram, Radio, Telephone, etc. were used to exchange information. However, it never actually contributed much to the community as information exchange in these platforms did not take place 'online' and was mostly used to send individual messages between two people. The rapid growth of the Internet in early 2000 set the real stage for the emergence of online media with the arrival of knowledge sharing sites and social networking sites. Examples of online media platforms include social networking sites (e.g., Facebook, LinkedIn), microblogs (e.g., Twitter, Tumblr), wiki-based knowledge-sharing sites (e.g., Wikipedia), social news sites and websites of news media (e.g., Huffington Post), forums, mailing lists, newsgroups community media sites (e.g., YouTube, Flickr, Instagram) social Q & A sites (e.g., Quora, Yahoo Answers), user reviews (e.g., Yelp, Amazon.com), social curation sites (e.g., Reddit, Pinterest) and location-based social networks (e.g., Foursquare).

1.2.1.2 Terms of service in online media

Terms of service in online media refer to the rules and regulations to be agreed upon by the users of the services. Each service has its own policy against fake and spam engagements - Twitter has Platform manipulation and spam policy[4], YouTube has Fake Engagement Policy[5], etc. These policies ensure that violations of the rules may result in permanent suspension of account and its content. The Terms of service followed by Twitter and YouTube forbids artificially inflating own or others' engagement (followers/retweets/views/likes/subscriptions). This includes selling or purchasing of engagements using premium services, using or promoting third-party apps by posting content that helps in gaining engagements, trading to exchange engagements using freemium services, etc. Some of the terms proposed by Twitter on collusion-related activities are *engagement churn* (first following a large number of unrelated Twitter accounts and then unfollowing them), *indiscriminate engagement* (using third-party APIs or

[4] https://help.twitter.com/en/rules-and-policies/platform-manipulation
[5] https://support.google.com/youtube/answer/3399767?hl=en

automated softwares to follow a large number of unrelated accounts in a short time period) and *aggressive engagement* (aggressively engaging with Tweets to drive traffic or attention to accounts).

1.2.2 Collusion in online media

1.2.2.1 Definition

In general, collusion is defined as a covert and secret consipiracy or collaboration to deceive others. *Collusion in online media* is a process by which users artificially gain social reputation, which violates the 'Terms of Service' of the online media platform. Online media entities involved in collusion often approach blackmarket services to artificially inflate their social status. This results in entities to appear credible and legitimate to the end-users, thus leading to activities such as fake promotions, campaigns, misinformation, etc., thereby creating an inadequate social space. The blackmarket services provide eminent online media services ranging from online social networks to various other platforms such as rating platforms, video-sharing platforms, and even recruitment platforms. For example, a boost in YouTube views can transform a small event into a big campaign or a promotional event. Despite its apparent presence in the real world, collusion has remained as an underexplored concept. We go deeper in the notion of collusion by discussing particular cases and examples in the next subsection.

1.2.2.2 Examples of collusive activities

In this section, we show several examples showing how collusion happens in online media. Fig. 1.1 shows the example of collusive activities in online media. Fig. 1.1(a) shows the official Twitter account of an organization registered in blackmarket services for collusive follower appraisals. Fig. 1.1(b) shows a video posted by a verified YouTube channel registered in blackmarket services for collusive like requests. In both examples, the accounts are marked as *verified* by Twitter/YouTube. The presence of verified online media entities in the blackmarket services clearly shows that the in-house algorithms deployed by these platforms have been unsuccessful in detecting such entities. This further motivates the problem and necessitates the development of automated techniques to detect these entities.

(a) (b)

Figure 1.1: Example of collusive activities in online media. (a) Official Twitter account of an organization registered in blackmarket services for collusive follower appraisals. (b) A video posted by a verified YouTube channel registered in blackmarket services for collusive like requests.

1.2.3 Reasons behind collusion

In order to keep up with the pace of today's turbo-charged world, a large number of users want to go against the stream. Though shortcuts to success always conflict with the enormous efforts needed to

be successful, it may also lead to some costly mistakes, eventually undermining the true goals. Online media is considered as the perfect convergence of communication and information. It has already become the most important source for public information, organizations, and for individuals to promote their ideologies, products, and services. Manipulating such an important source of information can result in a significant gain in terms of fame and finance for individuals/organizations. This is clearly evident in online social networking platforms like Twitter in the form of attaining fake followers, rating platforms like Amazon in the form of posting fake reviews, video streaming platforms like Youtube in the form of synthetic gain in viewership count, and many more. With a huge competition in the online media, it is hard for online media users who spend months to create new content but fail to reach their target audience. Blackmarket services serve as a complete social media toolkit helping an online media entity to gain a stronger social media presence among its competitors. It is mostly helpful for: (i) companies willing to hype a campaign, (ii) politicians promoting new election campaigns, (iii) entertainment companies publicizing shows, and (iv) online media moguls willing to obtain quick online media presence.

1.2.4 Challenges in detecting collusion

Detecting collusive activities is a challenging task. We first show in [3] how collusive users show a mixed behavior of organic and inorganic activities. In the case of Twitter, these users are sometimes involved in appraising organic content and sometimes involved in appraising inorganic content (entities that are submitted to the blackmarket services). This kind of mixed behavior makes it difficult to be detected by existing fake user detection methods [6]. Moreover, collusive users are not bots; they are normal human beings. This makes them difficult to be flagged by bot detection methods [4]. In our study [7], we investigate how collusion happens on YouTube. We design web scrapers to collect a large set of YouTube videos and channels submitted to the blackmarket services for collusive appraisals. However, the dynamic nature of the propagation of collusive activities is very complicated. Collusive activities can easily propagate and impact a large number of users in short time by spreading misinformation. Kim et al. [8] developed `CURB`, an online algorithm that leverages the information from the crowd to prevent the spread of misinformation. They also reported that fact-checking organizations like `Snope` and `Politifact` are not able to properly limit the spread of misinformation as it requires significant human effort. Moreover, as collusion happens in multiple online media platforms, it is also a difficult task to understand the complete adversarial intent of the collusive entities. This raises concern on developing systems for early detection of collusive entities to limit its artificial social growth. Finally, due to the API limitations and restrictions of the online media platforms on collecting the public data, the research community has a very limited training data which do not include all the information related to collusion.

1.2.5 Collusion and other related concepts

In the previous sections, we have defined collusion and presented some examples. Here, we compare collusion with several other related concepts. Some relatable concepts to collusion are spam, fake, bot, cyberbulling and fraud. We distinguish between these concepts and collusion in Table 1.1. Even though these concepts slightly differ from collusion, they are very well related to it.

1.3 Related work

Plenty of studies focused on the detection of fraudulent activities in a wide range of platforms. In recent years, fraudulent activities are most common in major online media sites such as Facebook [14, 15, 16],

Table 1.1: Comparison between collusion and other related concepts.

Concept	Definition of the concept	Difference from collusion
Fake	Fake is to increase the visibility of others' content [9]	Collusion is self-focused i.e. to increase the visibility of own's content
Bot	Bots have synchronous fraudulent activities [1]	Collusive entities have asynchronous fraudulent activities
Sockpuppetry	Sockpuppets are controlled by a puppetmaster who controls at least one other user account [10]	In collusion, every account is controlled by the real owner of the account
Malicious promotion	Promote a specific product/topic for a target audience [11]	Collusion is more general and not necessarily focused on a specific product/-topic
Spam	Consistently perform similar operations across multiple accounts to manipulate or undermine current trends [12]	Collusive entity may contain spammy contents, but not necessarily
Content polluters	Polluters post nearly identical contents, sometimes by randomly adding mentions to unrelated legitimate users [13]	In collusion, the content pollution never occurs as every actions happens through the blackmarket services
Relationship Infiltrators	They follow the reciprocity in relationship (follow/retweet/subscribe) to engage in spam activities [13]	In collusion, reciprocity happens in freemium services to gain credits

Instagram [17, 18] and Twitter [19, 20, 21, 22, 23, 24, 25, 26]. Kumar and Shah [27] presented a review on existing fake and fraud detection strategies in online media platforms. A number of fake comment detection strategies [28, 29] on YouTube were also proposed in recent years. Most of these studies rely solely on the textual data of the comment. Nevertheless, there is no prior work that considers the temporal properties of comments, which in the case of collusion, is the most important factor, the reason being collusive users perform appraisal operations aggressively in order to gain credits rapidly. Li et al. [30] proposed `LEAS`, an algorithm to detect fake engagements in video sharing platforms using a temporal engagement graph between users and video objects. Marciel et al. [31] proposed a set of tools to detect view fraud in online video portals. Chen et al. [32] investigated the problem of fake views caused by robots in video sharing platforms.

In the field of fraud detection in online advertising, Metwally et al. [33] proposed an advertising network model to discover coalitions between pairs of fraudsters in e-commerce platforms. Dave et al. [34] designed an automated approach for ad networks to detect click-spam attacks. Hussain et al. [35] analyzed disinformation and crowd manipulation tactics on YouTube by analyzing video metadata. Alassad et al. [36] examined intensive groups among YouTube commenter networks by constructing a two-level optimization problem for maximizing local degree centrality and global modularity measures. Faddoul et al. [37] conducted a longitudinal analysis of the promotion of conspiracy videos on YouTube. Yang et al. [38] studied the problem of injection attacks on the recommendation systems (fake co-visitations) of YouTube. A number of studies have been conducted on detecting spam on video streaming platforms. Alberto et al. [39] proposed `Tubespam`, a novel classification model for comment spam filtering on YouTube. Uysal [29] studied the performance of five state-of-the-art text feature selection methods for spam filtering on YouTube using Naive Bayes and Decision Tree. Yusof and Sadoon [40] detected video spammers on YouTube based on the EdgeRank algorithm to decide which post/stories should appear in each user's news feed. Chowdury et al. [41] proposed a spam detection system for YouTube using a set of spam-related attributes from videos. Sureka [42] detected forum spammers on YouTube based on the mining comment activity log of a user and extracting patterns indicating spam behavior. Aiyar and Shetty [43] detected spam comments on YouTube by showing the effectiveness of using character n-grams instead

of word n-grams to improve the accuracy of the classification model. More recently, YouTube has become the most important tool for live streamers. In our dataset, we found around 30% of the collusive channels involved in streaming live videos. Zhang et al. [44, 45] discussed the detection of copyright infringement on YouTube live videos. Zhang et al. [44] developed a crowd-sourced based copyright infringement detection (CCID) scheme from live chat messages on YouTube and Twitch to identify original copyright content from the owner. Zhang et al. [45] presented an end-to-end supervised detection framework to combat copyright infringement in live video streams using the live chat messages from the audiences. However, these methods tend to combat the problem of fake, fraud, and spam detection in video-sharing platforms but are not applicable for collusive entity detection.

Despite the fact that a plethora of studies exist on detecting fraud/spam activities in online media, there has been relatively less work on investigating blackmarket services providing collusive appraisals. Acker [46] focused on how manipulators create disinformation by fake engagement activities on YouTube. Keller [47] provided a broad overview of the blackmarket services supplying fake YouTube views. The authors reported that one of the blackmarket services, named Devumi had earned more than $1.2 million in around 3 years of service by selling 196 million views. Shah et al. [6] studied multiple types of blackmarket link fraud behaviors in Twitter by analyzing the connectivity patterns of fake followers via the egonet and boomerang networks. Arora et al. [48] proposed a multi-task learning approach to detect tweets submitted to freemium blackmarket services. Dhawan et al. [49] proposed `DeFrauder`, an unsupervised method to spot online fraudulent collusive groups in review websites. [50] proposed a review graph model to detect spammers in online review stores. Zhu et al. [51] proposed an automated approach to detect collusion behavior in online question-answering systems. Other studies identified fake followers on Twitter [9, 52, 53, 54]. Mehrotra et al. [55] and Jiang et al. [56] are some of those who used network-centric properties to detect fake followers. `Fake Follower Check`[6] is one such tool to detect fake followers based on profile-centric and behavioral features of Twitter users.

1.4 Thesis organization

We now discuss the organization for the rest of the thesis. The entire thesis is divided into four parts:

- **Modeling metadata of collusive users:** In Part 1, we propose two approaches (`SCoRe` and `HawkesEye`) based on metadata properties of Twitter users to detect whether a user is collusive or not. We also explore different facets of blackmarket services where we show how the working of premium blackmarket services differ from freemium blackmarket services.
- **Modeling network of collusive users:** In Part 2, we aim to improve the collusive Twitter user detection task. We propose a multiview learning based approach to detect collusive retweeters and show how each view can be helpful for our task. As labeling collusive entities is a difficult and tedious task, we further propose `CoReRank`, an unsupervised framework to detect collusive users and suspicious collusive tweets.
- **Collusion on other Twitter appraisals and online media platforms:** In Part 3, we look into another type of Twitter appraisal (followers) that are artificially inflated by blackmarket services. We propose `DECIFE`, a framework to detect collusive users involved in producing following activities through blackmarket services with the intention to gain collusive followers in return. As artificial boosting has become a regular practice with the increasing popularity of different online media platforms, we further propose `CollATe`, a novel end-to-end framework to identify YouTube videos and channels submitted to blackmarket services for collusive appraisals.
- **Core users in blackmarkets and released datasets:** In Part 4, we first look into the core users of the

[6] https://1tnr.short.gy/socialbakers

blackmarket services and show how they have different characteristics than compromised users. We then present ABOME, a multi-platform data repository that consists of data related to blackmarket-driven collusive entities collected from two credit-based freemium services.

Part I

Modeling metadata of collusive users

2. Detecting collusive retweeters using metadata properties

Twitter has increasingly become a popular platform to share news and user opinion. A tweet is considered to be important if it receives high number of affirmative reactions from other Twitter users via Retweets. *Retweet count* is thus considered as a surrogate measure for positive crowd-sourced reactions – high number of retweets of a tweet not only help the tweet being broadcasted, but also aid in making its topic trending. This in turn bolsters the social reputation of the author of the tweet. Since social reputation/impact of users/tweets influences many decisions (such as promoting brands, advertisement etc.), several blackmarket syndicates have actively been engaged in producing fake retweets in a collusive manner. Users who want to boost the impact of their tweets approach the blackmarket services, and gain retweets for their own tweets by retweeting other customers' tweets. Thus they become customers of blackmarket syndicates and engage in fake activities. Interestingly, these customers are neither bots, nor even fake users – they are usually normal human beings; they *express a mix of organic and inorganic retweeting activities*, and *there is no synchronicity across their behaviors*.
In this work, we make a first attempt to investigate such blackmarket customers engaged in producing fake retweets. We collected and annotated a novel dataset comprising of customers of many blackmarket services and characterize them using a set of 64 novel features. We show how their social behavior differs from genuine users. We then use state-of-the-art supervised models to detect three types of customers (bots, promotional, normal) and genuine users. We achieve a Macro F1-score of 0.87 with SVM, outperforming four other baselines significantly. We further design a browser extension, `SCoRe` which, given the link of a tweet, spots its fake retweeters in real-time. We also collected users' feedback on the performance of `SCoRe` and obtained 85% accuracy.

2.1 Introduction

Twitter, arguably the most popular micro-blogging site, provides its users two major ways to place their affirmative reactions towards different entities: (i) user-level affirmation (such as `follow`), and (ii) content-level affirmation (such as `retweet`, `like`). Here, we particularly focus on 'retweeting activity', a major content-level affirmative action, which provides a way of re-broadcasting messages and confirms retweeter's agreement of the message being important to others. Retweet count of a tweet gives the OSN users a sense of crowdsourced agreement on the tweet, and thus determines the influence of the tweet as well as the author of the tweet. It also helps in making a certain topic trending on Twitter. This further brings an opportunity of 'intentional manipulation' to the social adversaries who falsely create the impression of popularity of tweets by generating huge volume of retweets.

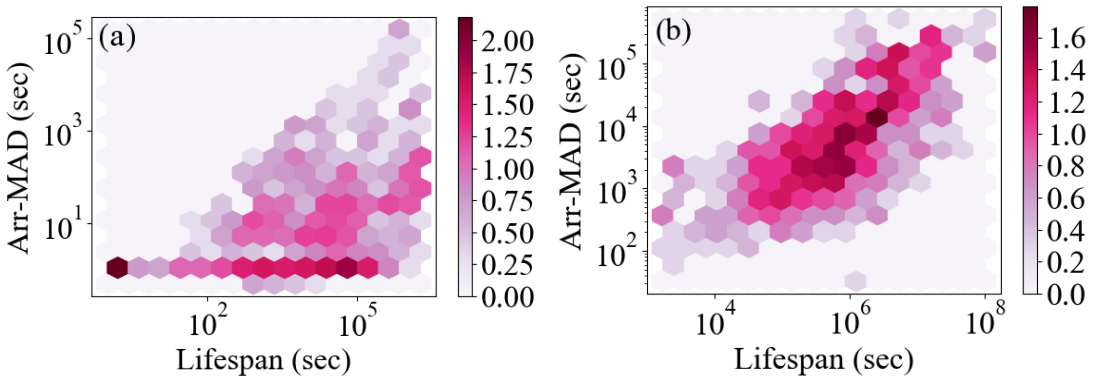


Figure 2.1: (b) Asynchronous behavior of collusive retweeters as opposed to the (a) synchronicity of normal retweet fraudsters mentioned in [1]. The figure shows two binned 2-D heatmaps (in logarithmic scale) of *Lifespan* (time elapsed between the first and last retweets) vs. *Arr-MAD* (mean absolute deviation of retweets' inter-arrival times) of retweet threads for (a) normal retweet fraudsters (data taken from [1]) and (b) collusive retweeters. Unlike (a) where *Arr-MAD* is almost invariant with *Lifespan*, (b) shows an increasing trend.

Table 2.1: Macro F1-score of the competing methods.

Bot Detection [57]	**Fake Account Detection [58]**	**Sync. Fake Retweeter Detection [1]**	**Our Method**
0.760	0.693	0.750	0.873

Challenges in detecting collusive retweeters: There exist several blackmarket agencies which have created thriving and intelligent ecosystems of producing spam retweets. Users can gain retweet count of their own tweets for free by retweeting tweets of other customers of those services. Thus users unwillingly become customers of blackmarket services. Such trend of inflating social reputation has become prevalent across different social media platforms. Detection of such blackmarket retweeters is challenging for many reasons – (i) They are not bots, but human beings; therefore, bot detection algorithms can not flag them (first column of Table 2.1). (ii) Their Twitter accounts are not fake; therefore, fake account detection algorithms can not detect them (second column of Table 2.1). (iii) They express a *mix of organic and inorganic behavior* in their retweeting patterns – they organically reweet some genuine tweets; at the same time, they inorganically retweet tweets submitted to blackmarket services. The extent of inorganic activities may differ across retweeters. (iv) They do not show any synchronicity across their retweeting patterns, thus making it difficult for the existing synchronous fake retweeter detection methods [1] to detect them (third column of Table 2.1).

Motivation and state-of-the-art: Despite previous efforts in understanding different fraudulent activities in Twitter such as fake account detection [58], social spam detection [24], content affirmation via retweeting has hardly been studied. A preliminarily attempt was made by [1] to detect spam retweeters with the hypothesis that suspicious activities are highly 'synchronized', i.e., a group of spammers retweet at the same time. However, we clearly observe that **synchronicity does not hold among collusive retweeters** (Figure 2.1) – this is intuitive since the blackmarket services do not have control on the customers involved in producing fraud retweets. A very recent effort was made by [59] to detect collusive followers in Twitter. They concluded that crowdsourced manipulation is a big threat to the credibility of OSNs. **However, to the best of our knowledge, ours is the first attempt to detect blackmarket-based collusive manipulation of content credibility via fraudulent retweeting activities.** Interestingly, Twitter has not yet been successful in flagging these customers – we observed 31 blackmarket customers who are marked as 'verified users' by Twitter.

Our Contributions: We begin our experiment by surveying different types of blackmarket services (Section 2.3.1) and collecting customers from multiple blackmarket services (Section 2.3.2). We employed human annotators to label each customer into one of the following categories – *bots*, *promotional* and *normal*. We also collected genuine users who are researchers / experts in machine learning, following the method in [6]. The timeline information of these users (customers and genuine users) were further scraped. These two sets of users thus form the retweeter set which we further analyze. We then propose an exhaustive set of 64 novel features to characterize retweeters (Section 2.4). We run several state-of-the-art supervised models to classify retweeters into four types – bot, promotional customer, normal customer, genuine user. We observe that Logistic Regression achieves the maximum accuracy, outperforming the baseline methods significantly (Section 2.5.3). Our method (with SVM) turns out to be equally successful in categorizing retweeters into binary classes – customers and genuine (Macro F1-socre of 0.87).

We finally build a browser extension, `SCoRe`, '**S**potting **Co**llusive **Re**tweeters', which, given a retweeter set of a tweet, classifies each retweeter as customer or genuine. The extension allows the end users to judge the quality of the classification `SCoRe` produces and provide correct labeling, in case it produces wrong result. The user-generated feedback is collected in the back-end and the supervised model is retrained in an incremental way. However, special care has been taken to ignore spurious feedbacks (more details in Section 2.6). We seek feedback of 25 volunteers on the performance of `SCoRe` and obtain 85% accuracy. We believe that the widespread use of such extension would help OSN administrators block the customers and measure the true credibility of a tweet based on its genuine retweet count.

Reproducibility: All the codes and processed dataset are available at `https://github.com/LCS2-IIITD/collusive-retweeters-ASONAM-2018`.

2.2 Related work

Here, we divide the related work into two parts – (i) detection of fraudsters in OSNs, and (ii) detection of blackmarket customers.

Detection of fraudsters in OSNs: Many studies have been conducted on detecting frauds in Twitter and other OSNs. [60] detected spammers in Twitter using features generated from tweet content and user social behavior. [61] studied the embedded URLs in Twitter considering correlated redirect chains of URLs in a number of tweets. [62] identified strangers on Twitter, who can be potential retweeters to help effectively propagate intended information within a desired time frame. [63] focused on detecting inorganic retweet behavior and proposed 'RTGEN', a realistic generator that imitates the behaviors of both honest and fraudulent users. [64] used machine learning approach to classify Twitter accounts into 'human', 'bot' and 'cyborg'. [65] discussed the rise of social bots on Twitter. They reported that social bots are responsible for stealing personal information, spreading misinformation and even manipulating the stock market prices. [66] used network properties to identify spammers on Twitter. [24] proposed a spam detection technique to identify suspicious users on Twitter. [22] studied 'link farming' on Twitter – it refers to a form of spamming on Twitter accounts that connect each other in a group. A recent study by [67] examined automated bot accounts on Twitter distributing links to scientific articles. [23] studied multiple spamming techniques, including creating fake Twitter accounts, generating spam URLs, and spam distribution. [68] studied malicious crowd-sourcing spammers in multiple OSNs. [69] used synchronous and abnormal behaviors of Twitter followers to detect suspicious following behavior. [70] detected spam campaigns on Twitter. [71, 72] studied the issues of 'identity', 'slacktivism' and 'puppetry' in micro-blogging platforms. [73] proposed a new ranking method for collusive group detection in online auctions and applied it to a real-world online auction site.

Detection of blackmarket customers: Understanding the dynamics of blackmarket services have recently gained substantial attention among researchers. [74] provided a detailed analysis of blackmarket services and their impact on multiple OSNs. Most of the prior works showed how fake followers in social media help in promoting a certain user [54]. [6] studied multiple types of blackmarket agencies by showing their multifaceted behavior. [75] studied various blackmarkets tied to fraudulent Twitter credentials, monitoring pricing, availability, and fraud perpetrated by these services. [76] tackled the problem of voluntary following activity detection to automatically detect malicious accounts that make profit in the follower markets. [77] performed an analysis of six different underground forums and showed how OSN users are engaged in selling goods and services. [78] introduced `FBOX`, a scalable method to spot fraudsters and their customers in online social networks and web services. [79] proposed a power law approach to estimate fake social network accounts on Twitter. [80] proposed a novel fake account detection method to identify fake followers on Weibo. [81] proposed a sampling-based method, called star sampling, to identify fake followers on Weibo.

Remarks. Our methodology differs from the existing approaches in many ways: (i) [1] identified fake retweeting behavior in Twitter on the basis of synchronicity. This type of behavior can only be observed if the retweeters are controlled by a centralized authority. However, collusive blackmarket customers are not controlled by anyone. They intentionally retweet other customers' tweets. Therefore, they do not exhibit any synchronized behavior. (ii) Existing fraud detection approaches classify a user into 'fraud' or 'genuine' [1, 6]. Our work classifies users into four categories – 'genuine', 'bots', 'promotional customers' and 'normal customers'. (iii) We are also the first to develop a real-time system that can identify customers and genuine retweeters, given the retweeter list of a tweet.

2.3 Blackmarket services

In this section, we describe our efforts in collecting and annotating a set of Twitter accounts corresponding to the customers of various blackmarket agencies.

2.3.1 Types of blackmarket retweet services

While investigating the mode of blackmarket services, we noticed that there are two prevalent models of services [6] – (i) ***Premium Services*** which only provide services upon receiving payment from the customers; (ii) ***Freemium Services*** which, similar to premium services, offer both paid services, as well as unpaid services that require the users to provide their Twitter login details; this in turn may involve the users unconsciously in the blackmarket activities. Here our primary focus is to understand the *activities of freemium services* which are easy to access due to their unpaid service model. This type of services can further be divided into three categories:

- **Social-share services:** These services (e.g., FreeFollowers[1]) ask customers to perform activities on social-media contents of other customers involved in those services. The activities on social-media contents can be 'Facebook Share', 'Facebook Like', 'Twitter Retweets' etc.
- **Credit-based services**: These services (e.g., Like4Like, YouLikeHits, TraffUp, JustRetweet) work on credit-based policies. Each customer has to put a value for the tweet, which gets deducted from his/her credits when the tweet is published. A customer retweets the tweets of other customers to earn credits and the value of each tweet is added up to his/her total credit.

[1]FreeFollowers: https://www.freefollowers.io/, Like4Like: https://like4like.org/, TraffUp: http://traffup.net/, JustRetweet: http://justretweet.com.

Table 2.2: Statistics of the dataset.

Service	# users	# users suspended / deleted	# users taken for our analysis	# tweets our dummy accounts retweeted
YouLikeHits	638	168	470	914
Like4Like	451	178	273	459
TraffUp	2	0	2	6
JustRetweet	11	3	8	1
Genuine Users	1000	0	1000	-

- **Auto-time retweet services**: The customers of these services (e.g., TweetsTool[2]) need to get access token from Twitter and login to the service. They can request for 10-50 retweets for each of their tweets in a 15-minute window. Other than the retweet service, it also provides the customers with the following services – 'Auto Favourite', 'Auto Follower', 'Auto Reply'.

2.3.2 Data collection

Collecting blackmarket customers: In order to collect the information of blackmarket customers, we focused our crawling only on credit-based services[3] because – (i) their service policies are easy to understand, (ii) as opposed to the social-share services, they only utilize a single platform (Twitter) to perform their activities, (iii) most of the freemium services follow credit-based strategy compared to the other strategies which would help us in collecting more data for our analysis. We adopted an 'active probing' strategy to collect the dataset – we created multiple dummy accounts in each of the credit-based freemium services (Like4Like, YouLikeHits, TraffUp, JustRetweet), kept retweeting tweets of other customers whose tweets were posted on the blackmarket services, and collected their IDs. We continued these activities for one month (Feb, 2018). Table 2.2 shows the statistics of the dataset for each service. During this process, Twitter suspended some of the accounts we collected, and therefore we ignored those accounts for further analysis. From the collected dataset, we noticed that the average number of tweets posted by a user to YouLikeHits is 1.43 which is relatively high compared to Like4Like (1.01). Interestingly, we found a significant number of customer overlap (76 customers) across YouLikeHits and Like4Like. It suggests that customers utilize several such blackmarket services audaciously without any obfuscation of their identities to promote their tweets. We further scraped the timeline information of these customers using Twitter's REST API.

Collecting genuine users: To compare normal users with blackmarket customers in Twitter, we selected 1000 genuine users (following [6]) – (i) We searched for Twitter lists containing users who are researchers, working in the field of Machine Learning/Data Mining/Information Retrieval. (ii) We also discarded genuine users who have more than 1 million followers since high follower count may resemble a celebrity and we wanted to discard any celebrity-like users from our analysis in order to remove any unnecessary bias. We further scraped the timelines of the remaining users.

[2]http://www.tweetstool.com/

[3]We also believe that the other two types services can be helpful to study how cross-platform collusive activities happen in online media.

2.3.3 Human annotation

Three human annotators[4] were asked to label the blackmarket customers into *Bots*, *Promotional Customers* and *Normal Customers* based on our definition of each customer type and Twitter's terms of service. Annotators were also given freedom to search for any information related to customers and apply their own intuition. Here, we describe the information we provided to the annotators about three types of customers:

C1. **Bots:** A Twitter Bot is a software which controls a Twitter account using the Twitter API [64]. Bots on Twitter perform both helpful and harmful activities working in a coordinated fashion [57].

C2. **Promotional customers:** Twitter has always been a source of advertising content to its target users. Promotional users are involved in promoting one of the three products: *tweets*, *accounts* or *services* [82]. The annotators were asked to search for tweets in the timeline of customers promoting one of the above products. We observed that many customers in our dataset were involved in promoting brands using keywords such as 'win', 'ad', 'Giveaway'.

C3. **Normal customers:** These customer do not fall under any of the two categories mentioned above.

Each annotator was given all the customer accounts for annotation. Finally, we considered only 743 customers whose labels were agreed upon by at least two annotators, and found 86 bots, 275 promotional customers and 382 normal customers. The average inter-annotator agreement was 0.79 based on Cohen's κ.

2.3.4 Interesting observations

- Customers of the blackmarket services often remove their tweets after a small duration. We found that around 3% customers have removed their tweets. This is obvious as deleting these retweets in quick time may help them evade the in-house fake detection algorithms deployed by Twitter.
- We also found 1% customers who are suspended by Twitter. This clearly indicates that Twitter is still inefficient in identifying customers of the blackmarket services.
- We found 31 customers who are marked as verified genuine accounts by Twitter. This also indicates the inability of current Twitter policy to flag blackmarket customers.
- The Twitter profiles of 4% customers are shown with the warning *"Caution: This account is temporarily restricted"*. Twitter enforces this warning on users in regards to either of the following cases : (i) repeatedly posting duplicate or near-duplicate content, (ii) abusing trending topics or hashtags, (iii) sending automated tweets or replies, and (iv) using bots or applications to post similar messages based on keywords, (v) posting similar messages over multiple accounts, and (vi) aggressively following and un-following people.

2.4 Experimental setup

In this section, we present novel features to characterize different types of users (both customers and genuine users) and supervised models for classifying users.

[4]They were experts in social media, and their age ranged between 25-35.

[4]https://maximizesocialbusiness.com/why-your-twitter-account-may-be-restricted-1169/

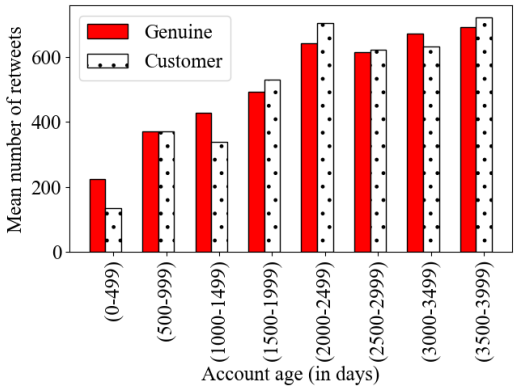


Figure 2.2: Relationship between retweet count and account age (in days; further divided into bins: bin 1: 0-499, bin 2: 500-999,$\cdots$, bin 8: 3500-3999).

2.4.1 Feature selection

We use an extensive set of 64 novel features to detect different types of users. We group these features into five buckets: (i) Profile Features (PF), (ii) Social Network Features (SNF), (iii) User Activity Features (UAF), (iv) Likelihood Features (LF), and (v) Fluctuation Features (FF).

(i) Profile Features (PF): We hypothesize that the older the account of a user, s/he will have more tendency to retweet others' tweets. Therefore, we take *account age* (**PF$_1$**) as one feature in our study. Figure 2.2 corroborates our hypothesis by showing that both genuine users and customers exhibit an increasing trend of retweet count w.r.t their account age. We further consider four other profile features: (**PF$_2$**) *length of the screen name*, (**PF$_3$**) *whether the profile has a description or not*, (**PF$_4$**) *length of profile description*, (**PF$_5$**) *whether profile has a URL or not*.

(ii) Social Network Features (SNF): Social network features such as *number of followees* (**SNF$_1$**) *and followers* (**SNF$_2$**) and *their ratio* (**SNF$_3$**) sometimes provide indicative information about the users. Figure 2.3(a) shows that customers tend to follow a lot of other users, compared to the genuine users – this may be explained by the typical tendency of customers to follow others with the intention that the followees will further draw attention to their profiles as well as their tweets. Figure 2.3(a:inset) shows that followee count has a more positive correlation (Spearman's $\rho = 0.58$) with retweet count in case of genuine user than the customers (Spearman's $\rho = 0.42$). Figure 2.3(b) shows that genuine users tend to have a variety of follower count, whereas all customers have a smaller follower count. Figure 2.3 (b:inset) shows that genuine users with a large number of followers tend to retweet more compared to those with less followers; however, it may not hold for the customers. The ratio of followees to followers in Figure 2.3(c) shows that genuine users have a lower followee to follower ratio – this is expected as they tend to have around equal number of followees and followers (recall that we excluded celebrities from our study, whose follower count may be much higher than the followee count). For genuine users, we also observe a declining trend of retweet count with the increase of followee to follower ratio; however customers do not follow any such trend (Figure 2.3 (c:inset)). We further measure the influence score of users using *Klout score*[5] (**SNF$_4$**) and take it as a feature. Klout score returns a value between 1-100 to rate a user based on his/her online social influence. The Klout score of genuine users and customers based on their influence is shown in Figure 2.3(d). The median value of Klout score for customers and genuine users is 48.11 and 43.57 respectively – such small difference in Klout score indicates that though customers use blackmarket services, it does not help them making their profile significantly popular. Fig. 2.3 (d:inset) shows that the retweet count tends to increase with the increase of Klout score for genuine users; however there is not such uniform pattern for customers.

[5]http://klout.com

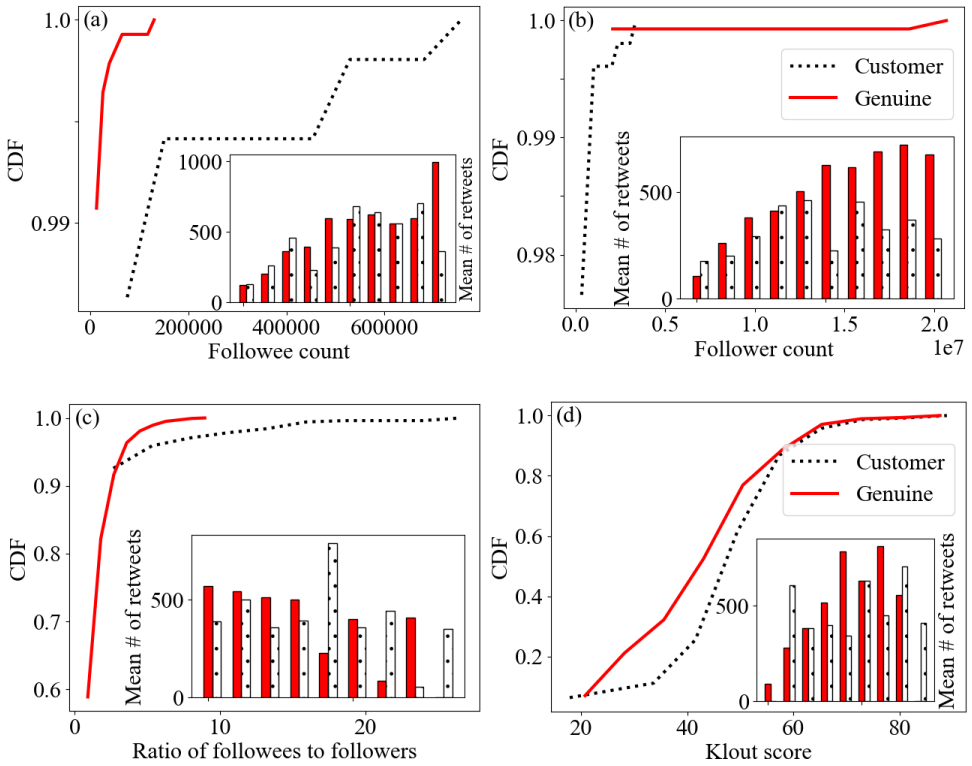


Figure 2.3: CDF of social network features – (a) followee count, (b) follower count, (c) followee to follower ratio, (d) Klout score, and their relation with the retweet count (inset). The variable in x-axis is same for the main diagram and its inset.

(iii) User Activity Features (UAF): User activity plays a significant role - higher the activity of a user, higher is the chance of his/her tweet to be retweeted by a stranger. We use the following features to measure the activity of users: (**UAF$_1$**) *total number of tweets*, (**UAF$_2$**) *number of direct mentions per tweet*, (**UAF$_3$**) *number of URLs per tweet*, (**UAF$_4$**) *number of hashtags per tweet*, (**UAF$_5$**) *number of tweets per day*, (**UAF$_6$**) *number of retweets per day*, and (**UAF$_7$**) *number of retweets per tweet*. To compute these features, we crawl all (max 3200) tweets from the timeline of each user. We also use *Bot-score* (**UAF$_8$**) of each user from Botometer service[6] [57] and consider it as a feature.

(iv) Likelihood Features (LF): If Twitter users want to publish tweets in a credit-based blackmarket service, they need to retweet a lot of others' tweets to earn credits for their own tweets to be retweeted by other customers. On the contrary, when they write tweets on Twitter and do not submit these tweets to any blackmarket services, they are not expected to perform such credit-based retweeting activity. We use the following set of features to capture this phenomenon: (**LF$_{1\text{-}7}$**) *tweeting likelihood per day for seven days* (*Monday-Sunday*), (**LF$_{8\text{-}14}$**) *retweeting likelihood per day for seven days*, (**LF$_{15\text{-}21}$**) *regularity of tweeting activity per day for seven days*, (**LF$_{22\text{-}28}$**) *regularity of retweeting activity per day for seven days*, (**LF$_{29}$**) *tweet steadiness*, (**LF$_{30}$**) *retweet steadiness*, (**LF$_{31\text{-}37}$**) *maximum tweet likelihood per day for seven days*, and (**LF$_{38\text{-}44}$**) *maximum retweet likelihood per day for seven days*. **LF$_{1\text{-}7}$** (*resp.* **LF$_{8\text{-}14}$**) is calculated by taking the ratio of the tweets (*resp.* retweets) of a user per day to the total number of tweets (*resp.* retweets) the user posted in a week. Regularity of tweeting activity per day (**LF$_{15\text{-}21}$**) is calculated by $-\sum_{i=1}^{24} p(x_i) \log p(x_i)$, where $p(x_i)$ is the fraction of tweets posted by the user at i^{th} hour of that day. We follow the same method to measure **LF$_{22\text{-}28}$** by replacing tweets with retweets. Tweet

[6] https://botometer.iuni.iu.edu

Table 2.3: Performance of different competing methods for multi-class classification.

Classifier	Micro				Macro			
	Precision	**Recall**	**F1**	**ROC-AUC**	**Precision**	**Recall**	**F1**	**AUC**
`BotoM` (RF)	0.638	0.638	0.638	0.853	0.497	0.596	0.540	0.780
`SpamBot`	0.565	0.565	0.565	0.658	0.321	0.308	0.287	0.576
`FakeAcc` (LR)	0.568	0.568	0.568	0.809	0.358	0.383	0.347	0.727
`NDSync` (LR)	0.573	0.573	0.573	0.764	0.383	0.293	0.269	0.545
Decision Tree	**0.754**	**0.754**	**0.754**	0.706	0.636	0.596	0.587	0.712
K-NN	0.693	0.693	0.693	0.862	0.666	0.519	0.544	0.779
Logistic Regression	**0.754**	**0.754**	**0.754**	**0.924**	**0.731**	**0.641**	**0.671**	**0.909**
Naive Bayes	0.518	0.518	0.518	0.712	0.129	0.250	0.170	0.724
SVM	0.746	0.746	0.746	0.918	0.592	0.558	0.557	0.896
Random Forest	0.668	0.668	0.668	0.863	0.658	0.637	0.639	0.833
Bagging	0.683	0.683	0.683	0.904	0.689	0.707	0.691	0.884
Boosting	0.693	0.693	0.693	0.892	0.666	0.519	0.544	0.869

Table 2.4: Performance of different competing methods for binary classification.

Classifier	Micro				Macro			
	Precision	**Recall**	**F1**	**ROC-AUC**	**Precision**	**Recall**	**F1**	**AUC**
`BotoM` (DT)	0.770	0.770	0.770	0.760	0.760	0.760	0.760	0.760
`SpamBot`	0.680	0.680	0.680	0.660	0.727	0.665	0.646	0.660
`FakeAcc` (SVM)	0.696	0.696	0.696	0.701	0.704	0.701	0.693	0.701
`NDSync`	0.595	0.595	0.595	0.573	0.576	0.573	0.573	0.573
Decision Tree	0.859	0.859	0.859	0.633	0.640	0.632	0.622	0.632
K-NN	0.799	0.799	0.799	0.795	0.814	0.795	0.795	0.795
Logistic Regression	0.859	0.859	0.859	0.795	0.860	0.858	0.858	0.795
Naive Bayes	0.518	0.518	0.518	0.500	0.259	0.500	0.341	0.500
SVM	**0.873**	**0.873**	**0.873**	**0.873**	**0.874**	**0.873**	**0.873**	**0.873**
Random Forest	0.782	0.782	0.782	0.786	0.797	0.785	0.780	0.786
Bagging	0.695	0.695	0.695	0.698	0.709	0.698	0.684	0.698
Boosting	0.799	0.799	0.799	0.873	0.814	0.795	0.795	0.873

(*resp.* retweet) steadiness is calculated by $1/\sigma_t$ (*resp.* $1/\sigma_{rt}$) where σ_t (*resp.* σ_{rt}) is the standard deviation of time difference between consecutive user-generated tweets (*resp.* retweets). $\mathbf{LF_{31\text{-}37}}$ (*resp.* $\mathbf{LF_{38\text{-}44}}$) is calculated by the ratio of per-day tweet (*resp.* retweet) count of a user to the maximum number of tweets or (*resp.* retweets) the user posted in a day of a week.

(v) Fluctuation Features (FF): Customers of credit-based blackmarket services want to acquire as many credits as possible in order to gain more retweets to their own tweets. We use the following features to validate this phenomenon: ($\mathbf{FF_1}$) *standard deviation of retweet counts for all user-generated tweets*, ($\mathbf{FF_2}$) *mean of log-time difference between consecutive retweets*, ($\mathbf{FF_3}$) *standard deviation of log-time difference between consecutive retweets*.

Note that unlike [83], we did not use any graph-related features for two reasons: (i) crawling the complete neighborhood structure of users requires huge computational resources and sometimes produces incomplete information due to several constraints such as restriction of API and user profiles, (ii) given an unknown user account, collecting its neighborhood structure is time-consuming, which in turn may affect scalability of the real-time system we intended to build (Section 2.6). However, we do not claim that graph-based features will not enhance the classification performance.

2.4.2 Classification models

We consider six state-of-the-art stand-alone supervised classifiers – Decision Tree (DT), K-Nearest Neighbors (K-NN), Logistic Regression (LR), Naive Bayes (NB), Support Vector Machine (SVM). We also consider three ensemble classifiers: Random Forest (RF), Bagging (BG) and Boosting (BO). The reason to use classifier ensemble is due to the fact that it leads to more efficient performance as compare to single models and is tremendously accurate from a set of models that are individually mediocre. We perform hyper-parameter optimization in order to find the parameters that generate the best results. For instance, we use CART with Gini gain criteria for DT; K-NN method with K = 5; multinomial logistic regression and SVM with linear kernel.

2.5 Experimental results

In this section, we start by briefly describing four baselines, followed by the detailed experimental results.

2.5.1 Baseline methods

Since there is no prior work on collusive retweeter detection, we choose four state-of-the-art methods as baselines, which are close to the problem we want to solve here.
Baseline I: `BotoM`: We use the social bot detection method proposed by [57] and measure bot score for each retweeter. This score is further used as a feature to run different supervised classifiers mentioned in Section 2.4.2.

Baseline II: `SpamBot`: We use the method proposed by [84] as our second baseline. It leverages a set of content-based and network-based features to detect spam bots from a collection of Twitter users. It assumes that *a genuine user is less likely to post duplicate tweets as compared to a bot*. To compare our method with this baseline, we use their proposed set of features and their suggested classifier (Naive Bayes) to classify retweeters as 'Genuine', 'Bots', 'Promotional' and 'Normal'.

Baseline III: `FakeAcc`: [58] proposed a fake account detection model in Twitter based on minimum weighted feature set. It first measures gain ratio of 22 features and considers those features whose ratio crosses a certain threshold. Several classifiers are used, among which LR and SVM turned out to be the best for multi-class and binary classification respectively.

Baseline IV: `NDSync`: The closest baseline of our method is `NDSync` [1] which identifies multiple synchronous patterns across spam retweet threads. It extracts a set of features related to retweet threads (such as number of retweets, lifespan, response time etc.), projects retweet threads into a multi-dimensional feature space, and segments the feature space. It then estimates the suspiciousness score of each thread, and combines these scores for all the retweet threads associated with a user to obtain a user-level score. `NDSync` is an unsupervised method to classify retweeters as 'genuine' and 'fake'. We use this method for binary classification. However, for multi-class classification, we utilize the suspicious user score returned by `NDSync` as a feature for supervised classifiers.

2.5.2 Evaluation setup

We combine the customers collected from blackmarket services (see Table 2.2), yielding 743 customers (comprising of 86 bots, 275 promotional and 382 normal customers). We also sample 743 users randomly

from a set of 1000 genuine users we collected, which makes the two classes balanced. The accuracy of each competing method is measured using the following metrics: Precision, Recall, F1-score, and Area under the ROC curve (AUC). Since multi-class classification in involved, we report all these metrics in both Micro and Macro settings (e.g., Micro-Precision, Macro-Precision etc.). We report the accuracy after averaging the result of 10-fold cross validation.

2.5.3 Results of multi-class classification

We design the first experimental setup as a four-class classification problem (genuine, bot, promotional, normal). Table 2.3 shows the results for the four-class classification. We obtain the best result of `BotoM` with Random Forest. Among four baselines, `BotoM` performs the best across all evaluation metrics. However, with our feature set, Logistic Regression achieves the maximum accuracy – it achieves 18% and 8.3% higher Micro-F1 and Micro AUC respectively, and 24.25% and 15.38% higher Macro-F1 and Macro AUC respectively w.r.t to the best baseline. The class-wise F1-score of Logistic Regression is as follows: 0.89 (genuine), 0.80 (bot), 0.58 (promotional), and 0.68 (normal).

2.5.4 Results of binary classification

One may only be interested to identify whether a retweeter is a customer or a genuine users, instead of a fine-grained classification of customers. Therefore, we conduct another set of experiments by combining all types of customers into a single class, and consider the problem as a binary classification problem. In Table 2.4, we notice that `BotoM` with Decision Tree turns out to be the best baseline. However, this time SVM turns out to be the best model with a score of 0.873 for all metrics, which is followed by Logistic Regression. SVM beats the best baseline by 11.3% and 11.3% in terms of Micro-F1 and Micro AUC respectively, and by 10.3% and 11.3% in terms of Macro-F1 and Macro AUC respectively. The class-wise F1-score of Logistic Regression is as follows: 0.87 (genuine), 0.88 (customer).

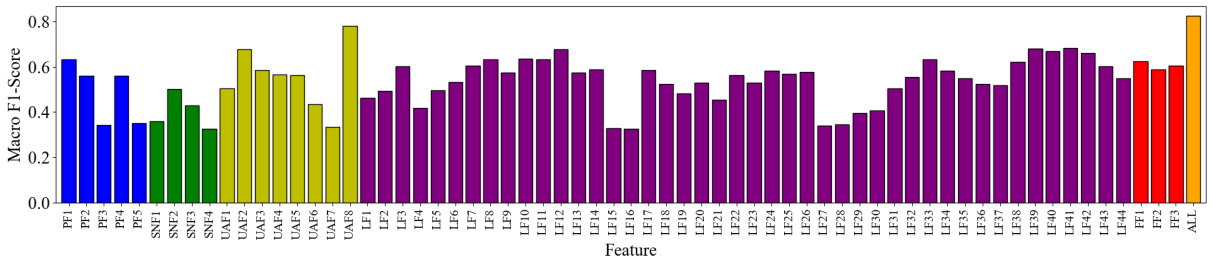


Figure 2.4: Feature importance (accuracy of SVM for binary classification considering each feature in isolation). We also plot the accuracy with all features at the right-most bar of the figure (labeled as 'All').

2.5.5 Feature importance

Figure 2.4 shows the importance of 64 features. Here we take each feature in isolation and run the best binary-classification model (SVM)[7]. The most important feature seems to be Bot-score (UAF_8) with which SVM achieves a Macro F1-score of 0.75. It also corroborates with the significantly high accuracy of our first baseline model (`BotoM`). The second and third ranked features are – maximum retweet likelihood in every Tuesday (LF_{39}) and in every Friday (LF_{42}) respectively. One possible reason why LF_{39} and LF_{42} have better discriminatory power is that these blackmarket services refresh their tweet database every

[7]The pattern of feature importance is same for the multi-class classification with Logistic Regression.

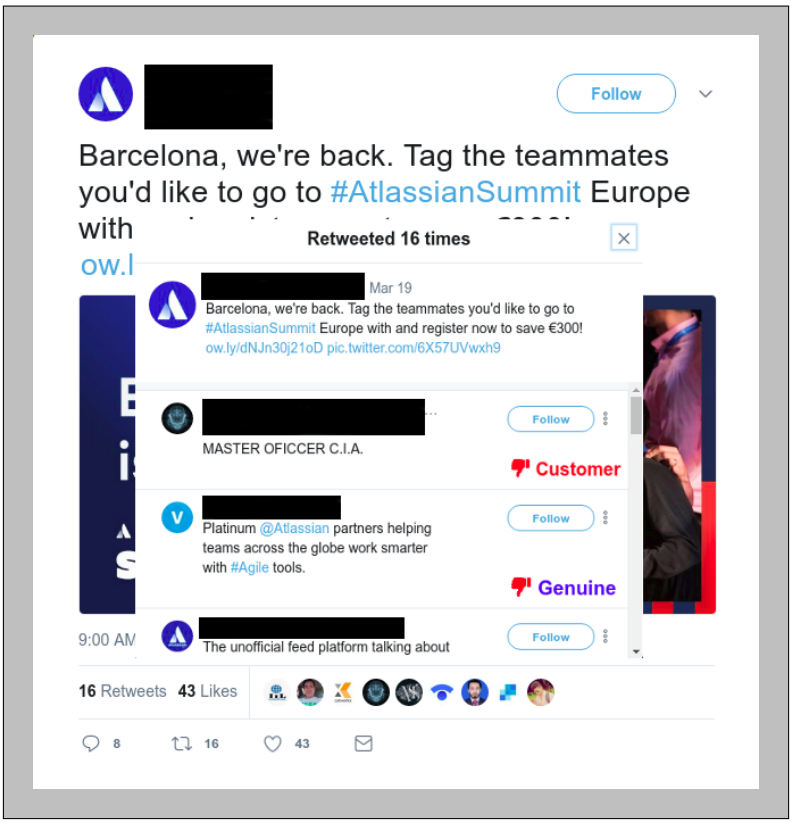


Figure 2.5: Browser extension: SCoRe.

3-4 days (possibly every Tuesday and Friday). Thus, in order to keep their credit high, customers must keep on retweeting tweets every 3-4 days. However, as a whole, fluctuation features turn out to be the best (Macro F1=0.61), followed by user activity feature (Macro F1=0.554) and likelihood feature (Macro F1=0.55).

2.6 Browser extension: SCoRe

In the previous section, we have showed that state-of-the-art supervised models are capable of producing a significant accuracy with our proposed feature set to segregate blackmarket customers from genuine users. In order to help users deal with such fake retweeting activities, we attempt to build an extension, named SCoRe for chrome browser. It allows users to spot the blackmarket customers, which in turn provides a better way to understand the importance of a tweet.

Figure 2.5 shows a snapshot of how SCoRe facilitates users. The input is a link of the tweet whose retweeters the user wants to analyze. It first extracts all its retweeters, calculates the features for each retweeter, and then feeds it to the pre-trained SVM. Finally, the label (Genuine or Customer) of each retweeter is displayed. It also allows users to provide feedback on each label. There is a 'thumbs down' symbol associated with each label, upon clicking of which the feedback will be forwarded to the back-end server. SCoRe has the capability to be trained incrementally with the feedback provided by the users. However, special care has been taken to make the learning process robust by ignoring spurious feedback. An attacker may want to pollute SCoRe by injecting wrong feedback. SCoRe handles these spurious feedback by first checking the confidence of the current model on labels associated with the feedback, and ignoring them if the confidence is more than a pre-selected threshold (currently set as 0.75). This makes

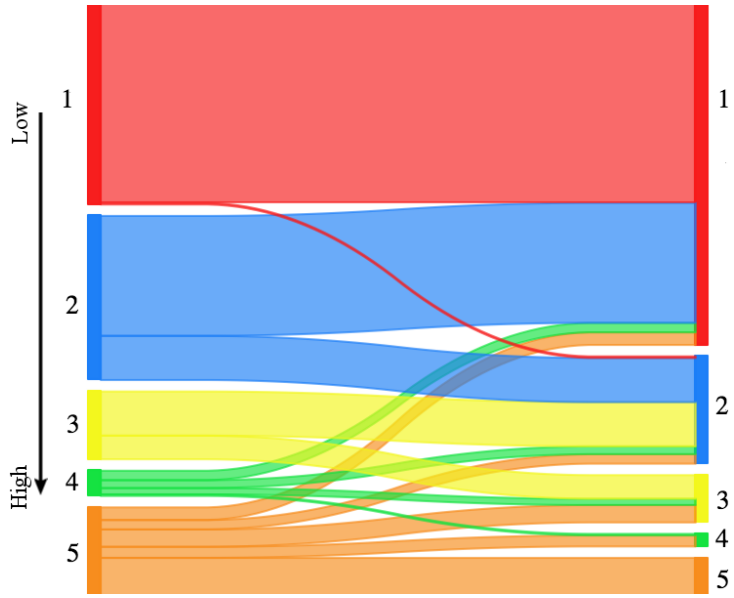


Figure 2.6: Alluvial diagram representing the flows of tweets between the bins (based on retweet count) before and after filtering customers. The colored blocks correspond to different bins (the range of retweet count is divided equally into five bins). Left (right) blocks correspond to bins based on retweet rank before (after) the filtering. The size of the block indicates the number of tweets in that bin, and the shaded waves joining the regions represent flow of tweets between the bins, such that the width of the flow corresponds to the fraction of tweets.

SCoRe more robust under adversarial attacks.

2.7 User study

To analyse the performance of SCoRe, we conducted a user study with the help of 25 volunteers. First, we randomly assigned 20 tweet URLs to each volunteer. Volunteers visited each URL with the extension activated in their browser, clicked on the retweeter list (each retweeter was labeled as 'Genuine' or 'Customer' by SCoRe) and marked the wrong label by pressing 'thumbs down' button. SCoRe took 10 sec to label each retweeter. We recorded the responses of the volunteers and measured the average accuracy. The results shows that SCoRe is highly accurate, achieving an average accuracy) of 85% (with standard deviation of 0.02).

2.8 Implication of collusive retweeter detection

Once we detect collusive retweeters, its immediate implication would be to re-rank the tweets based on modified retweet count (after filtering blackmarket customers). Figure 2.6 shows an alluvial diagram, indicating how tweets (500 tweets collected from the customers' timelines) change their ranking after such filtering. We show that top- and middle-ranked tweets are mostly affected by this filtering. We believe that this analysis opens the scope for a serious re-investigation of the existing metrics for ranking tweets/users.

2.9 Conclusion

In this work, we studied the problem of understanding and detecting blackmarket-based collusive retweeters. The major contributions of this work are fourfold: **Dataset:** We collected a dataset of blackmarket collusive retweeters and annotated them. This, to our knowledge, is the first dataset of such kind. We employed human annotators to label each customer into one of the following categories – *bots*, *promotional* and *normal*. We also collected genuine users who are researchers / experts in machine learning, following the method in [6]. **Characterization:** We propose 64 novel features to characterize customers and differentiate them from normal users. **Classification:** State-of-the-art supervised methods performed significantly well to classify customers and genuine retweeters. We observe that Logistic Regression achieves the maximum accuracy, outperforming the baseline methods significantly. Our method (with SVM) turns out to be equally successful in categorizing retweeters into binary classes – customers and genuine (Macro F1-socre of 0.87). **System design:** We developed `SCoRe`, a chrome extension that spots blackmarket retweeters in real-time. `SCoRe` takes input the link of a tweet, spots its fake retweeters. We also collected users' feedback on the performance of `SCoRe` and obtained 85% accuracy.

3. Exploring different facets of blackmarket services

The growth of online social media has led to a huge increase in the number of users who want to share and publicize various kinds of information. Twitter, the most popular micro-blogging platform, has become a hotbed for users who are involved in different activities such as news publishing, job hunting, recruiting, advertising and publicity. `Retweeting` a tweet is a major action to broadcast a user's message out to millions of users. Retweet action has two major advantages: (i) gaining quick exposure to the content, and (ii) increasing likelihood of gaining new Twitter followers in return. The organic way of gaining a larger number of retweets is a time consuming process, which leads to the creation of unfair methods to gain retweets. Thus, Twitter users often approach various blackmarket services to gain retweets inorganically in a short duration. Blackmarkets spread their collusive ecosystem in such a way that Twitter is unable to detect them even after devoting significant effort to purge the platform off bots, trolls, and fake accounts. One major reason behind the evasion is that the collusive users involved in blackmarket services exhibit a mix of organic and inorganic behavior – they organically retweet some genuine tweets; at the same time, they inorganically retweet tweets submitted to blackmarket services.
This work is the first attempt to provide a thorough study of the collusive users involved in two types of blackmarket services – Premium and Freemium. We collect a novel dataset of collusive users comprising of users from both types of blackmarket services. We provide network-centric, profile-centric, timeline-centric and retweet-centric characteristics of these users and show how users involved in premium blackmarket services exhibit diverse behavior as compared to those involved in freemium services. We further employ human annotators to label collusive users into three types: bots, promotional customers, and normal customers. We then curate 63 novel features to run state-of-the-art classifiers in two settings – binary classification (collusive vs. genuine) and multi-class classification (bot, promotional customers, normal customers, and genuine users). Bagging achieves the best accuracy (macro F1-score of 0.892) in the former setting, whereas Random Forest outperforms others (macro F1-score of 0.791) in the latter setting. We also develop a chrome extension, `SCoRe++` which can detect collusive retweeters in real time.

3.1 Introduction

Online Social Networks (OSNs) attract social entities (users or organizations) connected by social relationships such as followers, friends, etc. to establish a platform for information exchange. Statistics reported that in year 2018, 3.196 billion users were engaged with OSNs, which is a 13% increase over the statistics of 2017[1]. The major reason for the increase in attraction of users towards OSNs is the availability

[1] https://www.smartinsights.com/social-media-marketing/social-media-strategy/new-global-social-media-research/

of online news, job hunting, recruitment, advertisement, networking opportunities, and many more. OSNs not only spread these messages but also help in targeting audiences that might have potential interest in the same. OSNs help in connecting individuals whose likes and dislikes are similar, thereby creating a network around those commonalities. One of the major advantages of Twitter over other social networks in gaining such attraction is the availability of a large number of third-party tools for accessing the service. Moreover, there exist numerous tools for users to grow their presence on Twitter and some additional features such as analyzing brand performance, click-though rate, etc. There exists a bunch of tools to boost online presence for businesses on Twitter[2]. Tools such as Hootsuite[3] provide a common dashboard of multiple accounts managed by a single Twitter user and allow an easy way to retweet a tweet from multiple accounts. Advanced tools such as Tweriod[4] suggest users the best time to publish tweets by analyzing the followers of an account and determining the best time to post a tweet.

The tremendous increase in the involvement of users in OSNs has led to the rise of many fraudulent and spamming activities. In case of Twitter, a tweet's size limit is only 280 characters (previously 140 characters). Thus, Twitter users involved in the promotion of large campaigns/events/websites have to cram their thoughts within the tweet size limit. This becomes even more problematic as most of the tweets related to promotions consist of URLs which usually contain 50-80 characters. Many of these users resort to using various blackmarket services for these promotions. Moreover, most of these promoters try to gain popularity and visibility from specific target audience, which in turn helps them gain more retweets/followers to their tweets/accounts. This leads to users choosing artificial ways (through blackmarket services) to boost their social growth. The benefits of using these services are manifold: (i) making tweets popular by providing retweets, (ii) gaining new followers, and (iii) boosting social credibility. The main attraction of these services is that they provide 'real' retweets/followers/likes, where 'real' means that the users involved in supporting these services are active Twitter users but not bots or inactive users. These fake engagements on Twitter are strictly against the Twitter Terms of Service (TOS), and participation in these services may lead to account suspension[5]. To counter this, the blackmarket services promise to deliver their facilities in such a way that Twitter will not catch their activities. The users engaged in collusive activities are operated by regular human beings expressing a mixed behavior of organic and inorganic retweeting activities – on one hand, they organically retweet as genuine users, and on the other hand, being a part of blackmarket services, they inorganically and randomly retweet other tweets (c.f. Fig. 2.1). Thus, these users cannot be detected accurately by bot detection or fake user detection algorithms. Results in Section 3.8 empirically validate our claim. A bunch of academic research on Twitter studied the problem of bots [57, 85, 86] and spam [60, 87]; but the problem of detecting collusive activities has not been deeply investigated yet. We use the term *collusive users* and *collusive retweeters* interchangeably, both referring to collusive users.

Here, we focus on analyzing two types of collusive retweeters collected from two different types of blackmarket services – Premium and Freemium.
– *Premium services* - These services provide retweets when they receive payment from customers.
– *Freemium services* - These services are free to use, but also provide some subscription plans for higher usage. Here, we restrict our study on freemium services to only credit-based services. In credit-based services, users retweet others to gain retweets on their content, a type of 'give and take' policy.
Contributions. We extend our previous work [3], where our major focus was to detect collusive users involved only in credit-based *freemium* blackmarket services. In this work, our major aim is to provide an in-depth analysis of collusive users involved in *both premium and freemium* services and explore different facets of blackmarket services. In particular, the main contributions of this article are summarized below:

[2]http://blog.digitalinsights.in/twitter-tools-for-business-boost-your-brand/0572587.html
[3]https://hootsuite.com
[4]https://www.tweriod.com/
[5]https://help.twitter.com/en/safety-and-security/fake-twitter-followers-and-interactions

– We create a dataset of collusive retweeters collected from multiple premium and freemium black-market services. We further collect a set of genuine users who are experts in the domain of Machine Learning/Information Retrieval/Data Mining. Each collusive user in the dataset is labeled into one of the following types: *Bots*, *Promotional Customers* and *Normal Customers*.

– We analyze the activities of collusive retweeters (both premium and freemium) based on retweet-centric, network-centric, profile-centric and timeline-centric properties. The major findings are as follows. In the retweet-centric study, we observe that the credit system in freemium services is the primary reason for active participation of the freemium users. In the network-centric study, we observe that the premium services control a limited set of accounts which they use to provide retweets for their customers. On the contrary, freemium services accommodate a large set of registered customers who use the credit system to gain appraisals for the content the customers' submit. In the profile-centric study, we observe that freemium users are more involved in using social media tools such as blackmarket services owned APIs to perform the retweet operation, which in turn indicates that these users are more interested in gaining the credits rather than the content of the tweet. In the timeline-centric study, freemium users are more involved in quoting rather than retweeting tweets as compared to the premium users. The reason is that the freemium users perform the retweet operation using the APIs which embed their own content (hashtags/urls) into the tweets.

– We evaluate the performance of the state-of-the-art machine learning algorithms to distinguish users of blackmarket services from genuine users. We conduct experiments in two settings: multi-class (bots, promotional customers, normal customers, and genuine users) and binary-class (collusive and genuine). In the first setting, we achieve a macro F1-score of 0.791 with Random Forest classifier. In the second setting, Bagging turns out to be the best performer with a macro F1-score of 0.892.

– Finally, we develop a chrome extension `SCoRe++` to detect collusive users in real time. `SCoRe++` is an updated version of `SCoRe` [3] where we incorporate the features of both premium and freemium users. Note that `SCoRe` only had features of freemium users. We apply these features in the backend to train our best performing binary classification model (Bagging in our case).

3.2 Related work

In this section, we introduce relevant prior studies on the detection and characterization of collusive entities in OSNs. Most of the studies deal with detecting fraudulent users, spammer groups, fake followers, etc. to guide a wide range of problems such as information propagation, user behavior, malicious group detection, etc. However, these approaches are not suitable for the collusive retweeter detection as collusive users show a mixture of inorganic and organic behavior (see Fig. 2.1). In our work, we provide a detailed overview of the realm of previous studies from two different facets: (i) malicious activities in OSNs, and (ii) research on blackmarket services. In the latter part of this section, we mention how our method differs from the relevant existing studies.

3.2.1 Malicious activities in online social networks

There has been a considerable amount of studies on the detection of malicious activities in OSNs. Chu et al. [85] conducted a series of experiments to characterize the differences among the behavior of humans, bots, and cyborgs on Twitter. They proposed an automatic classification system using tweeting behavior, tweet content and account properties to distinguish among these types of users. Gianvecchio et al. [86] developed a classification system to detect bots and humans in Yahoo! chat. Davis et al.

[57] proposed Botornot to detect bots in Twitter. It leverages more than a thousand features to check whether a Twitter account is controlled by a human or a bot. Retweets/followers are used to increase the visibility of a tweet/user. Moreover, the usefulness of retweets is not only limited to Twitter. Search engines such as Google ranks websites based on tweets containing website URLs [6]. In other words, if a tweet is published with a URL, and the tweet is retweeted several times, it affects the ranking scheme of search engine. Giatsoglou et al. [1] proposed `NDSync` and observed how retweet threads of fraudulent retweeters are clustered together. `NDSync` also automatically detects fake retweeter groups by assigning a suspiciousness score to each retweeter of the group. Giatsoglou et al. [63] proposed `RT-GEN` to imitate retweet patterns of both fake and genuine users. They reported that fake retweeters retweet concurrently in a lockstep manner.

Multiple studies have focused on the detection of anomalous activities on different social media platforms. Beutel et al. [88] proposed a graph-based approach to detect lockstep behavior on Facebook pages searching for non-bipartite cores in a bipartite graph between users and pages. Leskovec et al. [89] determined the signs of links on multiple OSNs (Epinions, Slashdot, and Wikipedia). Several papers [49, 90, 91] investigated the problem of review spam on e-commerce websites. Gupta et al. [70] studied phone number based spam attack on Twitter.

3.2.2 Research on blackmarket services

There has been relatively less work on investigating OSN based blackmarket services. Stringhini et al. [54] examined multiple Twitter follower markets to understand the growth and dynamics of customers of these markets. Shah et al. [6] analyzed behaviors of link fraudsters. They created multiple honeypot accounts and purchased followers from several follower markets. They proposed different follower-centric features to differentiate between fraudsters and genuine users. De et al. [74] studied the pros and cons of the follower and retweeter based blackmarket services. They also reported that all interactions in Twitter can be easily counterfeited. Cresci et al. [52] identified 49 distinct features to distinguish accounts of human and fake followers. Thomas et al. [75] developed a classifier to detect fraudulent accounts that blackmarket merchants sold on Twitter. They even reported that around 10-20% of all the spam accounts originate from the blackmarket services. Singh et al. [92] studied the behavioral patterns of Twitter follower markets and developed an automated approach to classify these users as spammers. Liu et al. [76] detected crowdturfing following activities on Twitter (voluntary following). They termed these users as 'volowers'. They further proposed `DetectVC` to detect volowers using graph-based features and prior knowledge collected from the follower markets. There have been some recent efforts on collusion in various social media platforms such as live video-streaming [93, 94] and online recruitment [95].

3.2.3 Differences with our previous work

We extend our previous work [3], where our primary goal was to detect collusive users involved in credit-based freemium blackmarket services. We divided the freemium blackmarket services into three categories: social-share services, credit-based services, and auto-time retweet services. We collected and annotated collusive users into three types: bots, promotional customers and normal customers, and ran several supervised methods for classification.

This study differs from the previous work from various angles:

[6] https://blog.twitter.com/marketing/en_us/a/2015/new-measurement-and-buying-tools-through-doubleclick-coming-soon.html

- We provide a thorough study of various kinds of blackmarket services, which to our knowledge is the first attempt of this kind.
- We examine the diverse aspects of collusive retweeting activities on Twitter based on premium and freemium blackmarket services.
- We extend our human-annotated dataset of collusive users by adding the data of premium collusive users.
- We offer an exploratory analysis of the dataset of collusive users based on network-centric, profile-centric, timeline-centric, and retweet-centric observations. This analysis aims to differentiate premium and freemium blackmarket customers.
- We release a new version of our chrome extension, `SCoRe++`, by adding features calculated from both premium and freemium blackmarket services.

To summarize, previous studies provide a profound explanation of the methodologies related to the detection of fake OSN entities. But these studies tend to avoid equally important question of detecting *collusive entities*, which may not be same as fake entities. To overcome the limitations of the previous studies, we provide an in-depth analysis of the collusive users and propose an automated strategy to detect them.

3.3 Dataset collection

In this section, we present our effort to collect a large set of Twitter accounts that actively interact with blackmarkets. The data forms the ground-truth on customers who bought retweets from the blackmarkets (*premium retweeters*) and victims who were compromised by the blackmarket and traded as retweeters (*freemium retweeters*). We also collected a set of genuine users to make the data balanced.

3.4 Exploring different facets of blackmarket services

To analyze the behavior of the blackmarket services, we created one Twitter account (honeypot) per service. All the honeypot accounts were created on the same day with empty timelines (tweets/retweets) and empty social networking features (followers/friends). Hence, we asserted that all the retweeters of these honeypots accounts would be from blackmarket services. We conducted an initial experiment on two new honeypot accounts (one premium and one freemium) to check whether they retweet fresh Twitter accounts or not. It was surprising to see that most of the freemium services do not allow to add accounts/tweets if there is no profile-related information such as user bios, profile image, inactive accounts. Note that while creating our honeypot accounts for further experiments, we made sure that each of the accounts must have a profile picture, cover picture, user bio, user location, user language and other profile-related information. We next created a set of unique tweets which looked attractive (to attract more retweeters), and posted one tweet from each account. We next added each account/tweet to one blackmarket service. Summarily, we created 8 honeypot accounts and added 4 of them to premium services and the remaining 4 to freemium services. As premium services offer retweets in tiers, we selected the package which provides 100 retweets, and the retweeters delivery time is within 48 hours.

3.5 Creating honeypot accounts from blackmarket services

Blackmarket customers. On adding the tweets to blackmarket services (premium and freemium), we implemented several tracking scripts using Twitter API to monitor changes in the tweets. For each tweet submitted to any premium service, we collected retweeters' information at 10 minutes interval. Similarly, for tweets submitted to any freemium service, we visited the earning area of each freemium service at 6 hours interval per day and retweeted tweets of all other customers present in the dashboard to earn maximum credits. We also kept track of the retweets which were not present in the next time frame (deleted retweets). Summarily, we took four premium services, namely SocioCube (SC), RedSocial (RS), SocialShop (SS), and SocialKing (SK), and four freemium services, namely KingdomLikes (KL), YouLikeHits (YH), Traffup (TU), and Like4Like (LL).

Genuine users. We also collected a set of genuine users to compare them with the users of blackmarket services. We started by selecting a set of known users who are guaranteed not to have participated in any form of collusive activities before. To increase the set of genuine users, we added some well-known machine learning and deep learning scientists as suggested in [6]. From the set of genuine users, we discarded those users who had more than 1 million followers since such high follower count may resemble a celebrity-like users. This may further create a bias in our dataset.

Table 3.2 shows the statistics of the dataset. Overall, we collected 475 customers from the premium services (C_p), 196 customers from the freemium services (C_f) and 1000 genuine users. We further extended our set of customers from the freemium services by adding 279 collusive users from our previous dataset [3]. Interestingly, out of the overall 950 collusive users (C_p + C_f), 13 users turned out to have verified Twitter accounts. This clearly shows that Twitter in-house algorithms have been unsuccessful to detect such users. We further extracted profile information and timelines of customers of both types of service and genuine users using Tweepy[7] library. The summary of the entire dataset of this work is presented in Table 3.1.

Table 3.1: Summary of data collected from blackmarket services.

Type	Service	# retweets promised	# retweets delivered	# retweets deleted
Premium	SC	100	320	222
Premium	RS	100	60	16
Premium	SS	100	28	21
Premium	SK	100	67	36
Freemium	KL	–	46	2
Freemium	YH	–	74	10
Freemium	TU	–	13	0
Freemium	LL	–	63	11

Table 3.2: Statistics of the collected dataset. We used an extended version of this data in Chapter 5 where we showed that our model is robust to imbalance by picking different ratios of collusive to genuine users.

Type	# users	# verified users
Premium collusive	475	0
Freemium collusive	475	13
Genuine	1000	28

[7]http://www.tweepy.org/

3.6 Human annotation of collusive users for SCoRe+

So far, we have collected collusive users from multiple premium and freemium blackmarket services. Each collusive user was then categorized by three human annotators[8] into any of three classes (bots, promotional customers and normal customers) based on the instructions given to them. The annotators were also given complete liberty to search for any information on the web for proper annotation of the users. We developed the following set of rules to define each type of collusive user:

C1. **Bots:** Twitter bots are user accounts that can post/retweet tweets or communicate with other users in an automated way without direct human input. The rules that should be followed by a Twitter bot are mentioned in Twitter Automation Rules for Bots released by Twitter[9].

C2. **Promotional customers:** Promotional customers are engaged in campaigns to extend the scope of their content to a relevant audience on Twitter. These users often use keywords such as 'giveaway', 'free', 'win', 'gain' in their tweets in order to gain quick popularity. Twitter also has a set of rules that should be followed by a user for content promotion[10].

C3. **Normal customers:** For all the users who do not fall under the above two categories, we label them as normal customers.

Out of 475 collusive users collected from premium blackmarket services, we found 83 bots, 129 promotional customers and 263 normal customers. In case of freemium blackmarket services, out of 475 collusive users, we found 97 bots, 202 promotional customers and 176 normal customers. The inter-annotator agreement was 0.73 based on Fleiss' κ.

Next, we present numerous insights of blackmarket retweeters based on four properties: retweet-centric, network-centric, timeline-centric and profile-centric.

3.7 Analysis of blackmarket retweeters

We start our analysis on the data collected from four premium and four freemium blackmarket services. We discuss the retweet-centric, network-centric, timeline-centric and profile-centric characteristics of users.

3.7.1 Retweet-centric observations

3.7.1.1 Arrival time of customers

Here, we analyze the arrival time of customers for each of the blackmarket services. Fig. 3.1 shows the variation of the arrival of customers over time for different blackmarket services. Premium users have a steady increase whereas freemium users show stepwise behavior. One possible reason for such behavior in case of freemium services, is that the tweets of customers of these services only get retweeted, when the user has available credits in his/her account. A customer on adding his/her tweet to any of the credit-based

[8] Annotators were experts in social media with age range between 25-35.

[9] https://digitalinspiration.com/twitter-bots-automation-rules-5066

[10] https://help.twitter.com/en/rules-and-policies/twitter-contest-rules

freemium services has to set a credit/score for the tweet. When the tweet is retweeted by other customers, that score is deducted from the credit of the source user and is added to the credit of the retweeted user. When a user has no available credits, the freemium services stop working. The user can again earn credits by retweeting tweets of other customers in the earning area.

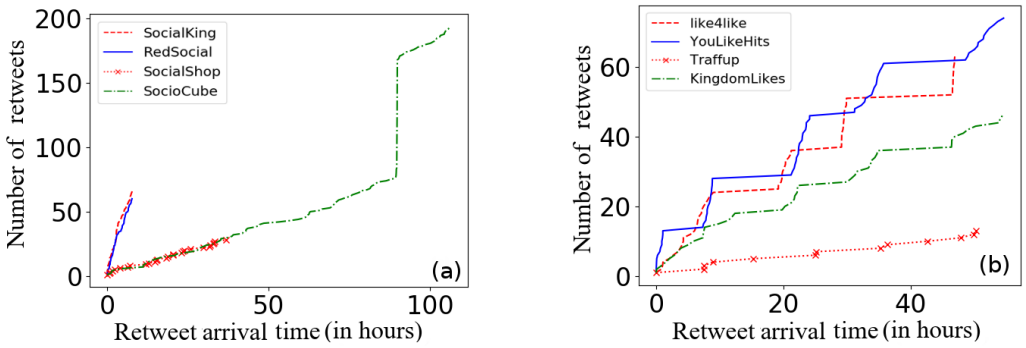


Figure 3.1: Retweet arrival time - (a) premium and (b) freemium.

3.7.1.2 Active time of customers

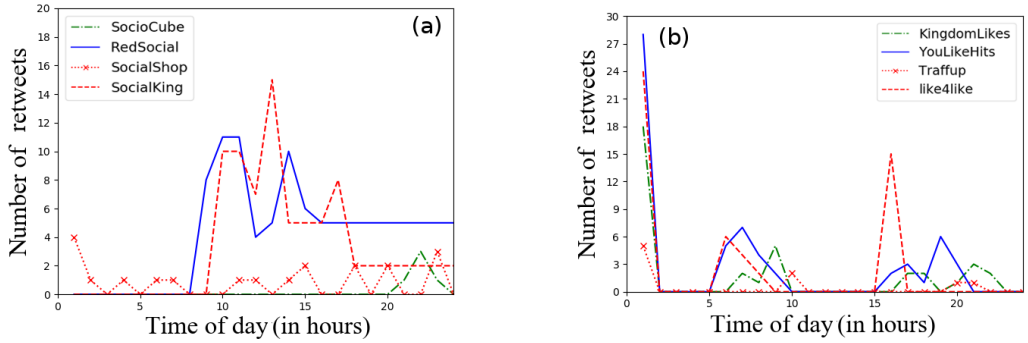


Figure 3.2: Active time - (a) premium and (b) freemium.

The active time is measured as the number of retweets a tweet receives during a particular interval of a day. Fig. 3.2 shows the active time for the customers of the blackmarket services. Premium users are active almost every hour with a peak during office hours (9 a.m.-5 p.m.) whereas freemium users are active during non-office hours. One possible reason of premium services showing such behavior is that premium accounts are mostly controlled by the media marketing companies and not the owner himself [11]. The premium users are target-specific users (involved in brand endorsements, event campaigns, product launches, etc.), who want to entice more people to keep track of the company activities. The media marketing companies help the premium users to create eye-catching tweets to target the right audiences for faster channel growth and lead generation. On the other hand, customers of freemium services control their accounts and have to earn credits in order to get retweeted by other customers. Retweeting tweets of random users to gain credits is a tiresome job which requires immense manual effort and is usually done during free time.

[11] https://www.digitalmarketingagency.com/twitter-management-services/

3.7.1.3 Retweet deletion time of customers

Even though the process of gaining artificial retweets in exchange of money or credits looks as simple as it sounds, customers tend to delete retweets after a certain point of time. Fig. 3.3 shows the number of retweets that we lost for a particular time interval. Premium retweeters tend to delete more retweets than freemium retweeters. The reason for such behavior by premium retweeters may be due to the irrelevancy of those tweets from all other tweets in their timeline. Moreover, deleting these retweets in quick time may help them evade the in-house fake detection algorithms deployed by Twitter. This also may be a trick deployed by the premium services to keep the customer in a vicious circle to buy more retweets from their service.

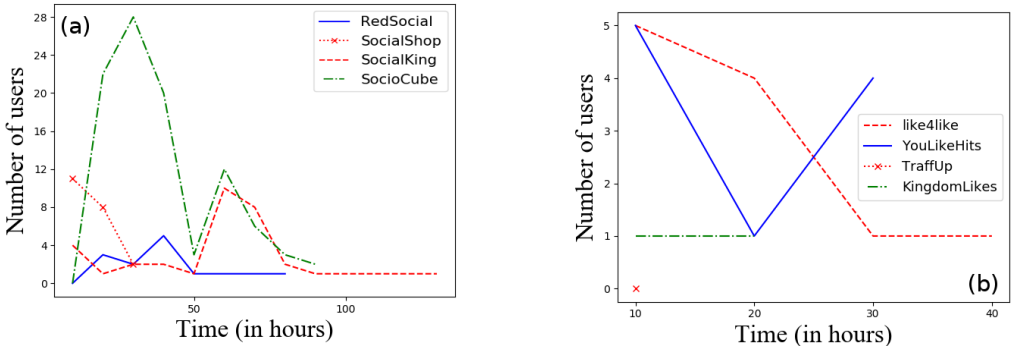


Figure 3.3: Retweet deletion time - (a) premium and (b) freemium.

3.7.1.4 Followers gained upon using blackmarket services

It is very interesting to see that customers who purchased retweets from blackmarket services also gain followers. Note that we created all our honeypot accounts with empty social networking features (followers/followees). Fig. 3.4 shows the relation of the number of followers gained to the number of retweets for both premium and freemium blackmarket services. Though premium services do not help in gaining followers, it is very clear that customers of the freemium services gain followers on retweeting tweets of other customers. However, it is not guaranteed that these followers belong to the blackmarket services or are real followers. We anticipate that some of the followers may be real users because the customers of the freemium services retweet tweets randomly without looking at the content which might create interest among genuine users following the real content to connect with these users.

3.7.2 Network-centric observations

In this section, we analyze the network generated from the data collected from the blackmarket services. We create a directed network for each type of services - nodes of each network are the honeypot accounts and the retweeters of their tweets, and edges represent retweets among these nodes. We use the metrics

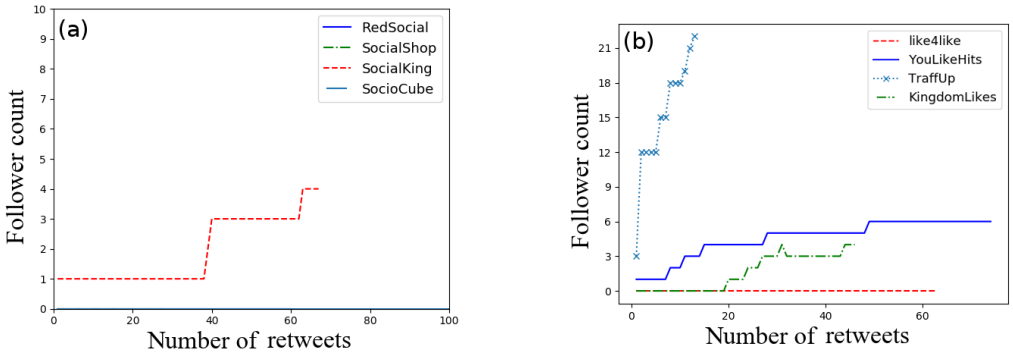


Figure 3.4: Followers gained - (a) premium and (b) freemium.

Property	Premium	Freemium
Density	0.038	0.017
Transitivity	0.338	0.126
Reciprocity	0.348	0.383

Table 3.3: Summary of the network statistics of premium and freemium blackmarket services.

defined in [96], i.e., *density*, *transitivity* and *reciprocity* to describe the network structure.

$$\begin{aligned} density &= \frac{\#\ of\ edges}{\#nodes.(\#nodes-1)} \\ transitivity &= \frac{3\ .\ \#triangles}{\#connected\ triples} \\ reciprocity &= \frac{\#bidirectional\ edges}{\#edges} \end{aligned} \tag{3.1}$$

Here, *density* is the ratio of number of edges in the network to the maximum possible edges. *Transitivity* represents the clustering coefficient of the network. *Reciprocity* measures the bidirectional property of the network – value of 1 indicates that all the edges in the graph are bidirectional. Table 3.3 represents the network property of both types of collusive users involved in blackmarket services. Analysis of the network reveals noteworthy difference between users involved in premium and freemium services. Premium retweeters have high values of *density* and *transitivity* but low value of *reciprocity* as compared to freemium retweeters. Higher value of *density* and *transitivity* for premium service accounts indicates that the same limited set of accounts is used to provide retweets to users that avail the services, as opposed to freemium services where the retweets are provided by a larger number of regular users. Higher value of *reciprocity* for freemium retweeters is expected as customers of these services are involved in a 'give and take' relationship where users retweet others to gain retweets on their own content. Fig. 3.5 shows the visualization of the network formed by our honeypot accounts and their retweeters in case of premium and freemium blackmarket services. It represents how each user in the network is retweeted by other users. We observe that premium users are retweeted by a limited set of users. However, in the case of freemium users, there is no such retweeting pattern among users.

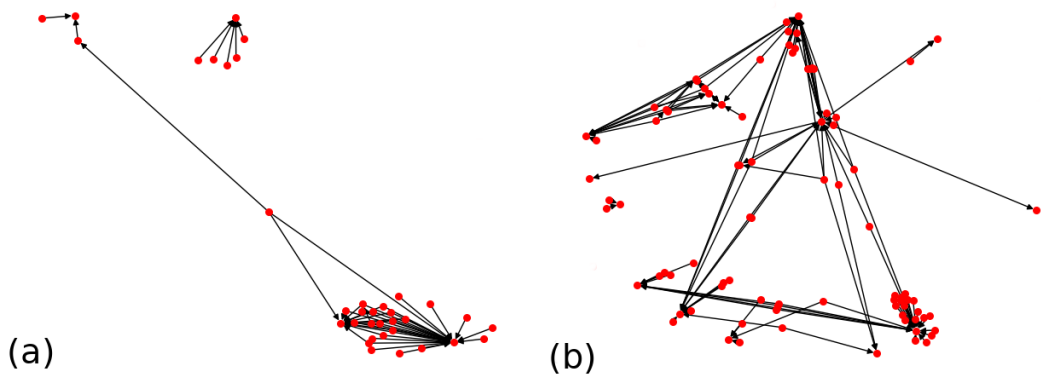


Figure 3.5: Network visualization - (a) premium and (b) freemium.

3.7.3 Profile-centric observations

3.7.3.1 User activeness per day/hour

Blackmarket users are active throughout the week publishing tweets/retweeting other content. Fig. 3.6 shows the activeness of both types of blackmarket users: (a) per day basis and (a) per hour basis. Both premium and freemium users are active equally on every day of a week. However, we see an increasing trend of activeness for these users on normal office hours (11 a.m. - 5 p.m.) as shown in Fig. 3.6.

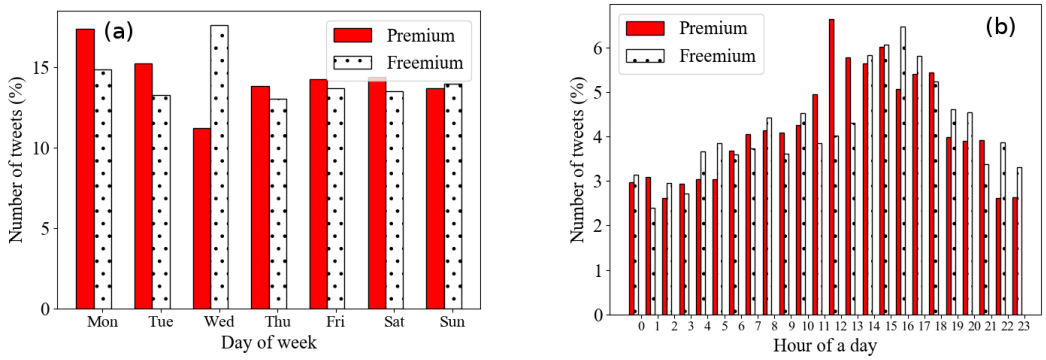


Figure 3.6: Number of tweets (a) per day of a week and (b) per hour of a day.

3.7.3.2 Wordcloud of user description/bios

We also study the 160-character user description field for each of the blackmarket users. Users of the blackmarket services use targeted words to attract more followers/retweeters. Fig. 3.7 shows the wordcloud of the description/bios text for premium and freemium users generated after removing two-letter words and common stopwords. Here, the font size corresponds to the frequency of the text. We find that premium users have words in the description such as 'CEO', 'official', 'speaker', 'Founder'. These words appear to be associated with high profile accounts. However, freemium users have keywords such as 'like', 'agency', 'SocialMedia', 'YouTubeMarketing'. Both premium and freemium users have many keywords related to BitCoin such as 'blockchain', 'crypto', 'btc'. One possible reason of the presence of

such keywords for freemium services could be the time when our dataset was collected; the news of the rise and fall of Bitcoin was primarily driven by OSNs.

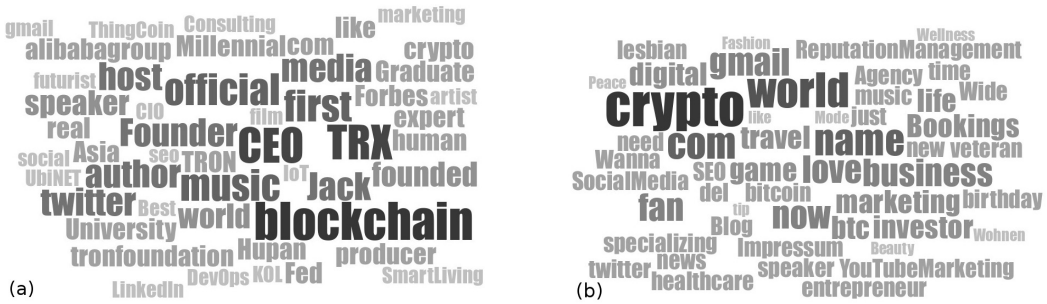


Figure 3.7: Wordcloud of profile description - (a) premium and (b) freemium.

3.7.3.3 Language used by blackmarket retweeters

We study the language of each tweet/retweet in the timeline of blackmarket retweeters. Table 3.4 lists the top 5 languages used by premium and freemium retweeters with the percentage of tweets for each language. We find that more than 50% of tweets posted/retweeted by both types of users are in English. Russian (Spanish) ranks second with 19.17% (10.46%) of all content being posted in the language by Premium (Freemium) users. Surprisingly, we notice around 9% users of both types whose language type is undetected by Twitter API. The primary aim of collusive users is to polarise discussion, influence and mislead users, boost campaigns, etc. where the primary mode of communication is in English.

Premium	**% of tweets**	**Freemium**	**% of tweets**
English (en)	51.41	English (en)	67.21
Russian (ru)	19.17	Spanish (es)	10.46
No language detected (und)	9.05	No language detected (und)	9.54
Spanish (es)	4.40	Arabic(ar)	1.64
Portuguese (pt)	3.48	Portuguese (pt)	1.46

Table 3.4: Top 5 languages used by premium and freemium retweeters.

3.7.3.4 API clients used by blackmarket retweeters

Twitter API can be accessed by an individual using various third-party APIs. As different blackmarkets promise that all their users are genuine users with activities resembling a normal Twitter user, they normally use web or mobile (Android/iPhone) interface to connect to Twitter. Table 3.5 shows the overall application usage for top 10 clients by premium and freemium blackmarket users. Unsurprisingly, we observe that freemium users use Twitter with the API provided by different blackmarket services, social media marketing websites, etc. Around 0.09% of tweets of freemium users are from the blackmarket services owned APIs such as *Easy Retweet API*, *NotFollow.me en Espaol* and *Pay with a Tweet*.

Premium	% of tweets	Freemium	% of tweets
Twitter Web Client	92.44	Twitter Web Client	75.24
Twitter for iPhone	1.62	Twitter for Android	9.19
Google	1.43	Twitter Lite	5.93
Mobile Web (M2)	0.80	Google	3.94
Facebook	0.77	Buffer	1.10
TweetDeck	0.76	Twitter for iPhone	0.94
vk.com pages	0.60	TweetCaster for Android	0.86
GTX1024	0.38	Twitter for Nokia S40	0.78
Twitter Lite	0.31	Facebook	0.61
Instagram	0.20	TweetDeck	0.27
Twitter for Websites	0.19	Mobile Web	0.24

Table 3.5: Top 10 clients used by premium and freemium users.

3.7.3.5 Account dormancy

To measure account dormancy, we calculate the time difference between the creation date of the account and the first activity of the account. Unsurprisingly, 71.6% of the premium retweeters and 55% of the freemium retweeters perform their first activity on the day the account was created. However, 19.5% of premium retweeters and 17.5% of freemium retweeters remain inactive for atleast a month after creating their account.

3.7.3.6 URLs shortening services

Blackmarket users also use a set of URLs to promote their contents. Various URL shortening services such as *bit.ly*, *tinyurl.com* are used by the users to advertise/promote contents. Thomas et al. [23] mentioned that spam URLs are commonly shortened as compared to non-spam URLs. We find that 5.8% of tweets of premium users are from URL shortening services with 2.1% from *bit.ly*. Similarily, we find 5.1% of tweets of freemium users from URL shortening services with a similar percentage of tweets from *bit.ly*. To identify whether these shortened URLs are malicious or not, we run PhishTank[12] for each of the expanded URLs generated from shortened URLs. We observe a very small percentage of tweets (premium-0.0012% and freemium-0.0019%) detected by PhishTank as suspected phises. Although the URL shortening services deploy strong spam filtering algorithms in the backend, the presence of alternate shortening services is used to create multiple redirects to finally lead to the expanded URL.

3.7.4 Timeline-centric observations

3.7.4.1 Embedded mentions, symbols and urls

We detect the number of tweets in the timeline of each collusive user with no mentions, symbols or URLs. In case of premium users, we find 7.7%, 99.7%, 73.2% tweets to have no mentions, symbols and URLs, respectively. However, in case of freemium users, we find 37.8%, 99.2%, 46.9% tweets with no mentions,

[12]PhishTank is an anti-phishing site https://www.phishtank.com/

symbols and URLs, respectively. We observe that both types of users do not use many symbols in the tweets. Nevertheless, freemium users are more involved in sequence of mentioning other users in their tweets. The reason for such behavior by freemium users is that retweeting targeted tweets (tweets with mentions to other users) may help them gain free followers, thereby increasing social popularity.

3.7.4.2 Quoted tweets

We also find the number of quoted tweets for both types of users. We find that only 1.56% of the tweets by premium users and 2.57% of the tweets by freemium users are quoted tweets. The reason for the higher value for freemium users can be because these users use third-party API to access Twitter. Using third-party API to retweet a tweet may add new entities (hashtags, mentions) to the retweet, making it a quoted tweet. Services providing third-party API will be interested to promote themselves by adding hashtags related to their services.

In this section, we provided a detailed analysis of the collusive users based on retweet-centric, network-centric, profile-centric and timeline-centric behavior. In the retweet-centric study, we observe that the credit system in freemium services is the primary reason for active participation of the freemium users. In the network-centric study, we observe that the premium services control a limited set of accounts which they use to provide retweets for their customers. On the contrary, freemium services accommodate a large set of registered customers who use the credit system to gain appraisals for the content the customers' submit. In the profile-centric study, we observe that freemium users are more engaged in using social media tools such as blackmarket services owned APIs to perform the retweet operation, indicating that these users are more interested in gaining the credits rather than the content of the tweet. In the timeline-centric study, freemium users are more involved in quoting rather than retweeting tweets as compared to the premium users. The reason is that the freemium users perform the retweet operation using the APIs which embed their own content (hashtags/urls) into the tweets.

3.8 Can machine learning detect blackmarket users?

All the analysis detailed in this study so far are based on the data collected from the premium and freemium blackmarket services. However, one might be interested to devise an automated technique to categorize different types of users involved in blackmarket agencies. In this section, we describe features used for our experiments and the details of our experimental setup. Further, we analyze the experimental results and discuss the importance of the features. We use 63 features[13] and divide them into two broad categories – profile-centric features and timeline-centric. These features are used to classify users into collusive or genuine [3].

- **Profile-centric features (PF)**
  - *Account age*: Number of days between the day the data was collected and the day the account was created.
  - *Screen name length:* Length of the screen name.
  - *User description existence:* Whether the user has a description or not.
  - *User description length:* Length of user description.
  - *User URL existence:* Whether the user has a URL present in his/her profile or not.
  - *Follower count:* Number of users the account follows.

[13]We had to remove Klout Score feature as the API was no longer working. https://www.cnet.com/news/klout-shutting-down/

– *Friend count:* Number of users the account is followed by.
– *Bot score:* This feature is calculated using the Botometer API[14]. The score is based on how likely the account is to be a bot.

- **Timeline-centric features (TF):**
– *Status count:* Total number of tweets the user has posted.
– *Retweet count:* Total number of tweets the user has retweeted.
– *Average mentions per tweet:* It is calculated as the ratio of total number of mentions to the total number of tweets the user has posted/retweeted.
– *Average URLs per tweet:* It is calculated as the ratio of total number of URLs to the total number of tweets the user has posted/retweeted.
– *Average hashtags per tweet:* It is calculated as the ratio of total number of hashtags to the total number of tweets the user has posted/retweeted.
– *Average tweets per day:* It is the average number of tweets the user publishes in a day.
– *Average retweets per tweet:* It is the average number of retweets the user retweets in a day.
– *Average symbols per tweet:* It is calculated as the ratio of total number of symbols to the total number of tweets the user has published/retweeted.
– *Tweeting likelihood ($TL_{1\text{-}7}$)*: It is calculated as the ratio of the per day tweet count of a user to the total number of tweets the user posted in a week. We calculate 7 different features for seven days of a week.
– *Retweeting likelihood ($RL_{8\text{-}14}$)*: It is calculated as the ratio of the per day retweets of a user to the total number of retweets the user posted in a week. We calculate 7 different features for seven days of a week.
– *Tweet regularity ($TR_{1\text{-}7}$)*: It is the fraction of tweets posted by the user at i^{th} hour of that day calculated as $-\sum_{i=1}^{24} p(x_i) \log p(x_i)$, where $p(x_i)$ is the fraction of tweets posted by the user at i^{th} hour of that day. We calculate 7 different features for seven days of a week.
– *Reweet regularity ($RR_{1\text{-}7}$)*: It is the fraction of retweets posted by the user at i^{th} hour of that day calculated as $-\sum_{i=1}^{24} p(x_i) \log p(x_i)$, where $p(x_i)$ is the fraction of retweets posted by the user at i^{th} hour of that day. We calculate 7 different features for seven days of a week.
– *Tweet steadiness*: Tweet steadiness is defined as $1/\sigma_t$ where σ_t is the standard deviation of time difference between consecutive user-generated tweets.
– *Retweet steadiness*: It is the reciprocal of the standard deviation of time difference between consecutive user-generated retweets.
– *Maximum tweet likelihood ($MTL_{1\text{-}7}$)* : It is the ratio of per-day number of tweets posted by a user to the maximum number of tweets the user posted in a day of a week.
– *Maximum retweet likelihood ($MRL_{1\text{-}7}$)*: It is the ratio of per-day number of retweets done by a user to the maximum number of retweets the user does in a day of a week.
– *Retweet count deviation*: It is the standard deviation of number of retweets for all user-generated tweets.
– *Retweet time deviation average*: It is the mean of log-time difference between consecutive retweets.
– *Retweet time deviation standard deviation*: It is the standard deviation of log-time difference between consecutive retweets.

We use these features for our classification experiment. We first conduct a multi-class classification experiment with the following four classes - bots, promotional customers, normal customers and genuine users. We then perform a binary classification experiment with two classes - collusive (combining all types of customers) and genuine users.

[14]https://botometer.iuni.iu.edu/

Table 3.6: Performance of different supervised classifiers for multi-class classification.

Classifier	Micro				Macro			
	Precision	**Recall**	**F1**	**ROC-AUC**	**Precision**	**Recall**	**F1**	**ROC-AUC**
Decision Tree	0.715	0.715	0.715	0.811	0.712	0.719	0.714	0.829
K-NN	0.627	0.627	0.627	0.739	0.584	0.567	0.569	0.669
Logistic Regression	0.662	0.662	0.662	0.749	0.622	0.585	0.584	0.671
Naive Bayes	0.534	0.534	0.534	0.631	0.529	0.495	0.494	0.592
SVM	0.323	0.323	0.323	0.441	0.329	0.256	0.133	0.391
Random Forest	**0.785**	**0.785**	**0.785**	**0.844**	**0.792**	**0.787**	**0.791**	**0.849**
Bagging	0.782	0.782	0.782	0.830	0.781	0.788	0.784	0.831
Boosting	0.323	0.323	0.323	0.441	0.329	0.256	0.133	0.391
MLP	0.708	0.708	0.708	**0.923**	0.737	0.718	0.712	**0.921**
NN	0.348	0.348	0.348	0.421	0.521	0.348	0.202	0.441

Table 3.7: Performance of different supervised classifiers for binary classification.

Classifier	Micro				Macro			
	Precision	**Recall**	**F1**	**ROC-AUC**	**Precision**	**Recall**	**F1**	**ROC-AUC**
Decision Tree	0.825	0.825	0.825	0.846	0.845	0.846	0.846	0.846
K-NN	0.811	0.811	0.811	0.813	0.808	0.813	0.809	0.813
Logistic Regression	0.825	0.825	0.825	0.819	0.824	0.819	0.820	0.819
Naive Bayes	0.788	0.788	0.788	0.788	0.787	0.788	0.785	0.788
SVM	0.568	0.568	0.568	0.500	0.284	0.500	0.362	0.500
Random Forest	0.890	0.890	0.890	0.884	0.893	0.884	0.886	0.884
Bagging	**0.895**	**0.895**	**0.895**	**0.889**	**0.896**	**0.889**	**0.892**	**0.889**
Boosting	0.568	0.568	0.568	0.500	0.284	0.500	0.362	0.500
MLP	0.840	0.840	0.840	0.827	0.824	0.828	0.823	0.827
NN	0.676	0.676	0.676	0.504	0.637	0.639	0.638	0.504

We use several standard supervised classifiers – Support Vector Machine (SVM), Decision Tree (DT), Naive Bayes (NB), K-Nearest Neighbors (K-NN) and Logistic Regression (LR), and three ensemble classifiers – Random Forest (RF), Bagging (BG), Boosting (BO), Multi-layer Perceptron (MLP) and a customized neural network (NN). Hyper-parameter optimization is also performed to generate the best results using multiple parameters e.g., we use CART with Gini gain criteria for Decision Tree, multinomial logistic regression and SVM with linear kernel. We initialize MLP with 1 hidden layer of 100 dimensions using ReLU activation function with Adam adaptive update rule. For NN, in case of binary classification, we pass the extracted features through dense layers of dimensionality 4. We use ReLU as a activation function independently in each of the 4 nodes of first 2 dense layers. Finally, we set the final activation function to be sigmoid and minimize the binary cross entropy loss for 500 epochs using Adam adaptive update rule. In case of multi-class classification, we use softmax activation function in the output layer. The performance evaluation of each of the methods is measured using the following evaluation metrics: Precision, Recall, F1-score, and Area under the ROC curve. The evaluation metrics are reported after averaging the result of 10-fold cross-validation.

Table 3.6 shows the result of the multi-class classification. With our feature set, Random Forest classifier turns out to be the best model with a macro F1-score 0.791. Using the MLP classifier, we achieve a very high ROC-AUC score of 0.923 in micro (0.921 in macro) setting. Further investigation reveals that the prediction results too many negative samples (most of the users are classified as genuine users) which is responsible for the high value of ROC-AUC score. Table 3.7 shows the result of the binary classification. With our feature set, Bagging outperforms the other classifiers in detecting collusive users in terms of all the evaluation metrics (macro F1-score 0.892).

Next, we discuss the working principle of the chrome extension to detect collusive users in real time. The chrome extension will notify Twitter users whether a retweeter of a tweet is 'collusive' or 'genuine'.

3.9 SCoRe++: Detecting collusive users in real time

In order to detect collusive users in real time, we develop a chrome extension SCoRe++ for end-users. The extension classifies a Twitter user into collusive or genuine and seamlessly integrates the result into user's Twitter pages. On installing the chrome extension, it automatically loads when an end-user opens a tweet page. SCoRe++ extracts the IDs of each retweeter of the tweet and makes a request to our backend server. The backend code then extracts the features using the Tweepy API and feeds the features into the best trained model (Bagging in our case) to return appropriate label for each retweeter. The returned label of each retweeter is showed in the tweet page. We also add a 'Report mislabeled' label for feedback from the end-user. Upon clicking on this button by the end-user, we store the retweeter ID and the label returned by SCoRe++ in our server. We retrain our classifier with the feedback from the end-user and check the confidence score of the new model. The current version of SCoRe++ is written in Javascript[15] and the backend code is written in Python (using Flask Framework[16]). The time taken to label each retweeter is dependent on various factors (internet bandwidth, Twitter API query time etc.). Fig. 3.8 shows the working of SCoRe++ on Chrome Browser.

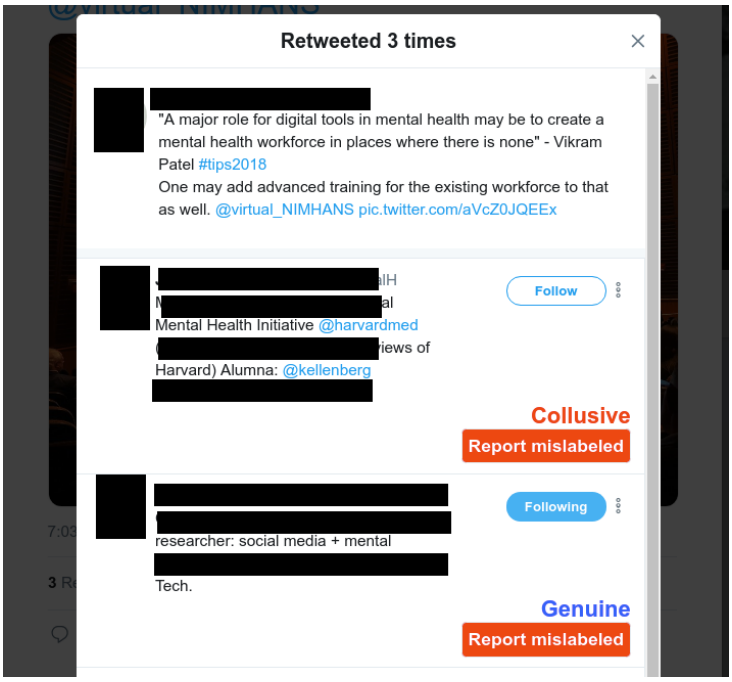


Figure 3.8: Working of our chrome extension SCoRe++.

3.10 Conclusion

In this study, we thoroughly investigated Twitter blackmarket services which provide retweets for a tweet. We conducted our study on two different types of blackmarket services: premium and freemium. We

[15]https://en.wikipedia.org/wiki/JavaScript
[16]http://flask.pocoo.org/

first followed an active-probing strategy to create a dataset of collusive users collected from both the blackmarket services. In case of premium services, we purchased a fixed set of retweeters by paying a certain amount to the service. To collect the users involved in freemium services, we chose only credit-based freemium services where we retweeted tweets of other users of the service to gain credits. We then provided a detailed analysis of these collusive users based on retweet-centric, profile-centric, timeline-centric and network-centric behavior. Following this, we ran several supervised classifiers to classify collusive users and genuine users based on a set of 63 features. We further developed a chrome extension `SCoRe++` to detect collusive users in real time. We would like to reemphasize that this work is the first attempt to present a rigorous analysis of various kinds of blackmarket services which, we believe, would help researchers understand the micro-level dynamics of these syndicates and lead to develop computational techniques to expose their activities.

In our continuing research, we are interested to explore the following avenues. First, we would like to detect suspicious tweets which are submitted to blackmarket services. Automatic detection of such tweets will surely trigger and fuel further research efforts for the detection of collusive users. Second, we will try to capture the interdependency of collusive users and the tweets which are submitted to the blackmarket services. Third, we also intend to expand our human-annotated dataset of collusive users (both premium and freemium users). Ultimately, our goal is to design a scalable real-world collusive user detection framework for Twitter.

4. Detecting fake retweeters using Hawkes process and topic modeling

Retweets are essential to boost the popularity of a tweet, and a large number of fake retweeters can contribute heavily to this aspect. We define a fake retweeter as a Twitter account that retweets spammy tweets, retweets an abnormally large amount of tweets in a short period, or misuses a trending hashtag to promote events irrelevant to the topic of discussion. We introduce an up-to-date, temporally diverse, trend-oriented labeled dataset to address the problem of fake retweeter detection. We develop a novel classifier, called HawkesEye which makes predictions based on a temporal window, in contrast to existing approaches which require a *graph-like* relationship between tweet entities, or the presence of the *entire retweeting timeline* of a retweeter. HawkesEye utilizes both temporal and textual information using a class-specific topic model and Hawkes processes. Experiments on our curated dataset show significant improvement over four state-of-the-art methods, with precision and recall scores of 0.964 and 0.960 on a balanced dataset, respectively – HawkesEye beats the best baseline by 6.16% and 25.98% relative improvement in terms of precision and recall, respectively. We also diagnose our model to understand the advantages and pitfalls of the underlying mechanism. We believe that the extent of this study is not restricted to Twitter, but generalizable to other social media systems such as Facebook and Instagram with similar reposting capabilities.

4.1 Introduction

In the Twitter ecosystem, the number of retweets of a tweet directly affects its exposure and visibility to other Twitter users, and hence acts as a key factor for information diffusion. Consequently, the number of retweets a user gets for his tweets directly affects the popularity of the user. Hence, with the motivation to widen outreach within the community, a user may try to artificially increase his popularity using blackmarket services which inorganically retweet his tweets in return for a fee. A funded user may employ a large number of people to manually create fake accounts and carry out activities on the social-media contents of other customers. According to our previous work [3], such blackmarket services may be *premium* or *freemium*; the latter can further be categorized into *social-share services* (peer-aided sharing activities), *credit-based services* (i.e., they offer credits in exchange for a certain number of retweets overall) or *auto-time retweet services* (where customers can request for a fixed number of retweets for each of their tweets within a time window). There are many such blackmarket services, such as *traffup.net*, *tweetfull.com*, or *justretweet.com*, which offer these services while being non-compliant with Twitter's terms and policies. Therefore, there is a need to identify such fraudulent activities and flag them with a definable level of confidence.

In our study, we notice that the difference of the retweeting patterns between fake and genuine

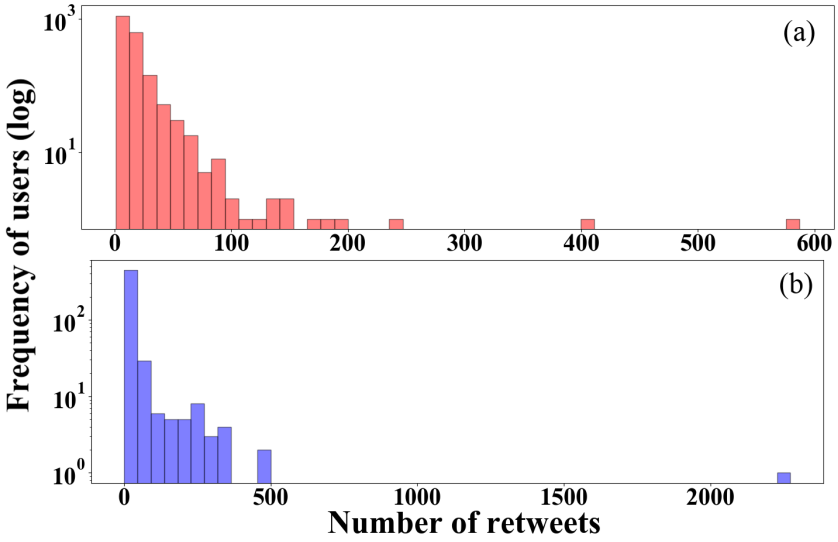


Figure 4.1: Histograms of (a) genuine and (b) fake retweeting activities present in our dataset.

retweeters is diminutive, making any fraud detection model extremely difficult to classify them. This corroborates to the fact that fake retweeters are more often humans rather than bots, and are part of a blackmarket agency that lends such collusive services [4]. However, there are some definitive observations that we can note from our dataset, as mentioned below:

Temporal characteristics: Fake retweeters tend to have a much higher retweeting frequency in a window of observation. We observe in Figure 4.1 that the distribution of users frequently retweeting is massive for the fake users in the 100-500 range. In contrast, for the genuine users, most are evenly distributed in the 1-100 range with a few outliers. Since fake retweeters ultimately have to lend their services, the task of retweeting gets assigned to several people, who collectively tend to retweet at a higher rate. However, on the flip side, there could be instances where genuine retweeters who devote their entire time for retweeting (for financial gains or advertisement purposes) appear to be exhibiting fake behavior, such as continuously clicking the retweet button on their tweets, posting new tweets with the same text, or retweeting at irregular hours. These constitute the outliers mentioned before for genuine users and can be observed in Figure 4.1 as those with the number of retweets in the range 100-600. Several news agencies have been observed to manifest this behavior since they easily hire their employees to retweet multiple times [97] manually.

Contextual characteristics: Fake retweeters tend to use words that have less correlation or relevance to the general topic of discussion. Figure 4.2 shows the marked difference between the topics which are paid the most attention by both classes of users. We observe that texts present in retweets of genuine users have words related to trends (e.g., 'delete facebook', 'trending', 'SpaceX', etc.) and descriptive personal information (e.g., 'elon musk', 'mark zuckerberg', etc.); however, texts present in retweets of fake users have words related to Twitter engagements (e.g., follow, likes, delete, etc.). The primary reason behind words related to Twitter engagements for fake retweeters is the use of third-party API to perform retweets, which add new entities (hashtags, mentions) to the retweet relevant to the API/service. Moreover, genuine users focus on topics close to the trending hashtag, whereas fake users focus on getting fake engagements, along with disseminating spam. Thus, as the hashtag related to a topic starts trending, fake retweeters can leverage its virality to retweet completely unrelated posts, essentially producing spam for other users. For example, a fake retweeter can promote or advertise a beauty product using a hashtag such as *#maga*, which has millions of users posting simultaneously about the American elections. In such cases, the original tweet writer is often responsible for producing the spammy text. (In rare cases, a fake retweeter

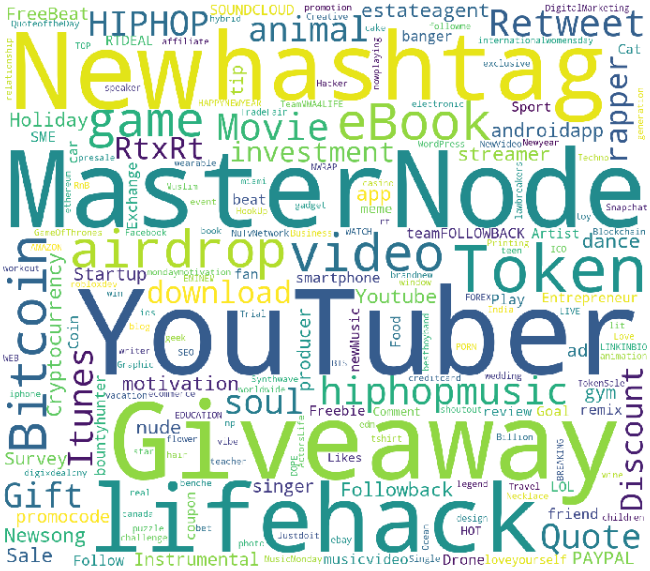


Figure 4.2: Wordclouds obtained from retweets of (a) genuine and (b) fake Twitter users. Font size corresponds to relative frequency in the text. For clarity, we remove common stopwords.

may add the irrelevant content at the beginning or the end of a retweet while keeping the authentic content intact.) It is easier to identify these fake retweeters, as the entire retweeting thread will simply consist of the same content, and the users are all part of the same network, and hence can be flagged with a much greater level of conviction.

We can thus leverage the temporal and contextual characteristics for classification to detect such fake retweeters. However, the very nature of the problem leads to some interesting challenges, which we tackle here to specify the classification task. We thus note the persisting challenges as observed from the data as follows:

- **A clear bifurcation between *Fake* and *Genuine*:** The very notion of *fake* retweeters is confusing, as most retweeters (whether fake or genuine) are humans rather than bots, and have similar retweeting patterns. Therefore, it would be intractable to classify a retweeter unless a robust definition of *fake* is followed. Thus, we define a fake retweeter as a Twitter account that:
  - retweets spammy tweets,
  - retweets an abnormally large number of tweets in a relatively small period, or
  - misuses a hashtag of a trending topic to promote content that is unrelated to the topic.

  If any of the above general characteristics are observably exhibited by the account, singularly or in combination, the user is marked as fake. Note that we do not include bots in our definition since the premise is that, at the very least, the accounts have not been created artificially. The purpose of the study is to investigate the inorganic behavior of actual Twitter users.
- **Twitter API restrictions:** Due to recent API restrictions, it is difficult for a third party to collect an extensive amount of data from Twitter, such as the entire tweeting history of a user, or the complete connected graph of users interacting with a particular tweet. We, therefore, mitigate this issue by collecting tweets within a *window of observation* and using only these tweets in our model. The tweets are collected based on hashtag search of a trending hashtag within the window of observation, which is enabled by the Twitter REST API[1]. The advantages here are twofold. Firstly, we can still

[1]https://www.programmableweb.com/api/twitter

work with the API restrictions in place. Secondly, our model performs efficiently with the curated data, instead of requiring whole retweet timelines of users, or mining of underlying graph structures in the crawled data.

- **Skewness of data:** There is an undeniable skew in the data collected, that is highly biased towards genuine retweeters. This is because a lot of genuine retweeters participate actively and vocally towards a trending topic, and retweeters as per our definition of 'fake' (hereafter referred to as 'fake retweeters') are a minority. We thus need to account for a balanced representation of both classes in our classification model. We solve this issue by using a class-specific topic model, which selectively generates topic vectors for the fake and genuine classes separately.

Our contributions

There are three major contributions of this work:

- **Novel Dataset:** We collected a latest Twitter dataset crawled using hashtags such as *#deleteface-book*, *#cambridgeanalytica*, *#facebookdataleaks*, which consists of over $30k$ tweet objects and around $2,508$ retweeters in all. We employed human annotators to manually annotate each retweeter as 'fake' or 'genuine'. This dataset, to our knowledge, is the first dataset that focuses on a narrow and trending topic and provides labeled data of retweeters based on their retweeting patterns in a very short duration. In our case, a short duration refers to the time period during which the trend or the hashtag starts trending, hits the peak, and then diminishes in popularity ending up in the same degree of base popularity where it started. It is a common observation that trends (hashtags) in Twitter tend to rise exponentially in terms of their usage and then drop steeply to the level at which it started. This is our area of concentration, since fake retweeters tend to use the hashtag for spammy content only when it has a significant virality in the Twitter community, so that it shows up in people's feeds following that hashtag, and/or catches the attention of someone just casually scrolling.
- **Novel Method:** We propose HawkesEye, a novel retweeter classification model that takes temporal information into account by utilizing Hawkes processes [98], and utilizes Latent Dirichlet Allocation (LDA) based topic modeling to represent the textual content in the tweets.
  Hawkes processes are a self-exciting temporal point process ideal for modeling tweeting activity in Twitter [99]. The principle of self-exciting processes is that all previous occurrences of an event will influence the future dynamics of the process. The basic model is ideal for application on predicting retweet evolution since it has been observed that a retweet of a tweet will influence the rate at which more tweets will arrive in the future[2]. LDA [100] is a popular topic summarization tool used in natural language processing, which is used to summarize a retweeter's retweet timeline. Our model does not use any explicit feature engineering methods and is also efficient in terms of computational complexity.
  Methods found in the literature have so far focused on graph-theoretic approaches for identifying fake retweeters or retweeter clusters [63] and neglected temporal information, which is an important component of analysis in the spread of tweets and retweets. To the best of our knowledge, this is the **first approach to classify retweeters in a limited data setting, with only a window of observation taken into account**.
- Exhaustive experiments are conducted on the collected dataset to show the performance of HawkesEye. HawkesEye turns out to be highly efficient, outperforming four baselines by a significant margin. HawkesEye achieves 0.96 F1-Score, which is 18.9% (relative) higher than the best baseline. We also present a feature ablation study, and a side-by-side diagnostic of HawkesEye to understand

[2] https://www.entrepreneur.com/article/285999

the utility of individual components of the model.

4.2 Related work

Most of the prior studies deal with the detection of spammers [101, 102], bots [57, 103] and fraudulent accounts [23, 69] on Twitter. Here, we briefly elaborate previous literature in two directions – fake retweeter detection and point process used for different applications in online social media.

4.2.1 Fake retweeter detection

Fake retweeters are those Twitter users who inorganically retweet others' tweets to increase the visibility of the tweets and authors of the tweets. The reason may be their involvement with paid blackmarket services to earn money/credit, or self/organized collusion to promote/demote certain events. They also act as genuine users when they perform organic activities. Giatsoglou [104] proposed a set of features extracted from retweet thread for spotting synchronized retweeting behavior based on textual and tweet properties. The authors developed NDSync, a generalized method that detects retweet fraud based on the suspiciousness score. However, their model is unable to detect fake retweeters when their *entire retweet threads* are not available (i.e., in the case when we are simply presented with a snapshot of tweets in their timeline over a very specific period), or when the users under consideration are not inter-connected via direct follower-followee relationship (a sparse graph, more representative of the true Twitter ecosystem). NDSync also depends heavily on the intuition that organic retweeting behavior has more variability, whereas fraudulent behavior is more synchronized. In truth, fake retweeters such as blackmarket users are more subtle and resort to means such as posting through different accounts at different times, suggesting that they are not entirely synchronized or periodic in nature [3].

Another framework to identify malicious retweeter groups based on temporal and content-based features was developed by Nguyen [105]. They proposed Attractor+, which measures classifies retweeter groups as malicious or genuine based on the pair-wise similarity of users on their retweeting behavior. However, their algorithm lays too much emphasis on the difference in group-based features, assuming that all retweeters in a group have synchronized behavior. Therefore, this method is not applicable for individual-level fake retweeter detection. In one recent work, Gupta [106] superseded the above, proposing MalReG to extract and prune the retweeter groups.

However, most of these latter methods depend on efficient feature extraction techniques, which may be computationally time-consuming and may require domain expertise. Moreover, a more generalized methodology needs to be adopted for temporal profile classification. Yuan et al. [107] proposed Ianus, a Sybil detection method to detect fake accounts on WeChat by leveraging only account registration information. BalaAnand et al. [108] proposed EGSLA, a graph-based semi-supervised learning algorithm to detect fake users in Twitter. Van and Eloff [109] tackled an interesting problem of how fake users created by bots are different from the fake users created by humans. Swe and Myo [110] detected fake accounts in Twitter using a blacklist creation approach using LDA and TF-IDF methods.

4.2.2 Point processes in online social media

There is a growing use of, and related literature on the application of Hawkes processes in Twitter and other online social media in general [111], [112]. In particular, Twitter dynamics and information cascades have

been studied and modeled many times using Hawkes processes. Zhao [99] came up with an application of a self-exciting point process model, called SEISMIC, to predict the current estimate on the number of reshares of a post (i.e., its popularity) in real-time, given its sharing history till a point. Gao [113] proposed a similar work for modeling the retweeting dynamics using an extended reinforced Poisson process model. Kobayashi [114] applied a time-dependent Hawkes process model, called TiDeH, to predict the overall dynamic curve of the time evolution of the frequency at which retweets appear after a window of observation.

Recently, Chen [115] proposed a Marked Self-Exciting Process with a time-dependent excitation Function, called MaSEPTiDE for predicting retweeting dynamics and tweet popularity. COEVOLVE [116] is yet another recent approach based on the point process to jointly model information diffusion and network evolution mechanisms, keeping in mind their highly intertwined nature. In popular social networking sites such as Twitter, users continuously get exposed to new information sources and share it with their peers accordingly, creating links for information diffusion, thus making COEVOLVE an apt modeling mechanism to mimic social network dynamics.

Rizoiu [117] proposed a mechanism to predict the popularity of online content using Hawkes processes and quantified the relationship between the external promotions a product (say, a video) receives via various social networks and its consequent popularity. The final goal was to have a prediction mechanism of how popular/unpopular a product might turn out to be, given these extrinsic promotions. Lukasik [118] used a multivariate Hawkes process model for the task of rumor stance classification on sequences of tweets. Prediction for the most likely label for each test tweet was performed by maximizing a likelihood function of the occurrence of the tweet, which considers both the textual information and the temporal dynamics. This is the only known work before having used Hawkes processes for retweet classification, and our work bears some similarities to them. However, our work has several advantages such as its applicability for retweeters and not for individual tweets, and inclusion of LDA for the summarization of retweeter timelines.

4.2.3 How HawkesEye is different from others?

Our method primarily focuses on an API restricted scenario, wherein we only have a limited number of retweets given a particular trending topic and have no access to the entire retweet timeline of a user. Due to recent API restrictions set by Twitter, we can only fix a period of observation and collect retweets related to the hashtag within that period, which is why we propose a solution that maximizes utilization of the limited quantity of data, using both the temporal and textual information that is provided in the Twitter JSON objects.

To the best of our knowledge, the Hawkes process has never been used to detect fake retweeters in the past. Our model is unique in the following sense: (i) unlike traditional supervised models, it does not require any manual feature extraction, and (ii) it takes account-centric information of a user to make the prediction. The scenario is more realistic since, in order to design a scalable real-world system, we may only have access to the recent behavior of a user, and the underlying user-user interaction is difficult to collect in real-time. It is also useful to handle the cold-start problem where you have limited information about a new user. Table 9.1 summarizes a comparison of `HawkesEye` with other relevant approaches.

Table 4.1: Comparison of HawkesEye and other relevant methods w.r.t different dimensions of an algorithm.

	Davis et al.[57]	Dutta et al.[3]	Yuan et al.[107]	Balaanand et al.[108]	Walt and Eloff.[109]	Swe et al.[110]	Our (HawkesEye)
Address collusion phenomenon		✓					✓
Use engineered features	✓	✓			✓		X
Consider temporal information							✓
Consider graphical information			✓	✓			X
Consider topic information						✓	✓
Handle cold-start problem							✓
Scalable real-world system	✓	✓		✓			✓

4.3 Dataset

To take into consideration a realistic data and time-bound scenario wherein it is inconvenient to mine the entire timeline of a user, we limited all user timelines to a given observation period (slightly more than a week in our case). This period of around a week has been set, keeping in mind the drop in popularity in the use of the hashtag by performing exploratory analysis on the dataset. All retweets of the timeline lying outside the observation period were discarded. This dataset was particularly collected for this study due to the non-existence of a retweet-rich, timestamp denoted, content-filtered, and authentic dataset by any previous work on similar topics. Hence, we used our own curated dataset to resolve the unavailability of other suitable annotated datasets matching our requirements.

As mentioned earlier, we also restricted our domain of study to a particular trending topic. When a topic is trending, new tweets (containing hashtags related to the topic) are added to a list of tweets about the trend, bolstering the trend's popularity. Using trending topics helps Twitter users reach out to their target audience by sharing the tweet with whoever searches for the trend, instead of just his/her followers. A trending topic in Twitter may be characterized by the usage of several related hashtags, all of which are used together or separately across various tweets. We, therefore, took advantage of the 'Delete Facebook' campaign that was deemed viral in the latter half of March 2018, which was mainly characterized by the related hashtags: **#DeleteFacebook**, **#CambridgeAnalytics**, and **#FacebookDataLeaks**. The campaign has been a prominent and greatly debated topic around the world with a large number of potential fake users having malicious intent, harboring a biased agenda against Facebook, or indiscriminately retweeting spam and marketable content while misusing these hashtags.

4.3.1 Data collection and preprocessing

For the period of March 20-29, 2018, we collected 12 million raw tweet objects on the above hashtags using Twitter's Streaming API, out of which users with lesser than 5 retweets were filtered out, to threshold the timelines to a minimum limit of 5 timestamps. From the remaining data, a randomly selected subset of more than $43,000$ tweets and retweets ($3,000$ users) was considered for annotation. Out of these users, those who posted only the original tweets during the period were removed (as we are only interested in retweeters), resulting in $2,508$ users.

Genuine retweeters	$2,001$
Fake retweeters	507
Number of unique retweets	$26,883$
Total number of tweets and retweets	$43,650$
Mean retweet count per user	20.553
Median retweet count per user	12.0

Table 4.2: Statistics of the annotated dataset.

The textual and temporal components of the tweet objects were subjected to different preprocessing techniques, which were carried out independently. The text was preprocessed using stemming, stop word removal, hashtag removal, as well as URL removal. All tags in the text content were also removed to leave behind only the most discriminative words. Timestamps were also converted to UNIX timestamp format, and min/max normalization was further done for each retweeter timeline separately to shrink the timestamp range to $[0, 1]$. This preprocessing does not bias the modeling in any manner, as it assumed that each retweeter timeline corresponds to a separate Hawkes Process, as explained later in Section 4.6.

4.3.2 Dataset annotation

We hired three human annotators[3] to manually annotate each of the $2,508$ retweeters as fake or genuine, based on the retweet text, user profile information, and timestamp. These retweeters were labeled on the basis of observable retweeting activity and spam content relative to the topic. We followed a labeling procedure similar to the one described by [104]. Hence, a retweeter was labeled as 'fake' if any of the conditions mentioned below satisfies:

- The text in a majority of his/her retweets contains spammy links and common spam keywords,
- There are multiple retweets of the same original tweeter with promotional or irrelevant content,
- The profile information is fabricated or mentioned promotional activity,
- A large number of tweets or retweets were done within a short time period (a few seconds).

Annotators were also given full freedom to search for any information related to retweeters and apply their own intuition.

Thus finally, out of the $2,508$ retweeters, 507 were labeled as fake and $2,001$ as genuine by the majority of the annotators. The average inter-annotator agreement was 0.78 based on Cohen's κ. Further statistics of the dataset are mentioned in Table 4.2.[4]

4.4 Problem statement

Informally, our problem definition is essentially a classification of Twitter accounts into fake or genuine, wherein for each user, we are given the timeline of his retweets within a short period of observation. Therefore, this can be thought of as a sequence classification task. More formally, given a topic H

[3]The annotators were social media experts and their age rangeed between 25-40 years.

[4]One may argue that the annotated dataset is small in size. We would like to emphasize that such annotation is extremely challenging and time-consuming since the annotators need to go through the entire timeline of a user to label the retweeting activity as fake/genuine. In our case, each annotator took 15-20 minutes on an average to annotate a single retweeter. Therefore, unlike conventional Twitter data collection, collecting *large data* for fake/genuine retweeter detection is extremely difficult. However, one of our immediate future directions would be to increase the size of the labeled dataset.

Notation	Denotation
RT	Set of retweets sorted in chronological order
N	Total number of retweets of a retweeter
u	A retweeter
t_N	Timestamp of the last retweet of a retweeter
$\lambda_u(t)$	Intensity function of retweeter u
α_u	Adjacency of retweeter u
μ_u	Base Intensity of retweeter u
ω	Exponential decay constant
LDA_g	LDA model learned on genuine retweeter data
LDA_f	LDA model learned on fake retweeter data
V_g	Feature vector obtained from LDA_g
V_f	Feature vector obtained from LDA_f

Table 4.3: Important notations and denotations.

(related to the popularity of some trending hashtag(s)), let $\mathbf{RT}$ be the set of retweet objects of a certain user, posted with respect to a particular trending topic and sorted in order of creation time. $\mathbf{RT} = \{RT_1, RT_2, ..., RT_n\}$, where n is the timestamp of the last retweet object in the interval of observation I. An object in this setting is a tweet object, provided by Twitter as a JSON file, which consists of all the details relevant to the tweet. For our purpose, we define a retweet object RT_i as a tuple, i.e., $RT_i = (t_i, W_i)$, where t_i is the creation timestamp of the retweet object, and W_i is the text content in the retweet object. The problem can therefore be framed as: given $\mathbf{RT} = \{RT_1, RT_2, ..., RT_n\} \in I$, can we predict whether the retweet set RT corresponds to the timeline of a fake user or a genuine user?

4.5 Preliminaries

4.5.1 Hawkes processes

Here, we present important definitions related to Hawkes processes, which are needed to explain our proposed method. Table 4.3 summarizes the notations used throughout the study.

A *point process* is a random process spanning the non-negative real line represented as time. It consists of a set of timestamps of events $T_1, T_2, \cdots$, occurring along the same line. Hence, T_i is interpreted as the time of occurrence of the ith event, and the T_i's are often referred to as arrival times [119].

A *counting* process, denoted by N_t, simply counts the number of events of a point process till time t $(0 \leq t < T_{n+1})$. If $T_n \leq t < T_{n+1}$, then $N_t = n$.

The *Inter-arrival times* are a sequence of τ_js consisting of the interval difference between each T_j and $T_{j+1}, j \geq 0$. In other words, $T_n = \sum_{j=1}^{n} \tau_j$.

A *Poisson* process is a simple case of a point process, where the inter-arrival times τ_i are independent, and each follows an exponential probability distribution with parameter λ [119]. λ is called the event intensity function, and in general, varies with time (i.e., $\lambda(t)$). It is constant for a Poisson process. This intensity function is an indicator of the average number of events happening or arriving between two-time intervals. A Poisson process with constant λ is termed as *homogeneous*, whereas when λ is a function of t it is *non-homogeneous*.

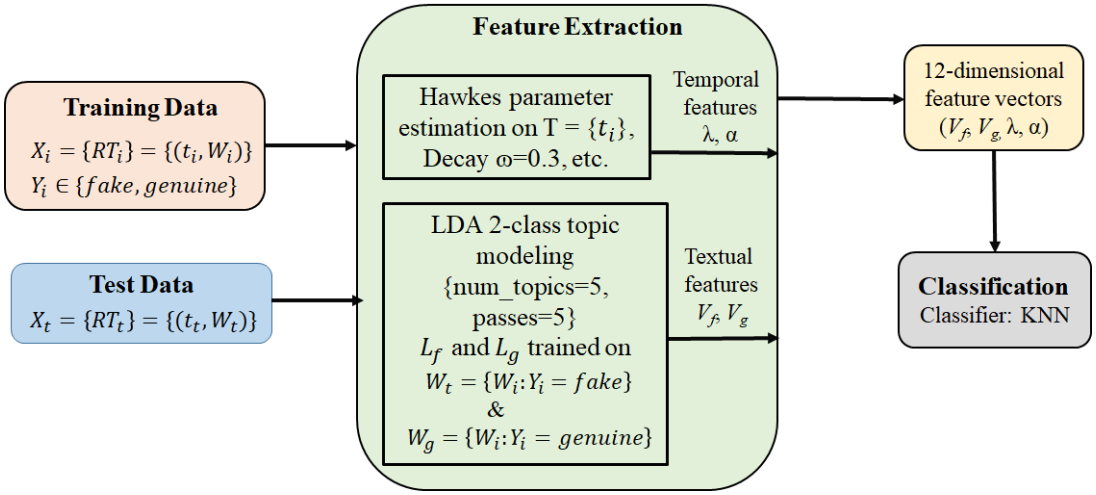


Figure 4.3: The methodology of the HawkesEye model. The orange block (training data) represents the input data used for training our model, and the blue block (test data) represents the test input for our trained model. The green block indicates a feature extraction strategy to calculate the temporal and textual features based on the Hawkes process and LDA topic modeling, respectively. The yellow block (12-dimensional feature vectors) represents the entire feature vector for our classification experiment. The grey block (classification) represents applying the best-performing classifier (KNN in our case) to differentiate between fake and genuine retweeters.

The general intensity function $\lambda(t)$ is only defined when we are given a history H_t of the event times occurring before time t, and hence is further termed as the *conditional* intensity function $\lambda(t|H_t)$.

A *self-exciting process* is a point process in which the event arrival rate is dependent upon past events, and the arrival of a new event causes the conditional intensity function to increase [119].

A Poisson point process is said to be a *Hawkes* process if its conditional intensity function takes the form:

$$\lambda(t|H_t) = \lambda_0(t) + \sum_{i:t>T_i} \phi(t - T_i)$$

where, H_t denotes the history; $\lambda_0(t)$ refers to the base intensity of the process, which can be a constant or a function of time; and $\phi(t)$ is the decay function, which models how the past event times influence the probability of a new event occurring at time t [119]. Hence, a Hawkes process is a special type of self-exciting point process where the intensity function is explicitly dependent on *all* previously occurred events. In other words, a future event is influenced by all the events that occurred in the past. For our modeling purposes, we use the exponential decay function, as it has been shown to be effective in modeling tweet influences in the past. Intuitively, the effect of a tweet or a retweet in the process dies out exponentially as time $t \to \infty$, i.e., the *interestingness* of the tweet reduces as time goes on.

4.5.2 LDA based topic modeling

Topic modeling refers to the modeling and inference of underlying 'topics' in a given collection of documents. Most topic modeling approaches assume multinomial distribution of topics over words and seek to identify those topics which lead to the distribution of words observed in the given documents. However, since tweets are normally very short in size and limited to 280 characters, we perform topic modeling on the aggregation of tweets as suggested in [120] where it was shown that the aggregation of similar tweets into individual documents significantly increases the topic consistency.

LDA is an unsupervised model that generates a mixture of latent topics from the collection documents, where each topic vector contains words from the collection with corresponding probabilities. Previous studies such as [121] also suggest that LDA gives successful results on tweets. Each of the documents is represented as multinomial *Dirichlet* distributions over the topics, denoted by θ. LDA further assumes that each topic is a multinomial Dirichlet distribution over words, denoted by ϕ. It then attempts to find the topics relevant to the words of a document by reverse-engineering the process. A topic z is sampled first from the distribution θ for each word in a document, and subsequently, a word is sampled from the Dirichlet distribution ϕ associated with z [121]. This procedure is repeated as many times as the number of words in the document to generate the LDA representation of the document. For our work, we consider the set of all retweets of a user as a single document, and using a collection of such documents, we infer the topics.

4.6 HawkesEye: Our proposed model

We adopt a pseudo model and feature-level fusion approach to propose HawkesEye that tackles the classification task. The parameters estimated from fitting models to two different aspects (temporal and textual) of the data are combined and used as features for further classification. Temporal characteristics are modeled using the Hawkes process, owing to its self-excitation property, whereas textual characteristics are modeled using LDA in a class-specific manner (c.f. Figure 4.3). As suggested in [122], the main idea to use Hawkes process to utilize temporal information is that it supposes that past events can temporarily raise the probability of future events, assuming that such excitation is either positive, additive over the past events, and exponentially decaying with time. It complies with the retweeting behavior of a Twitter user where a user who has frequently been tweeting in the recent past is more likely to tweet in the recent future, as opposed to a dormant user, who has little to no activity in the recent past. The main idea behind choosing LDA for topic modeling is because of its unsupervised nature, which does not require any pre-assigned topics. Neural-network based approaches such as LSTMs could do a better job given labels (supervised nature), which may perform better with document-level topic modeling as those labels are easier to come by, but word-level topic modeling is a challenge. We found a neural-network-based topic modeling approach named LDA2Vec, where for each text fragment (size of a paragraph), a dense vector representation is learned, similar to the learned word vectors. The downside of this approach is that the context/paragraph vectors resemble typical word vectors, making them less interpretable as the output of LDA[5]. We refrained from using any explicit feature engineering, such as the inclusion of user profile information (number of followers, a profile description, timezone, etc.), geolocation features, or derived features such as sentiment, emotion, etc., as they require domain experts for manual feature curation.

4.6.1 User specific Hawkes process

To model the time-series behavior of the retweeters, we associate a separate univariate Hawkes process for each user. Taking their normalized retweet timestamps as input, we fit them into the model and infer the required parameters. These parameters are then used as features in the classification step. The intuition is that every user will be characterized by these parameters, and users with similar retweeting patterns in their timelines will have minimal variation in the inferred parameters. Therefore, we expect that users of the two classes will be differentiated well with these parameters as features.

[5]https://www.datacamp.com/community/tutorials/lda2vec-topic-model

4.6.1.1 Intensity function

We have till now assimilated that each Hawkes process has an intensity function associated with it, which is used for parameter inference. Let the intensity function for each process be defined as:

$$\lambda_u(t) = \mu_u + \alpha_u \sum_{j<t} K(t-j) \tag{4.1}$$

where u is a given user, μ_u is the base intensity of the intensity function for a user, α_u is the adjacency of the user, and

$$K(t-j) = e^{-\omega(t-j)}$$

with ω being a small constant value.

On inspecting this intensity function, we note the following with regards to each term:

- The summation term is the one that takes into account the effect of all tweets before the current tweet, which allows for the influence of previous tweets or retweets on a current retweet.
- The μ_u term takes into account the initial retweeting tendency of the user. In other words, a higher μ_u denotes a higher tendency to start retweeting.
- The α_u term takes into account the influence of the previous retweets posted by the user on the current one. In other words, a higher α_u indicates a stronger influence of the previous retweets.
- As the influence of a tweet naturally decays over time, the term $K(t-j)$ captures the decay in the influence of a tweet at time j with respect to current time t. This decay has been assumed to be exponential as it best describes any tweeting influence behavior.

We estimate α_u and μ_u parameters for each user while keeping ω fixed at 0.1, following the suggestion of Lukasik [118].

4.6.2 Likelihood function

To estimate the parameters for the intensity function, the likelihood function is defined as follows:

$$L(t,u) = \prod_{n=1}^{N} \lambda_u(t_n) \times e^{-\int_0^{t_N} \lambda_u(v)dv}$$

The log-likelihood function thus becomes

$$L(t,u) = \sum_{n=1}^{N} ln(\lambda_u(t_n)) - \int_0^{t_N} \lambda_u(v)dv \tag{4.2}$$

Let

$$\Gamma(t_N) = \int_0^{t_N} \lambda_u(v)dv$$

This can further be derived as follows:

$$
\begin{aligned}
\Gamma(t_N) &= \int_0^{t_N} \lambda_u(v)dv \\
&= \int_0^{t_1} \lambda_u(v)dv + \sum_{i=1}^{N-1} \int_{t_i}^{t_{i+1}} \lambda_u(v)dv \\
&= \int_0^{t_1} \mu_u dv + \sum_{i=1}^{N-1} \int_{t_i}^{t_{i+1}} \mu_u dv + \sum_{t_j<v} \alpha_u e^{-\omega(v-t_j)}dv \\
&= \mu_u t_N + \alpha_u \sum_{i=1}^{N-1} \int_{t_i}^{t_{i+1}} \sum_{j=1}^{i} e^{-\omega(v-t_j)}dv \\
&= \mu_u t_N + \alpha_u \sum_{i=1}^{N-1} \sum_{j=1}^{i} \int_{t_i}^{t_{i+1}} e^{-\omega(v-t_j)}dv \\
&= \mu_u t_N - \frac{\alpha_u}{\omega} \sum_{i=1}^{N-1} \sum_{j=1}^{i} [e^{-\omega(t_{i+1}-t_j)} - e^{-\omega(t_i-t_j)}] \\
&= \mu_u t_N - \frac{\alpha_u}{\omega} \sum_{i=1}^{N-1} [e^{-\omega(t_N-t_i)} - e^{-\omega(t_i-t_i)}] \\
&= \mu_u t_N - \frac{\alpha_u}{\omega} \sum_{i=1}^{N-1} [e^{-\omega(t_N-t_i)} - 1]
\end{aligned}
$$

Thus, following [123] we get the final log-likelihood expression as:

$$
l(t,u) = \sum_{i=1}^{N} ln(\mu_u + \alpha_u \sum_{j=1}^{i-1} e^{-\omega(t_i-t_j)}) - \mu_u t_N + \frac{\alpha_u}{\omega} \sum_{i=1}^{N} [e^{-\omega(t_N-t_i)} - 1] \tag{4.3}
$$

As pointed out by Ozaki [124], this can be expressed in a recursive form due to the properties of the first exponential summation. Ozaki [124] also gave the partial derivatives and the Hessian of the function. The parameters can be calculated by maximizing this log-likelihood function using methods such as gradient descent. Also, each derivative calculation can be carried out in $\mathcal{O}(N)$ complexity with the recursion. This makes the algorithm particularly faster than other feature-based techniques.

4.6.3 Class-specific LDA

To model the textual content in the retweets, we generate summary topic vectors for each retweeter based on the textual content of their retweets. Assuming the collection of retweets over the entire retweet timeline of each retweeter as a 'document', we input the collection of documents to the LDA model.

Due to the skewness of data which is observed between fake and genuine retweeters (with only 20% of the retweeters being fake), if we feed the entire data into one LDA model, the topic vectors would inevitably be biased towards the genuine users due to the unsupervised nature of LDA. Thus, to enable better class representation, we perform class-specific LDA, i.e., we train two LDA models – one model is

trained on the fake retweeters' documents (LDA_f) and the other on the genuine retweeters' documents (LDA_g). Once the training is complete, for a given document, we can then fuse the two topic vectors V_f and V_g obtained from LDA_f and LDA_g respectively to yield a final topic vector V_{res}, which is equally representative of both the classes.

We find that choosing the number of topics as 10 and then combining the fake and genuine topic vectors generally gives the best performance in terms of classification accuracy. However, we also vary the number of topics (dimension of V_g and V_f) and show the results in Figure 4.4(b).

4.7 Detailed methodology

Armed with the tools discussed in the previous section, we divide our methodology into two broad steps: Feature Extraction and Classification (c.f. Figure 4.3).

4.7.1 Feature extraction

In this step, we take the set of retweets of a retweeter as input and generate the corresponding feature vector. Given $\mathbf{RT} = \{RT_1, RT_2, ..., RT_n\}$, we compute the following features.

(i) Topic Vector Weights

From $RT_i = (t_i, W_i)$, we extract the textual component i.e., W_i in each retweet and create $\mathbf{W} = \{W_1, W_2, ..., W_n\}$ which is to be used for extracting topic vector features. We concatenate all W_i in $\mathbf{W}$ to convert it into a document and then pass it through both LDA_f and LDA_g for inference of topic vector weights to create the corresponding vectors V_f and V_g. These features thus correspond to the textual content of the tweets. Keeping in mind that the number of topics chosen for both LDA_f and LDA_g is 10, we obtain a 20 dimensional vector.

(ii) Hawkes parameters

From $RT_i = (t_i, W_i)$, we extract the time component, i.e., t_i in each retweet and create $\mathbf{T} = \{t_1, t_2, ..., t_n\}$ which is to be used for extracting temporal features. All timestamps in $\mathbf{T}$ are fitted into a Hawkes process; the corresponding λ and α parameters are learned, and then used as features. The features V_f, V_g, λ and α obtained from the above methods are then concatenated, resulting in a 12-dimensional vector. Once all such features are calculated for each user, we then proceed to classification.

4.7.2 Retweeter classification

Here, we classify retweeters based on the feature vector obtained from the previous stages. We reiterate our intuition that retweeters who are similar in behavior will tend to have similar feature vectors and hence lesser distance between them due to similarities in retweeting content as well as temporal tendencies. As a logical consequence, we consider a K-Nearest Neighbour (KNN) based classification method, which takes into account labels of vectors which are nearest to the sample in question. For a more rigorous comparison,

we also implement other classification models such as Naive Bayes, SVM, Logistic Regression (LR), and compare their performances.

We perform hyperparameter optimization to obtain parameter values that generate the best results. For instance, we use KNN method with $K = 1$; multinomial logistic regression, SVM with linear kernel, etc. We use Euclidean distance as the distance metric for KNN. Since KNN turns out to be the best model, we also show the accuracy of KNN by varying K in Figure 4.4(a).

The pseudocode of the algorithm is shown in Algorithm 1. We also provide the implementation details and time complexity analysis of HawkesEye.

4.7.2.1 Implementation details of HawkesEye

We tuned the following hyperparameters in Hawkes process while fitting each retweeter's timeline:

- *decay* or ω - as explained above, this is the exponential kernel decay parameter, and results in ideal performance when set to 0.1.
- *gofit* - refers to the metric used to calculate the goodness of fit of the parameters estimated for each retweeter's timelines. The goodness-of-fit metric used is 'likelihood'.
- *penalty* - in order to avoid overfitting, the loss function has a penalty factor added to it. For our purposes, it is chosen as l2 ridge regression penalty.
- *solver* - the solver used for obtaining the parameters from the likelihood function is an accelerated gradient descent ('agd') solver.
- other hyperparameters - the learning rate is set as 10^{-6}, *maximum_iterations* as 1000, and

The following hyperparameters were adjusted in LDA topic modeling:

- *num_topics*, the number of topics to be extracted from the collection of documents is set as 10.
- *passes*, the number of passes of LDA made on the collection of documents is kept as 5.

All experiments were conducted on a system with 8GB RAM and Intel i7-7700HQ chipset. For LDA topic modeling, the *LdaMulticore* [125] API of the python library **gensim** was used. For parameter estimation in Hawkes baseline, the *HawkesExpKern* API of the **tick** library was used [126]. All competing methods were implemented in Python version 3.5.

4.7.2.2 Time complexity analysis

For M retweeters, let K be the average number of retweets per retweeter in the period of observation. Let L be the average length of a retweet, i.e., number of tokens per retweet.

Topic vector weights: Since LDA is run on the collection of retweets per user as one document, and since LDA is linear in the number of documents [127], we observe that the time complexity in this step is $\mathcal{O}(MKL)$, since KL is the size of the document per retweeter.

Hawkes parameters: In our case, i.e., Hawkes processes with exponential kernel functions, parameter estimation is linear in the number of timestamps [128]. Thus, since we have L timestamps per retweeter, and M such retweeters, the total time complexity for Hawkes process parameter estimation will be $\mathcal{O}(ML)$.

Thus the total time complexity for feature extraction is $\mathcal{O}(ML(K+1)) \approx \mathcal{O}(MLK)$.

Finding nearest neighbours: In the KNN calculation, we have as many data points as the number of retweeters, and D is the feature dimension per user. Clearly, for M samples of D dimensions, the basic runtime complexity for KNN is $\mathcal{O}(MD + KM)$.

Algorithm 1: Pseudocode of HawkesEye

Input: Set of retweets of a retweeter.
Output: Label of a retweeter (fake/genuine).
Let $\mathbf{U} = \{U_1, U_2, \ldots, U_m\}$, where U_i is i^{th} retweeter.
Let $\mathbf{R_{u_i}} = (W^1_{u_i}, t^1_{u_i}), (W^2_{u_i}, t^2_{u_i}), \ldots, (W^n_{u_i}, t^n_{u_i})$, where
R_{u_i} = Retweet timeline of user u_i.
$W^j_{u_i}$ = Textual content of j^{th} retweet by user u_i.
$t^j_{u_i}$ = Timestamp of j^{th} retweet by user u_i.
$LDA_f \rightarrow$ LDA model trained on fake training data.
$LDA_g \rightarrow$ LDA model trained on genuine training data.
Step 1: *Feature extraction*
foreach *user* u_i *in* U **do**
    $\mathbf{R_{u_i}} = \{(W^1_{u_i}, t^1_{u_i}), (W^2_{u_i}, t^2_{u_i}) \ldots (W^n_{u_i}, t^n_{u_i})\}$
    $\mathbf{D_{u_i}} = \{W^1_{u_i} \cup W^2_{u_i} \cup \cdots \cup W^n_{u_i}\}$
    $\mathbf{V^f_{u_i}} = LDA_f(D_{u_i})$
    $\mathbf{V^g_{u_i}} = LDA_g(D_{u_i})$
    $\mathbf{T_{u_i}} = \{t^1_{u_i}, t^2_{u_i} \ldots t^n_{u_i}\}$
    $\lambda, \alpha = LearnHawkesParameters(\mathrm{T}_{u_i})$
    $\mathbf{V} = \{V^f_{u_i} \cup V^g_{u_i} \cup \alpha \cup \lambda\}$
end
Step 2: *Perform classification on* ***V*** *using KNN*

4.8 Experiments

In this section, we lay out the evaluation setup for our model. We start with the data splitting for training and testing, baseline methods considered for comparison, followed by comparative evaluation.

4.8.1 Cross validation

To resolve the underlying skewness in the data, we perform the evaluation using stratified 5-fold cross-validation, implying that each split approximately contains an equal number of instances for each class for training and testing. Stratification is essential to this problem, as the skewness in the class distribution is very high ($\sim$ 4:1) and, as such, hinders the performance of any classifier if trained on an imbalanced dataset. We perform stratification by sampling as many genuine retweeters as fake retweeters for each fold of the evaluation. Thus, in each fold, for each of 507 fake retweeters being sampled from the dataset, 507 genuine retweeters are sampled, and they are combined together to form the training and test data for that fold. The sampling is repeated 20 times, and the average results are reported (see Section 4.8.4). We also show the performance of the competing methods on the imbalanced dataset (see Section 4.8.5).

Model	Precision	Recall	F1-Score	Accuracy
SCoRe	0.669	0.735	0.647	0.706
SpamBot	0.551	0.502	0.406	0.592
Language Model	0.908	0.762	0.807	0.889
LSTM	0.397	0.501	0.443	0.595
HawkesEye (KNN)	**0.964**	**0.960**	**0.960**	**0.950**
HawkesEye (NB)	0.747	0.759	0.668	0.714
HawkesEye (SVM)	0.715	0.779	0.695	0.744
HawkesEye (LR)	0.691	0.743	0.658	0.701
HawkesEye (KNN) + Textual	0.831	0.767	0.783	0.817
HawkesEye (KNN) + Temporal	0.779	0.788	0.744	0.734

Table 4.4: Performance of the competing methods on *balanced dataset*. The results are reported after taking the average of 5-fold cross validation. We repeated the experiment for a fixed number of times (hyperparameter optimization) and choose the parameter set that yields the best value for the evaluation metrics.

4.8.2 Baseline methods

To the best of our knowledge, such a classification based on a window of observation, and on a single trending topic is the first one ever attempted. Thus, we have no suitable state-of-the-art to compete against and resort to utilizing some relevant state-of-the-art that is only approximate for the task at hand. We briefly describe them in the following section.

Baseline I: ScoRe

Our previous method *SCoRe* [3] is a recent machine learning-based classifier that uses an extensive list of 64 features for classification of retweeters as fake blackmarket retweeters or genuine. The spectrum of features comprises of *profile features, social network features, user activity features* and *fluctuation level features*. Each feature set targets a specific part of a user's characteristics. These features, once extracted for every user, are used to run state-of-the-art classifiers (SVM is reported to perform the best), and show significant performance improvements over existing traditional feature-dependent classifiers. We use their feature extraction techniques on our dataset and run SVM to classify fake and genuine retweeters.

Baseline II: SpamBot

SpamBot [129] is another supervised classifier that uses a set of content and network-based features for classification. Content-based features are targeted at the textual content of the tweet object, i.e., *number of user mentions, number of links*, etc. Network-based features correspond to heuristics to describe the user's network on Twitter, i.e., *number of followers, number of followees*, etc. The basic assumption is that the likelihood of a genuine user to post duplicate tweets is lower than that of a bot. To compare our method with this baseline, we use their proposed set of features and their suggested classifier (Naive Bayes). Another motivation behind taking SpamBot as a competing method is to show that the bot detection algorithm fails miserably to detect fake retweeters as the fake retweeters are generally normal human beings [4].

Model	Fake			Genuine		
	Precision	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**
SCoRe	0.708	0.244	0.360	0.836	0.974	0.902
SpamBot	0.890	0.104	0.184	0.816	0.998	0.898
Language Model	0.524	0.650	0.582	0.906	0.850	0.876
LSTM	0.698	0.194	0.304	0.39	0.450	0.430
HawkesEye (KNN)	0.730	0.626	**0.674**	**0.910**	0.940	0.924
HawkesEye (NB)	0.690	0.600	0.640	0.909	0.940	0.919
HawkesEye (SVM)	**0.968**	0.416	0.580	0.872	**0.996**	**0.931**
HawkesEye (LR)	0.926	0.476	0.626	0.882	0.990	0.930
HawkesEye (KNN) + Textual	0.684	**0.642**	0.660	0.912	0.924	0.916
HawkesEye (KNN) + Temporal	0.430	0.614	0.420	0.882	0.656	0.724

Table 4.5: Performance of the competing methods on *imbalanced dataset*. The results are reported after taking the average after 5-fold cross-validation. We cannot report accuracy for class-wise performance measurement because there is no notion of true negative for each class. We repeated the experiment for a fixed number of times (hyperparameter optimization) and choose the parameter set that yields the best value for the evaluation metrics.

Baseline III: Language model

Since there is also a textual classification component to our model, comparing it with naive natural language classifiers also leads us to some interesting observations. We observe the advantages of our topic modeling strategy as opposed to using *TF-IDF vectors* with *multinomial Naive Bayes* as this baseline. The unique nature of text in tweets also makes it an interesting comparison between documents which have a far higher number of words per document than a tweet, with such few characters.

Baseline IV: LSTM

We also compare HawkesEye with a neural network-based classifier, which takes the sequence of retweets per user as input and predicts a class label. Neural networks with *LSTM* (Long Short-Term Memory) [130] cells have been commonly found to be useful for sequence classification problems which deal with the time-series data. Therefore, we use the same as one of our baselines. However, due to the absence of large amounts of annotated data, neural networks are generally prone to overfitting and thus give poor performance.

Baseline V: HawkesEye + Textual features

We also evaluate how the textual and temporal features affect the overall predictive power when given independently to a classification model using the same evaluation metrics, as discussed above. Since ours is a feature fusion approach, surveying the contributions of each type of feature helps us realize the underlying nature of the data. This baseline is designed by taking into account only the textual features obtained from LDA.

Baseline VI: HawkesEye + Temporal features

Following the previous idea, we design another baseline by considering only the temporal features obtained from the Hawkes process.

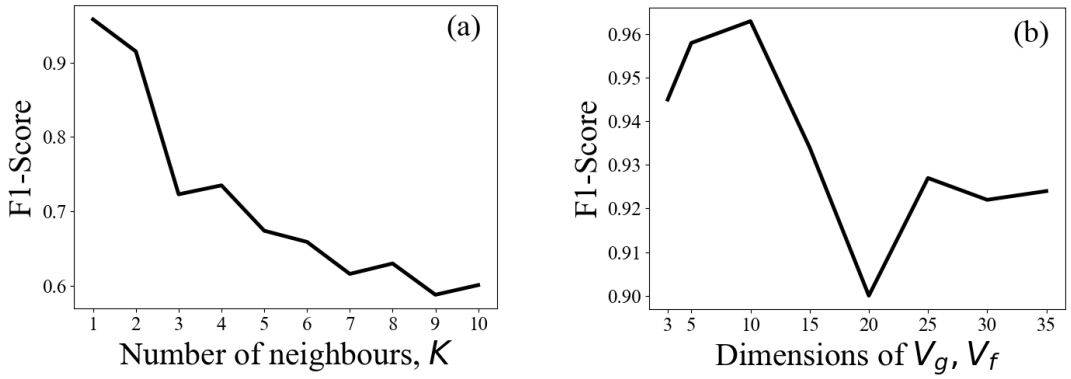


Figure 4.4: F1-Score of HawkesEye by changing (b) K, the number of nearest neighbors in KNN, and (b) the size of the dimension of V_g and V_f (i.e, number of topics).

Note that the performance of Baselines V and VI can also be considered as a study of feature ablation of our model.

4.8.3 Evaluation metrics

Since the task is a binary classification, we evaluate our model using standard metrics used for the same – accuracy, precision, recall, and F1-Score. All the results displayed are macro-averaged over 5-fold cross-validation[6].

4.8.4 Comparative evaluation on balanced dataset

Table 4.4 shows the performance of the competing methods on the balanced dataset. We notice that HawkesEye with KNN classifier outperforms all other models significantly in terms of all the evaluation metrics used. It also shows a considerable increase in performance over the language model, indicating that topic vectors give the model better discriminating power than using tokens as is. Among all the existing baselines, the language model seems to be the best baseline method. We also note that KNN with $K = 1$ (c.f. Figure 4.4(a) for the performance with other values of K) performs the best with HawkesEye as compared to SVM, Logistic Regression, and Naive Bayes on the same feature set, which reinforces the intuition we previously had that the retweeters with similar retweeting behavior tend to come closer in the feature space. SCoRe and the Language model follow behind, with SpamBot performing the worst among the baselines.

The feature-wise evaluation demonstrates that textual features play a more significant role than the temporal information. This is not surprising as most fake retweeters are human retweeters, and as such, know how to mimic human-like behavior so as to evade classification as bots. However, due to the diversity of requirements by clients, more often than not, the text in their tweets have a lesser correlation to the topic being discussed than those retweeters who genuinely voice their opinions over the topic. Therefore, they are decidedly discriminated by their textual features.

[6]The results are the average of precision, recall and F1 score values over 5-fold cross validation, hence the direct mathematical relation may not be applicable.

4.8.5 Comparative evaluation on imbalanced dataset

We further repeat a similar study, as mentioned in Section 4.8.4 on the imbalanced dataset. Here, we consider all the users (i.e., 2508 users) and maintain the same ratio of fake and genuine retweeters both in the training and test sets as present in the overall dataset. The purpose of this experimental setup is to show how effectively HawkesEye is able to spot fake retweeters from the user population, which is already biased towards the genuine retweeters (note that this experimental setup is more realistic than the previous setup). Table 4.5 shows that in the case of imbalanced data distribution, there is a difference in the predictive performance for fake versus genuine retweeters. As the genuine retweeters are much higher in number, their F1-Scores can be observed to be much better than those of the fake retweeters. However, despite the drastic difference in relative proportion (80:20), the performance across all the HawkesEye models is nearly comparable. At the same time, all HawkesEye models clearly outperform the baseline models in terms of overall class-wise F1-Score as well, indicating a significant improvement over the current state-of-the-art. Interestingly in case of fake class, while HawkesEye with SVM and LR outperform that HawkesEye with KNN in terms of precision, their recall value is quite small. The latter model maintains a balance between precision and recall, thus outperforming others in terms of F1-Score.

4.9 Conclusion

In this work, we proposed HawkesEye, a novel classifier based on Hawkes process and LDA to identify fake retweets from their retweet timeline by taking into account textual information as well as temporal information. Exhaustive experiment on a novel dataset showed that HawkesEye outperforms state-of-the-art baselines by a significant margin. We also provided the parameter selection strategy of the model.

As a byproduct of the study, we curated a new dataset and manually annotated retweeters into fake and genuine. To our knowledge, this is a unique dataset which focuses on a very narrow but trending topic and provides a complete timeline of each user during a specific time period. Such a dataset is more realistic and feasible to obtain for researchers who find difficulty in accessing the entire Twitter feed. With our proposed model `HawkesEye`, it may be difficult to identify shallow retweeters due to the difficulty to identify textual and profile features. However, this problem is handled in our proposed `CoReRank` model, which takes care of the "cold start" parameter where for some users as well as tweets, multiple supports (retweets/quotes) may not be available. The codes used in this study are available at https://github.com/LCS2-IIITD/HawkesEye.

Part II

Modeling network of collusive users

5. Detecting collusive retweeters by incorporating multiple views of user

With the rise in popularity of social media platforms like Twitter, having higher influence on these platforms has a greater value attached to it since it has the power to influence many decisions, in the form of brand promotions and shaping opinions. However, blackmarket services that allow users to inorganically gain influence are a threat to the credibility of these social networking platforms. Twitter users can gain inorganic appraisals in the form of likes, retweets and follows through these blackmarket services either by paying for them, or by joining syndicates wherein they gain such appraisals by providing similar appraisals to other users. These customers tend to exhibit a mix of organic and inorganic retweeting behavior, making it tougher to detect them.
In this work, we investigate these blackmarket customers engaged in collusive retweeting activities. We collect and annotate a novel dataset containing various types of information about blackmarket customers, and use these sources of information to construct multiple user representations. We adopt Weighted Generalized Canonical Correlation Analysis (WGCCA) to combine these individual representations to derive user embeddings that allow us to effectively classify users as: genuine users, bots, promotional customers and normal customers. Our method significantly outperforms state-of-the-art approaches (32.95% better macro F1-score than the best baseline).

5.1 Introduction

Twitter, being a micro-blogging site, provides a platform to share various kinds of content – personally curated content, product promotions, and opinions which give rise to online social movements. It also provides its users an option to appraise other tweets based on their liking of the content in two different ways: (i) *Content-level affirmation* (Primary method of Twitter appraisal), such as `Retweets`, `Quotes` and `Likes` – this provides a way of rebroadcasting messages and confirms the user's agreement with the message being important to others, and (ii) *User-level affirmation* (Secondary method of Twitter appraisal), such as, `Follows`, implying the general liking of most of the tweets by that user.

The count of these appraisals, such as number of likes, retweets or follows, indicates a sense of crowd-sourced agreement on that tweet and user, and thus, they also determine the influence of the tweet as well as the author of the tweet. Most of the time, excessive appraisal of certain content or user can make that topic or user *trending* on Twitter. This system of appraisal on Twitter creates motivation to falsely gain excessive number of likes, retweets or follows. This has also led to the creation of certain **blackmarket services**, that facilitate such inorganic appraisal (see Section 5.3). These blackmarket services are either paid, or comprised of a group of individuals who are ready to barter appraisals among themselves. These users, who are involved in "intentional manipulation" using such blackmarket services, and who try to

Table 5.1: Macro F1-score of the competing methods. None of the existing methods can detect collusive retweeters accurately.

Bot Detection [57]	Spam Detection [84]	Fake Account Detection [58]	Sync. Fake Retweeter Detection [1]	Our Method
0.680	0.645	0.688	0.269	0.869

create a false impression of the popularity of their tweets or accounts through a higher number of retweets, likes or follows, are referred to as **collusive users**. In this study, we focus on detecting users involved in collusive retweeting activities (*henceforth called as 'collusive retweeters'*). For our purposes, we use the notation *collusive users*, *collusive retweeters* and *customers* interchangeably in this work, all referring to *collusive retweeters*. Although we are interested in detecting individual collusive users, the term *'collusion'* has been used because these users act in deceitful cooperation via blackmarket services, which is in contempt of Twitter's Terms of Service.

Technical challenges of detecting collusive retweeters: The blackmarket services that provide a platform to these collusive retweeters have set up an intelligent ecosystem for providing inorganic appraisal to these users. Users can gain retweets on their own tweets for no cost at all by retweeting the tweets submitted by other users on the platform, making them a part of the ecosystem. Furthermore, the design of this ecosystem makes their detection even more challenging because:

- Collusive retweeters are not completely bots, but mostly human accounts. Therefore, bot detection algorithms perform poorly in detecting them (first column of Table 5.1).
- Collusive retweeters cannot be categorized into spam or fake accounts either, as these accounts also show significant amount of genuine user activity, with a well maintained profile. Hence, spam or fake account detection algorithms also do not detect them accurately (second and third columns of Table 5.1).
- They show a mix of *organic* and *inorganic* retweeting activity, i.e., they participate in retweeting activity out of genuine interest for content, but also, for retweeting the tweets that other users have put up on these blackmarket services, in order to gain retweets for their own tweets. Furthermore, the proportion of organic and inorganic activity that a user might display is not fixed. It depends upon each individual's own behavior.
- Lastly, collusive retweeters are not synchronous in their retweeting activities. Hence, algorithms that leverage the synchronous behavior of fraudulent retweeters, like [1], fail to make a mark when it comes to detection of collusive retweeters (fourth column of Table 5.1). We detail the asynchronous behavior of these collusive retweeters below.

There has been a lot of research on detection of fraudulent activities on Twitter such as detection of bots, fake followers and social spam detection. However, the problem of detecting manipulation of content affirmation is largely untouched. Giatsoglou et al. [1] recently tried to tackle this problem, by stating that spam retweeters have a synchronous behavior in terms of their retweeting activity and retweet the same content at the same time. However, we observe that such synchronous behavior is not followed by collusive users, as shown in Fig. 1 of [3]. It shows that unlike normal retweet fraudsters where Arr-MAD (mean absolute deviation of retweet's inter-arrival times of retweet threads) is almost invariant with respect to lifespan, collusive retweeters shows an increasing trend. This is because most of these users who are involved in blackmarket services are users seeking to appraise other content in order to get appraisal on their own content. Thus, they may not have any motive for targeted appraisal of any particular tweet. Also, blackmarket services do not mandate their users to retweet content at a particular time. Thus, these services have no control over the synchronicity of the collusive users. Aggarwal et al. [53] made a recent attempt at detection of collusive followers, but not towards collusive retweeters. They also pointed out that the existence of such blackmarket services are a threat to the credibility of Online

Social Networks (OSNs). It is also important to notice that even the in-house algorithms of Twitter have been unsuccessful at detecting such collusive retweeters – based on the observation that in our dataset, 72 collusive retweeters were actually verified users marked by Twitter.

In our previous study [3], we provided a supervised method to detect collusive retweeters affiliated to blackmarket services. We had only explored user attributes in order to mark their retweeting pattern and only relied on attribute-level features that can be extracted from the user's own profile only, such as their retweet intervals, the length of their profile name, etc. However, collusiveness is a multifaceted characteristic, and we can detect it in a better way by utilising both attribute-level as well as network-level features, where we study features extracted from the user's Twitter network. In this work, we present a comprehensive evaluation of collusive retweeters from a multiview perspective, along with our previous study. We also show the superiority of our new approach over our previous work [3], as well as other state-of-the-art approaches dealing with fake activity detection. In particular, the contributions of this work are as follows:

C1. We present a detailed study of the collusive activities controlled by the blackmarket services. To our knowledge, this is the first work which presents the entire landscape of the blackmarket activities in a thorough and rigorous manner.

C2. We propose a multiview learning based approach to detect collusive retweeters involved in blackmarket services. The idea behind the approach is to create user representations from the content, attributes and social network of the users. We create six different views: two attribute-level views and four network-level views.

C3. We describe an approach to learn a combined representation for a user from multiple views using Weighted Generalized Canonical Correlation Analysis (WGCCA). We show how each view can be more or less helpful for our task (using t-SNE visualizations) and weigh each view differently for optimal performance.

C4. We conduct experiments by training state-of-the art classifiers on the combined user representations to detect three types of collusive users: bots, promotional collusive users and normal collusive users. For this multi-class classification problem, our approach outperforms the best baseline (`BotoM`) by 32.95%, and our previous work (`SCoRe`) by 4.55% in terms of F1-score (macro). We also conduct an experiment by combining all types of collusive users to consider the problem as a binary classification problem. Here, we find that our approach outperforms the best baseline (`FakeAcc`) by 26.31%, and our previous work (`SCoRe`) by 14.34% in terms of F1-score (macro).

C5. We collect a novel dataset of 1807 collusive and 2706 genuine users (data collected between March 2018 and August 2018). All the collusive users are manually annotated by human annotators into three categories: bots, promotional collusive users and normal collusive users, based on the guidelines given to them. We also collect the profile information and timelines of these users.

The remainder of this work is comprised of the following sections: Section 5.2 reviews the related work. We discuss the different types of blackmarket services in Section 5.3. Section 5.4 describes our dataset. Section 5.4 also outlines the guidelines given to the human annotators for labeling of collusive users. We present the user representations using attribute and network level features in Section 5.5. Section 5.6 contains the experiments conducted using this dataset, and Section 5.7 describes the results we obtained. Section 5.8 presents the case studies. Finally, Section 5.9 closes the study with concluding remarks.

5.2 Related work

Several anomaly detection techniques have been designed to identify malicious individuals, online fraudsters, spammers etc. across multiple online platforms. Despite the fact that a lot of literature exists on detecting these fraudulent users and bots in various OSNs, detection of collusive activities has hardly been studied. We discuss the past efforts by dividing the existing literature into two parts: (i) Detection of fraudulent activities, and (ii) Study of collusion in OSNs.

5.2.1 Detection of fraudulent activities in OSNs

Various studies have been conducted on the detection of fraudulent and spamming activities on online media. Shah et al. [6] provided a detailed analysis of the blackmarket services and divided them into two types based on the mode of service - Premium and Freemium. Benevenuto et al. [60] generated features from tweets and user behavior to detect spammers. Giatsoglou et al. [63] proposed , which uses a weighted cascade model to simulate the retweeting activity of genuine and fraudulent users. Chu et al. [85] classified a user into human, bot or cyborg based on tweeting behavior, tweet content and user account properties. Hu et al. [66] identified spammers based on the networking properties. Gupta et al. [131] investigated spam campaigns by detecting spammers who use phone numbers to promote these campaigns on Twitter. A huge amount of work has been done on identifying bots on Twitter [57, 103, 132]. Some other studies focused on detecting frauds using URLs embedded in tweets [101, 133, 134] and blacklisted URLs [135, 136]. Recently, a series of works investigated fake followers on Twitter. It is reported that there is an increase of 1-3% in fake followers for a Twitter account[1]. Castellini et al. [9] designed an anomaly detector using denoising autoencoder to detect fake followers. Cresci et al. [52] used 49 distinct features and 8 different 'glass-box' and 'black-box' machine learning classifiers to detect fake followers on Twitter. [55, 56] are some of the network-based approaches to detect fake followers in Twitter. There also exists a tool, called Fake Follower Check[2], which detects fake followers based on features such as ratio of friends to followers, too many retweets than tweets, incessant use of spam phrases such as 'diet', 'make money', etc. Giatsoglou et al. [1] proposed `NDSync` to tackle the problem of synchronous fraudulent activities. `NDSync` spots retweet fraudsters, based on features collected from the retweet threads.

Several other studies attempted to detect fraudulent and spam activities on different social media platforms. Beutel et al. [88] detected lockstep behavior in Facebook Page Likes. Jindal et al. [90] investigated spam detection on e-commerce websites such as Amazon. Li et al. [137] identified review spam by employing supervised machine learning methods based on the crawled reviews from Epinions. Chen et al. [32] detected fake views caused by robots generating requests or reports in video platforms.

5.2.2 Study of collusion in OSNs

Though collusion has not been studied much in the literature, it has recently gained a considerable amount of attention among researchers because of the techniques used to offer the services. Thomas et al. [75] investigated underground market profiting from Twitter credentials and its affects. They identified 27 account traders from blackhat forums, freelance websites, etc. linked to Twitter, who bypass the automatic registration using compromised hosts and CAPTCHA solvers. Liu et al. [76] tackled the problem of a new type of malicious crowdturfing following relationship, called 'voluntary following'. They proposed

[1] https://www.gshiftlabs.com/social-media-blog/the-fake-followers-epidemic/

[2] https://www.socialbakers.com/blog/1099-fake-followers-check-a-new-free-tool-from-socialbakers

which incorporates both structural information in user behavior graphs and prior knowledge gained from the follower markets. Motoyama et al. [77] analyzed six different underground forums to understand the dynamics of social networks present on these forums. Stringhini et al. [54] analyzed the growth and dynamics of Twitter follower markets. They reported the properties of the customers of these markets. Arora et al. [48] detected tweets posted on blackmarket sites by using a multitask learning framework to classify tweets as blackmarket or genuine. Zheng et al. [138] identifed sockpuppets - a set of aliases (different user-ids) controlled by a single user (aka 'Puppetmaster') in online discussion forums. Kumar et al. [10] studied sockpuppetry across nine online discussion communities. They used behavioral traces of a user such as IP addresses and data collected from a user session to identify sockpuppet groups. They also revealed that sockpuppets use more singular first-person pronouns, write shorter sentences, and swear more and participate in more controversial discussions. Solorio et al. [139] proposed a semi-supervised approach to detect linked identities in Wikipedia. Gupta et al. [140] proposed an approach to detect malicious retweeter groups.

Several other studies reported collusiveness on other platforms such as e-commerce sites [49]. Chen et al. [141] detected multiple algorithmic pricing strategies adopted by sellers in Amazon marketplace. They identified how sellers change prices of their product in order to win the Buy Box more frequently. Hannak et al. [142] studied two prominent online freelance marketplaces of online labor markets and reported how real-world bias can manifest these markets and harm the employment opportunities. Vidros et al. [143] examined the diverse aspects of employment scam and showed its resemblance with the existing fraudulent activities such as vandalism, cyber bullying etc.

5.2.3 Differences with our previous studies

In our previous study [3], we attempted to detect collusive retweeters using a supervised approach. We focused on the freemium service model, mainly due to easy accessibility of data. We divided the freemium services further into three types: social-share services, credit-based services and auto-time retweet services. We then collected a set of 64 features, that can help distinguish the different types of users. The features were then used to run six state-of-the-art classification algorithms.

This study was further extended where we provided an in-depth analysis of collusive users based on features from user profile, timeline and network involved in both freemium and premium blackmarket services and classified users as premium customers, freemium customers and genuine users [5]. In the current work, our methodology differs from both [3] and [5] in the following ways:

- We create task-specific representations of users using their content and network level features.
- We demonstrate the utility of these representations for detecting collusive users.
- With respect to [3], we extend our dataset of collusive and genuine users from 753 and $1,000$ users to $1,807$ and $2,706$ users, respectively.
- We create a user representation using the features in [3] and combine it with other representations to improve the overall performance.

5.3 Twitter and blackmarket services

5.3.1 What is Twitter appraisal?

There are various kinds of content shared everyday on micro-blogging sites such as Twitter. The content can range from personal opinions, publicity of products, quotes, facts and simple online content promotion. There are two main ways to get more attention to such tweets:

- The user who posts the tweets takes several measures to make their tweet look attractive. They add emoticons, images, hashtags, links and mention other users. By doing so, their tweet is displayed on the timeline of other people who might follow this particular hashtag or people who follow the account that was mentioned in the tweet. We call this approach the **Primary Method** of obtaining Twitter appraisal.
- Each tweet on Twitter can also gain attention with the help of other Twitter users in the community. These users have an option to like or retweet the given tweet, or also follow the tweet creator for future updates on similar tweets. Thus, this tweet appears on the feed of the users who are connected to these users. This creates a cascading affect and is called the **Secondary Method** of obtaining Twitter appraisal.

5.3.2 What are blackmarket services?

It is often observed that the Primary Method of obtaining Twitter appraisal tends to be tedious and cumbersome. It requires the tweet poster to curate the content that can attract the most amount of Twitter users. Thus, this is not the most efficient way of gaining popularity as it only favours content that is, in general, very popular and with a large number of of posts tweeted every millisecond, a majority of the tweets go unnoticed. Thus, most people resort to the Secondary Method. Even with the Secondary Method, there is an important hindrance. Most content that will be liked or retweeted by the other users cater to their interests. They would not want to appraise any other tweet that is not even remotely aligned to their interests. Blackmarket services, thus try to eradicate this problem by creating a community of users who can appraise any tweet, given an incentive.

The blackmarket services provide services for various OSNs e.g., Facebook (followers, likes, shares, comments), Twitter (followers, retweets, likes), Instagram (followers, likes, comments). Other than OSNs, the blackmarket services also provide service to video subscription-sharing platforms e.g., YouTube (views, subscribers, likes, comments), Vimeo (plays, followers), music-sharing platforms e.g., SoundCloud (plays, followers, likes, reposts, comments), ReverbNation (fans), business and employment-oriented platforms e.g., LinkedIn (followers, connections, endorsements) etc. Shah et al. [6] divided the blackmarket services into two types based on the model of service - *Premium* and *Freemium*.

5.3.2.1 Premium services

If a customer pays a platform to receive appraisals, the platform is said to be offering premium services. Most of these purchases are divided into tiers, that cost according to their quantity - for example, purchasing retweets happens in batches of 100, 1K, 5K, etc. Almost every platform has a different way of operating. *SocialShop*[3] asks the customers for only their Twitter Username before payment - after which they can provide the link of the tweets they want to appraise. Premium services also ensure that their customers

[3]https://www.socialshop.co/

have fully completed bios and have eye-catching profile pictures. Apart from selling packages based on the quantity of the service, some premium services like *GettwitterRetweet*[4] and *SlickSocials*[5] are also customizable to accommodate the time at which the customer wants to receive most of the appraisals. For example, if a Twitter movement is taking place, a customer can pay to get the most number of likes in their tweet at that moment so that his/her tweets can get highlighted in the movement. For content-based tweets, a customer can also choose the target audience that would be the most appropriate for them. *Twesocial* [6] provides such target-specific services where it requests the customers to enter the hashtags they want to target for a particular tweet.

Like any other business model, premium services often rope in customers by offering attractive deals. *Devumi*[7] and *SocialShop* include a 100% Money-Back Guarantee if they are unable to deliver the services that were mentioned in their packages. Some services like *tweetboost*[8] also offer a one-time free trial period to the customers so that they can avail the services initially without installment and commitments. Premium services like *buyrealmarketing*[9] also provide pre-paid subscription plans to customers. In their platform, a user can opt for a single-payment, monthly subscription, three months prepaid or even six months prepaid subscription. As much as the question of the legality of such services is thought upon, platforms like *Devumi* mention in their FAQ page that they only boost the social media “presence" of an account; so it is completely legal. One can also set a daily limit on the number of appraisals one needs on his/her account. *Socialmediadaily*[10] can also accommodate more than one tweet for appraisal if a higher costing package is bought from the service.

5.3.2.2 Freemium services

Unlike Premium services that require payment from the customers, several other services work on different operating models that do not require payment. Such services are termed as Freemium Services. Some of them start with a basic free service, which gets the customers hooked – thereby motivating them to subscribe to the platforms and also publicize the platforms via word-of-mouth. A Freemium service functions by creating a community of customers. Thus, each member of the community avails the facilities by allowing other members to promote the content. This implies that once a user is a part of a Freemium service, he/she is a customer as well as a service provider, with the service merely as an interface for such communication.

Most of the freemium services operate by having a dashboard, called the ‘Earning Area’, where the customers can view the content of other customers. If the customers appraises the other customers – by liking their content, retweeting it, or following them, they earn credits. These credits are utilized when the customers would want to get appraisals for their own tweets via the service. Some of the freemium services (e.g., Like4Like [11], YouLikeHits [12]) also have a referral system for gaining credits, i.e., the customer gets points when someone joins the service using their referral code. One can also purchase credits from the freemium services using online payment systems. Freemium services (e.g., Like4Like, TraffUp [13]) also provide daily bonus points for staying active on their website.

[4] https://www.gettwitterretweet.com/
[5] https://slicksocials.com/
[6] https://www.twesocial.com/
[7] https://devumi.com
[8] http://tweetboost.net/
[9] https://www.buyrealmarketing.com/
[10] https://www.socialmediadaily.com/
[11] https://www.like4like.org/
[12] https://www.youlikehits.com/
[13] https://traffup.net/

Freemium services can be divided into three categories:

C1. **Social-share services**: Social-share services ask the customers to appraise the content of other customers on social media. Often, it is by content-matching (like hashtag matching, topical similarity of tweets) of one customer with another. This is a rather slow process as it is not guaranteed that the other customers would appraise the content.

C2. **Credit-based services**: In credit-based services, customer gain credits to appraise other customers' content, which can then be utilized in a similar fashion for their own content. This creates some sort of a give-and-take relationship, which is a very fast and effective model.

C3. **Auto-retweet services**: In auto-retweet services, the services create several accounts internally. A customer is supposed to use an access token from Twitter to login to the service. Only limited service is provided in these platforms, which is again limited by a fixed time window.

Since the most effective and popular service model followed by freemium service is credit-based services, our analysis will be based on data taken from such services.

5.3.3 Why Twitter?

Even though blackmarket services operate on several social media platforms, we chose Twitter because of several reasons. Unlike Facebook and other social media platforms that create exclusive communities, Twitter is dynamic and an open platform. Most of the tweets can be viewed by others, most of the profiles are public and almost all of the blackmarket services offer a variety of services like retweets, likes, follows, etc., that provide a very diverse scenario to study. Also, data collection from Twitter is less cumbersome as compared to other platforms.

5.4 Dataset description

We conduct our experiments on a dataset comprising of collusive users collected from the blackmarket services, as well as genuine users.

5.4.1 Dataset collection

Collecting blackmarket customers: For the purpose of collecting data for blackmarket customers, we restricted ourselves to credit-based services. We did this for the following reasons: (i) the simple model of credit-based services makes them easier to understand and hence makes it easier to collect features. The idea is simple, in order to get more retweets a customer would have to retweet tweets of other customers, thus intermittently displaying explicit anomalous retweeting behaviour mixed in with normal behavior. (ii) Credit-based services are more prevalent on most of the blackmarket sites as opposed to other types of services. This is because for availing these services, customers do not need to pay the service providers any money. All that is required is simple manual labor of retweeting the tweets of other customers. Thus, these services are more prevalent and highly popular on various blackmarket service providers. Thus, focusing on credit-based services allowed us to obtain a larger dataset of blackmarket customers for analysis. In our previous work [3], we collected an initial dataset, detailed in upper half of Table 5.2. To collect this dataset, we adopted the "active probing" strategy. Using multiple dummy accounts, we

Table 5.2: Statistics of the initial and extended datasets.

	User Type	# initial users	# suspended/deleted	# verified	# users taken for analysis
Initial Dataset [3]	Collusive Users	1102	349	31	753
	Genuine Users	1000	0	51	1000
Extended Dataset	Collusive Users	1854	47	72	1807
	Genuine Users	2721	12	550	2706

enrolled ourselves into these blackmarket services (after careful IRB approval). Then we kept retweeting tweets that were displayed to us. Every time a tweet on the service was retweeted, the Twitter User ID of the creator of the tweet and the Tweet ID of the retweeted tweet were recorded. These became our dataset of blackmarket customers. For our initial work [3], we collected data in this fashion over a period of ~3 months. The upper half of Table 5.2 summarises the statistics of this dataset. For this study, we continued the same process of data collection over a period of another ~6 months. Lower half of Table 5.2 shows the statistics of the extended dataset that we obtained over this period.

Collecting genuine users: First, we looked into the users whose tweets had been retweeted by the collusive users that we had extracted. From these users, we extracted users that had been marked as verified users by Twitter. To create our genuine user set, we took the followees of these verified users with the assumption that verified users are more likely to follow genuine users. Then, we removed users with more than 100,000 followers since high follower count may resemble a celebrity, which we wanted to discard from our analysis to avoid any unnecessary bias. Further, we removed users with fewer than 50 followers or tweets as these users do not contribute much to our large scale analysis. Finally, the human annotators (who also annotated the collusive users as discussed in Section 5.4.2) were also asked to verify whether these users seem genuine by examining their profile and timeline information. We use this genuine user set for our experiments.

Further, we scraped the profile-centric information and the timeline of each user in our dataset using the Tweepy [14] library. Table 5.2 shows the statistics of the dataset. In this work, we conducted all our experiments on the larger (extended) dataset of collusive and genuine users. As mentioned before, this extended dataset was collected in the same manner as our previous work [3], but over a longer period of time (6 months). In our extended dataset, we collected 1854 collusive users and 2721 genuine users. Out of these users, 47 collusive users and 12 genuine users turned out to have suspended/deleted accounts. We removed these users from their respective user set. We also found 3 genuine users in our collusive user set, which we removed from our genuine set. Strangely, we found that ~4% collusive users have been marked as verified users by Twitter, which clearly shows that even Twitter is not yet efficient in detecting collusive users. All the experiments in this study have been conducted on this new, extended dataset.

5.4.2 Human annotation of collusive users

We asked three human annotators [15] to divide the collusive users by labeling them as *Bots*, *Promotional collusive users* or *Normal collusive users*. Annotators were properly given the definition of each type of collusive user along with Twitter's Terms of Service. Annotators were also given complete freedom to search for any information associated with the collusive users and apply their own intuition. The guidelines given to the annotators to label a user into any of the three types of collusive users are as follows:

[14] http://www.tweepy.org/

[15] Annotators were experts in OSNs with age range between 25-35.

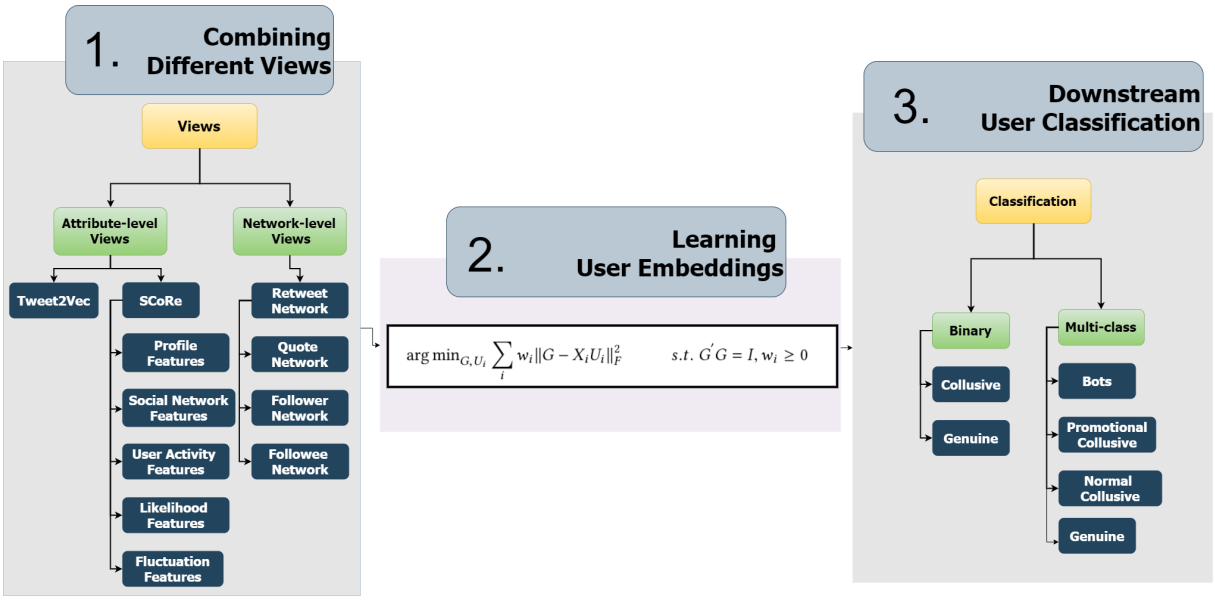


Figure 5.1: Brief overview of our methodology - divided into different steps for ease of understanding and reproducibility.

C1. **Bots:** A Twitter bot is a software which is used to access Twitter using the Twitter API. It has the capability to perform all kind of actions such as retweeting, following, liking, etc. The rules that should be followed by a Twitter Bot are mentioned in Twitter Automation rules [16].

C2. **Promotional collusive users:** Users/organizations who join Twitter for promotions, big product launches, etc. would want to explore some shortcuts to gain reputation. These promotional collusive users normally use some specific set of keywords such as 'win', 'ad', 'giveaway', 'free', 'boost', etc.

C3. **Normal collusive users:** We consider all the collusive users who do not fall under the above two categories as normal collusive users.

We found high inter-annotator agreement (Fleiss' $\kappa = 0.76$). We considered the final label for a collusive user to be the one which is marked the same by at least two annotators. For the users whose labels are not agreed upon by at least two annotators, we labeled them as normal collusive users.

5.5 Methodology

Fig. 5.1 shows a pictorial depiction of our methodology. Six different user representations are taken as distinct views, which are combined into a single user embedding in an unsupervised manner using Weighted Generalized Canonical Correlation Analysis (WGCCA). Finally, we use the user embedding to classify users in a supervised manner.

[16]https://help.twitter.com/en/rules-and-policies/twitter-automation

5.5.1 Creating views for Twitter users

Multiple studies implemented various design alternatives for user modeling strategies. Abel et al. [144] used semantic entity and topic-based user modeling strategies of Twitter users and showed advantages over hashtag-based user modeling strategies. Xu et al. [145] modeled the user posting behavior on social media. They proposed a mixture latent topic model to combine multiple behavioral properties of a user. Ritter et al. [146] proposed an unsupervised approach to model conversations of Twitter users. However, some recent papers model users from a different aspect of user social participation. Benton et al. [147] considered a multiview approach by learning vector representations of social media users from multiple views created by capturing information from user's online activities. They also proposed an approach using Weighted Generalized Canonical Correlation Analysis (WGCCA) to combine multiple views to generate a single embedding for a user. Ding et al. [148] used multiview learning to combine heterogeneous information (likes, status updates) for representation of a user.

In this work, we use views corresponding to the profile attributes and the network of a user, and generate a combined representation based on these views. We have multiple sources of deriving actionable information in our Twitter dataset: the content (tweets) posted by a user, attribute features like number of followers, number of tweets per day, number of retweets per tweet, etc. and the user's network of interactions, as indicated by their followers, followees, retweets, and quotes. We use these sources of information to generate six different views, which are divided into the following two categories:

5.5.1.1 Attribute-level views

We use the content posted by users and their attributes to get the attribute-level views.
1. `Tweet2Vec` *(AV_1)*: We use the `Tweet2Vec` model [149] to find vector-space representations of all the tweets of a user. `Tweet2Vec` is a character-level encoder for social media posts trained using the associated hashtags. It considers the assumption that posts with the same hashtags should have similar embeddings. It uses a bi-directional Gated Recurrent Unit (Bi-GRU) for learning the tweet representations. To get embeddings for a particular tweet, the model combines the final GRU states by going through a forward and backward pass over the entire sequence. We use the pre-trained model provided, trained on a dataset of 2 million tweets, to get the tweet embeddings. In our case, we take the mean of embeddings of all the tweets of a user to get a representation of the type of content the user tweets about.

2. `SCoRe` *(AV_2)*: We use various user attributes to create a feature vector. We use a set of 64 features proposed by [3] and used further in [5], that can help distinguish the different types of users. These features can be divided into five categories:

- *Profile Features (PF):* We first incorporate the profile features of a Twitter account. Here, our hypothesis is that an account with higher age tends to retweet more as compared to newly created accounts. We consider the following profile features:
  – (PF_1) *Account age* : Time (in second) that has elapsed since the creation of the Twitter account till the date when the data was collected.
  – (PF_2) *Screen name length*: String length of the name displayed on the public profile of the user on Twitter.
  – (PF_3) *Profile description presence*: Binary value that indicates whether the user has a description in his/her profile or not.
  – (PF_4) *Profile description length*: String length of the profile description of the user. It is set to 0 if profile description is not present.
  – (PF_5) *Profile URL presence*: Twitter allows users to add a URL to the profile separately to display

publicly. This is a binary value that indicates whether the profile URL is present (1) or not (0).

- *Social Network Features (SNF):* We incorporate these features to consider the connectivity between users. Users involved in collusive retweeting activities follow a lot of other users as compared to the genuine users. Such tendency of these users is due to the reason that this may draw attention of other users to their profiles and eventually may follow them. We consider the following social network related features:
  – (SNF_1) *Followees count*: Count of the number of users that a user follows on Twitter.
  – (SNF_2) *Followers count*: Count of the number of followers that the user has on Twitter.
  – (SNF_3) *Followees to followers ratio*: Ratio of SNF_1 to SNF_2.
- *User Activity Features (UAF):* We hypothesize that highly active users have a higher chance of getting their tweet retweeted by a stranger. We consider the following features to validate this hypothesis:
  – (UAF_1) *Total number of tweets*: Total number of tweets that have been authored by the user since the time of account creation till the time of data collection.
  – (UAF_2) *Number of direct mentions per tweet*: Average number of mentions per tweet.
  – (UAF_3) *Number of URLs per tweet*: Average number of URLs mentioned by the author per tweet..
  – (UAF_4) *Number of hashtags per tweet*: Average number of hashtags per tweet.
  – (UAF_5) *Number of tweets per day*: Count of the number of users that a user follows on Twitter.
  – (UAF_6) *Number of retweets per day*: Average number of tweets that a user retweets in a day.
  – (UAF_7) *Number of retweets per tweet*: Average number of retweets that a user has received on one tweet.
  – (UAF_8) *Bot-score*: We also consider `BotoM` score developed by Davis et al. [57] as one of the *UAF* features, which gives us an indication whether an account is operated by human or machine.
- *Likelihood Features (LF):* Collusive users demonstrate a mix of organic and inorganic behavior. When they submit tweets to blackmarket services, they start aggressively retweeting others' tweets on these services to gain credits. On the other hand, they do not follow such aggressiveness when they publish a tweet not submitted to any of the blackmarket services. We capture this behavior through the following features:
  – ($LF_{1\text{-}7}$) *Tweeting likelihood per day for seven days* (*Monday-Sunday*): Ratio of the tweets of a user per day to the total number of tweets the user posted in a week.
  – ($LF_{8\text{-}14}$) *Retweeting likelihood per day for seven days* (*Monday-Sunday*): Ratio of the retweets of a user per day to the total number of retweets the user performed in a day of a week.
  – ($LF_{15\text{-}21}$) *Regularity of tweeting activity per day for seven days* (*Monday-Sunday*): Regularity of tweeting activity per day is calculated by entropy, $-\sum_{i=1}^{24} p(x_i) \log p(x_i)$, where $p(x_i)$ is the fraction of tweets posted by the user at i^{th} hour of that day.
  – ($LF_{22\text{-}28}$) *Regularity of retweeting activity per day for seven days* (*Monday-Sunday*): Regularity of retweeting activity per day is calculated by entropy, $-\sum_{i=1}^{24} p(x_i) \log p(x_i)$, where $p(x_i)$ is the fraction of retweets posted by the user at i^{th} hour of that day.
  – (LF_{29}) *Tweet steadiness*: Tweet steadiness is defined as $1/\sigma_t$ where σ_t is the standard deviation of time difference between consecutive user-generated tweets.
  – (LF_{30}) *Retweet steadiness*: Retweet steadiness is defined as $1/\sigma_{rt}$ where σ_{rt} is the standard deviation of time difference between consecutive user-generated retweets.
  – ($LF_{31\text{-}37}$) *Maximum tweet likelihood per day for seven days* (*Monday-Sunday*): It is the ratio of per-day tweet count of a user to the maximum number of tweets the user posted in a day of a week.
  – ($LF_{38\text{-}44}$) *Maximum retweet likelihood per day for seven days* (*Monday-Sunday*): It is the ratio of per-day retweet count of a user to the maximum number of retweets the user posted in a day of a week.
- *Fluctuation Features (FF):* Collusive users of credit-based freemium services tend to show erratic behavior while retweeting as their only goal is to gain credits. We consider the following features to

capture this behavior:
– (FF_1) *Retweet count standard deviation*: It is the standard deviation of retweet counts for all user-generated tweets.
– (FF_2) *Retweets log-time difference average*: It is the mean of log-time difference between consecutive retweets.
– (FF_3) *Retweets log-time difference standard deviation*: It is the standard deviation of log-time difference between consecutive retweets.

The importance of each of these features has been studied in our previous work [3].

5.5.1.2 Network-level views

A user's network is the network of different type of interactions, like Retweet, Follow, Reply, etc. with other users. Each user is considered as a node and a unidirectional edge between two users indicates presence of the interaction of the given type (follow, retweet, etc.) between them, with the edge weight indicating the count of the number of interactions of the given type between them. We encode a user's network in a vector representation by considering the interaction between the users in our dataset. We create the representation by constructing an adjacency matrix of the network and then considering each row that corresponds to a user as a representation of the user's network. This gives us a vector of size $n = 4,010$ (i.e., total number of users considered here), to which we apply Principal component analysis (PCA) to get our final representation of size $n_{PCA} = 1,000$. Applying PCA helps us in capturing the user's network as a dense vector, and improves the computational efficiency of weighted generalized canonical correlation analysis (WGCCA) used later in Section 5.5.2 to combine the representations. We construct these network representations with the motivation that similar users may have similar interaction networks. We define multiple types of network views based on the type of interaction between users.

- *Retweet network (NV_1)*: In this network, an edge between two user nodes represents the number of times the first user has retweeted some tweets of the second user.
- *Quote network (NV_2)*: In this network, an edge between two user nodes represents the number of times the first user has quoted some tweets of the second user.
- *Follower network (NV_3)*: In this network, an edge between two user nodes indicates that the second user follows the first user on Twitter.
- *Followee network (NV_4)*: In this network, an edge between two user nodes indicates that the second user is a followee of the first user, i.e., the first user follows the second user on Twitter.

Fig. 5.2 shows the visualization of all the views defined in this section by applying t-SNE [150] on each view vector. Collusive and genuine users are represented by the red and green points respectively. It can be observed that each view contains some information that can be used to classify users as collusive or genuine, and that if we can effectively combine the information contained in the views, we should be able to get better performance. The importance of the views defined here can be seen in Fig. 5.3, denoted by the red bars which indicate the accuracy of the classifier achieved considering each view individually.

5.5.2 Learning multiview user embeddings

Each of the views described in Section 5.5.1 contains some information that can be used to identify collusive users. However, using just a single view may lead to missing out on valuable information that can help improve the detection performance. A naive approach of doing so would be to simply concatenate the views together. But this would give us a large user embedding and also ignore the complementary

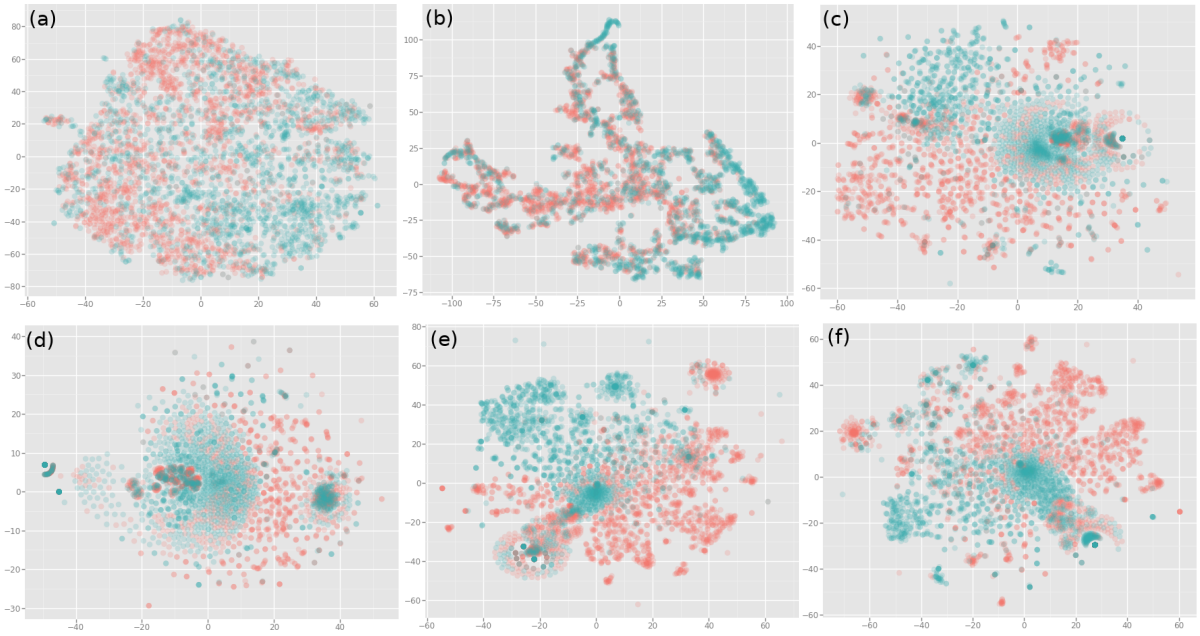


Figure 5.2: t-SNE visualization of representations of collusive (red) and genuine (green) users created using (a) `Tweet2Vec` (AV_1), (b) `SCoRe` (AV_2), (c) Retweet network (NV_1), (d) Quote network (NV_2), (e) Follower network (NV_3), (f) Followee network (NV_4).

nature of information contained in some of the views, particularly in the network views. In this section, we describe an alternate approach to learn a single embedding from multiple views using weighted generalized canonical correlation analysis.

5.5.2.1 Generalized Canonical Correlation Analysis (GCCA)

As detailed by Velden et al. [151], GCCA is a generalization of Canonical Correlation Analysis (CCA). CCA finds linear combination for two sets of variables in such a way that the correlation between them is maximal. GCCA generalizes this to more than two sets of variables. GCCA is used to analyze several set of variables simultaneously, which makes the method suitable for the analysis of various types of data containing different attributes. Several generalizations of CCA have been proposed [152, 153, 154]. However we use the one proposed by Carroll [155] for our work since it has some attractive properties that makes it well-suited for multiple-set data - a) the method is computationally straightforward since its solution is based on an eigenequation, b) it is closely related to well-known multivariate techniques like principal component analysis, and c) it takes regular CCA as a special case.

The GCCA objective can be expressed as:

$$\arg\min_{G,U_i} \sum_i \|G - X_i U_i\|_F^2 \qquad s.t.\ G'G = I \tag{5.1}$$

where $X_i \in \mathbb{R}^{n \times d_i}$ represents the data matrix for the ith view, $U_i \in \mathbb{R}^{d_i \times k}$ maps from the latent space to observed view i, $G \in \mathbb{R}^{n \times k}$ contains the learned user embeddings, G' represents the transpose of the matrix G and F is the Frobenius (or Euclidean) norm.

5.5.2.2 Weighted Generalized Canonical Correlation Analysis (WGCCA)

As we can observe from the t-SNE visualizations of the different views (c.f. Fig. 5.2), each view may be more or less helpful for our task of detecting collusive users. Hence, we weigh each view differently instead of treating each view equally - which is referred to as WGCCA [147]. Given weight w_i for each view i, where w_i represents the importance of the i^{th} view in determining the user embedding, the learning objective changes to:

$$\arg\min_{G,U_i} \sum_i w_i \|G - X_i U_i\|_F^2 \qquad s.t.\ G^{'}G = I, w_i \geq 0 \tag{5.2}$$

The columns of G are the eigenvectors of $\sum_i w_i X_i (X_i' X_i)^{-1} X_i'$, and the solution for $U_i = (X_i' X_i) X_i G$. An identity matrix scaled by 10^{-8} for regularization is added to the per-view covariance matrices before inverting, for numerical stability. The modification suggested in [147] to WGCCA, where G is scaled by the square-root of the singular values of the data matrix to improve performance in downstream tasks, is also employed. Note that the GCCA objective remains unsupervised.

5.6 Experimental setup

We choose the same baselines for this study as our previous work [3]. In this study, we also include our previous work [3] as a baseline. From the context of this study, this is equivalent to using the second attribute-level view AV_2 as a baseline. Further, as in our previous work, we use the scores provided by these baselines to train five traditional supervised classifiers – K-Nearest Neighbors (K-NN), Logistic Regression (LR), Support Vector Machine (SVM), Naive Bayes (NB), Decision Tree (DT), three ensemble classifiers – Bagging (BG), Boosting (BO) and Random Forest (RF). We report the results of the classifier that shows the best results on these baselines. We start this section by briefly explaining the baseline methods followed by the details of our experiments.

5.6.1 Baseline methods

• **Baseline I:** `BotoM`: Davis et al. [57] proposed Botometer, a system to evaluate social bots on Twitter. It computes a bot-likelihood score for a Twitter user calculated from the account's recent activity. `BotoM` service is also publicly available via REST APIs [17]. We use the bot-likelihood score provided by this baseline to train the classifiers listed above, and report the results of the classifier that gives us the best results.
• **Baseline II:** `FakeAcc`: Elazab et al. [58] attempted to detect fake accounts in Twitter using a effective minimum weighted feature set derived from a set of 22 features. They applied Gain measure [156] to find the weight for the attributes selected in the final feature set. The final feature set proposed by them was used to train the set of classifiers mentioned earlier, and the results of the classifier that gave the best precision were reported.
• **Baseline III:** `SpamBot`: We use the method proposed by Wang et al. [84] as our third baseline. It takes into account the graph-based features and content-based features extracted from the user's social graph and most recent tweets respectively to distinguish the spam bots from the genuine users. We run their suggested classifier (Naive Bayes) on our dataset for multi-class and binary classification.
• **Baseline IV:** `NDSync`: Giatsoglou et al. [1] proposed `NDSync`, a method to detect synchronous retweet

[17] https://botometer.iuni.iu.edu/

fraudsters by computing a user-level suspiciousness score by combining the suspiciousness score for each retweet thread projected into a multi-dimensional feature space. We use the suspiciousness score returned by `NDSync` as a feature for training the set of classifiers mentioned above. Once again, the results of the classifier that gave the best performance were reported.
• **Baseline V:** `SCoRe`: We also consider our previous work `SCoRe` [3] as our fifth baseline. We reported that SVM is the best state-of-the-art supervised classifier on the set of 64 features.

5.6.2 Weight assignment for WGCCA

We tune the weights assigned to the views for both our tasks of multi-class and binary classification by generating user embeddings of different sizes $\in \{50, 100, 200\}$ for all possible weight assignments, and choose the one that performs the best. The set of assignments, $\mathcal{A}$ is given by: $\mathcal{A} = \mathcal{W}^{|\mathcal{V}|} - \mathcal{A}_0$, where $\mathcal{W} \in \{0, 0.25, 1\}$ is the set of possible weights, $\mathcal{V} = \{AV_1, AV_2, NV_1, NV_2, NV_3, NV_4\}$ is the set of views, and $\mathcal{A}_0$ is the assignment $(0, 0, 0, 0, 0, 0)$, where all the views are assigned a weight of 0.

For each tuple in $\mathcal{A}$, we assign weight w_i to each view (corresponding to the elements of the tuples). We choose the best assignment of weights from the set $\mathcal{A}$ based on the result of the best performing classifier, from a set of state-of-the-art classifiers, trained on the embeddings generated for each assignment. The best performing assignment of weights is discussed in the following section.

5.6.3 Alternate view fusion methods

Apart from using WGCCA to generate a combined user embedding, we use alternate methods for view fusion (VF) to compare their performance against WGCCA. We concatenate the views together to get a combined user representation of size $n = 4564$, and use two different methods to generate a combined user representation:

• **Neural network (VF_{NN})** Post concatenation, we feed the representation to a neural network with a hidden layer of size $n = 100$ and use it to classify the users.

• **Principal component analysis (VF_{PCA})** Post concatenation, we apply PCA to get a final representation of size $n = 100$. This representation is then used to train the different classifiers listed above, and the results of the classifier with the best performance are reported.

5.7 Experimental results

We measure the performance of the competing methods using the following metrics: Precision, Recall, F1-score, and Area under the ROC curve (AUC). We report all the above metrics in both Micro and Macro settings. All the supervised baselines scores are reported after 5-fold cross validation. Note that the test set remains same across all the methods.

We design the first experiment by considering it as a multi-class (bot, promotional, normal and genuine) classification problem. We also conduct another experiment by considering it as a binary classification problem, by combining all types of collusive users (bots, promotional and normal) into a single class (customer). We run our experiments by training the following set of classifiers: Support Vector Machine (SVM), Logistic Regression (LR), Decision Tree (DT), Naive Bayes (NB) and K-Nearest Neighbors (K-NN). We also use three ensemble classifiers: Random Forest (RF), Bagging (BG) and

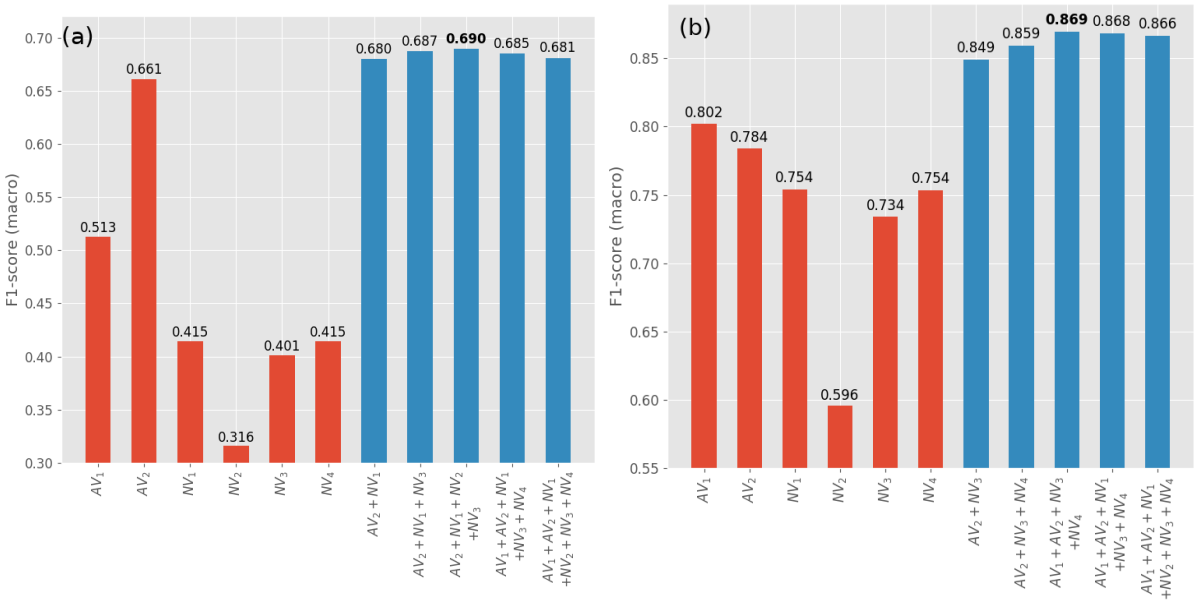


Figure 5.3: Best F1-score (macro) obtained across embeddings of size $\in \{50, 100, 200\}$, taking different combinations of views (blue) as well as taking the views individually (red) for: (a) multi-class classification, and (b) binary classification.

Boosting (BO). We further use a neural network based classifier – Multi-layer Perceptron (MLP). We perform hyperparameter optimization to get the best results. We report the best performance achieved in terms of the evaluation metrics by sweeping across user embeddings of size $\in \{50, 100, 200\}$, considering all possible weight assignments, and report the embedding size and the classifier for which we obtain the best performance.

Fig. 5.3 shows the best F1-score (macro) obtained by sweeping across output embeddings of size $\in \{50, 100, 200\}$, and all weight assignments detailed in Section 5.6.2 - for both multi-class and binary classification. The figure shows the best performance achieved after taking different number of views at a time (views assigned a weight = 0 are essentially dropped). When considering the views individually ($WGCCA_1$), we observe that a SVM trained on the `SCoRe` view (AV_2) gives us the best result for multi-class classification, with a F1-score (macro) of 0.661, and a SVM trained on the `Tweet2Vec` view (AV_1) gives us the best result for binary classification, with a F1-score (macro) of 0.802.

As opposed to our previous study [3], where we tried to classify collusive users based on the user representation obtained from `SCoRe` (AV_2), here we have a combination of different views that can be used for classifying collusive users. This allows us to capture the multi-faceted nature of collusive users. In our previous study [3], we explored the feature importance of the different features that make up the `SCoRe` representation (AV_2). Similarly, in this study, we explore the importance of each view in classifying collusive users. To achieve this, we use WGCCA, to combine different combinations of views, and study how each combination performs in the task of classifying collusive users. When we apply WGCCA to combine the views, we observe the following:

- We find that the best result when taking two views at a time ($WGCCA_2$) is achieved by using `SCoRe` (AV_2) and Retweet network (NV_1) views for multi-class classification, with a F1-score (macro) of 0.680 (using LR classifier on embeddings of size = 200); and by combining the `SCoRe` (AV_2) and follower network (NV_3) views for binary classification, with a F1-score (macro) of 0.849 (using MLP on embeddings of size = 100).

- When we take three views at a time ($WGCCA_3$), the best result is achieved by combining the

Table 5.3: Performance of different competing methods for multi-class classification.

Classifier	Micro				Macro			
	Precision	**Recall**	**F1**	**AUC**	**Precision**	**Recall**	**F1**	**AUC**
`BotoM` (RF)	0.546	0.546	0.546	0.638	0.518	0.532	0.519	0.621
`SpamBot`	0.497	0.497	0.497	0.571	0.389	0.375	0.355	0.434
`FakeAcc` (LR)	0.562	0.562	0.562	0.635	0.398	0.320	0.298	0.499
`NDSync` (LR)	0.511	0.511	0.511	0.602	0.375	0.388	0.372	0.443
`SCoRe` (SVM)	0.637	0.637	0.637	0.893	0.668	0.665	0.660	0.880
VF_{NN}	0.571	0.571	0.571	0.802	0.542	0.540	0.541	0.788
VF_{PCA} (MLP)	0.561	0.561	0.561	0.819	0.521	0.518	0.520	0.804
$WGCCA_1$ (SVM)	0.637	0.637	0.637	**0.893**	0.668	0.665	0.660	**0.880**
$WGCCA_2$ (200, LR)	0.663	0.663	0.663	0.864	0.678	0.689	0.680	0.861
$WGCCA_3$ (50, SVM)	0.687	0.687	0.687	0.848	0.691	0.688	0.687	0.834
$WGCCA_4$ (50, SVM)	0.687	0.687	0.687	0.849	0.693	**0.691**	**0.690**	0.836
$WGCCA_5$ (50, SVM)	0.684	0.684	0.684	0.866	0.687	0.685	0.685	0.856
$WGCCA_6$ (50,SVM)	**0.692**	**0.692**	**0.692**	0.868	**0.696**	0.675	0.681	0.854

Table 5.4: Performance of different competing methods for binary classification.

Classifier	Micro				Macro			
	Precision	**Recall**	**F1**	**AUC**	**Precision**	**Recall**	**F1**	**AUC**
`BotoM` (RF)	0.683	0.683	0.683	0.780	0.685	0.682	0.680	0.780
`SpamBot`	0.682	0.682	0.682	0.695	0.699	0.661	0.645	0.695
`FakeAcc` (RF)	0.694	0.694	0.694	0.692	0.694	0.688	0.688	0.692
`NDSync` (LR)	0.573	0.573	0.573	0.671	0.383	0.293	0.269	0.671
`SCoRe` (SVM)	0.766	0.766	0.766	0.799	0.771	0.759	0.760	0.799
VF_{NN}	0.781	0.781	0.781	0.779	0.779	0.779	0.779	0.779
VF_{PCA} (MLP)	0.785	0.785	0.785	0.782	0.783	0.782	0.782	0.782
$WGCCA_1$ (SVM)	0.804	0.804	0.804	0.801	0.803	0.802	0.802	0.801
$WGCCA_2$ (100, MLP)	0.851	0.851	0.851	0.849	0.850	0.849	0.849	0.849
$WGCCA_3$ (50, MLP)	0.861	0.861	0.861	0.859	0.860	0.859	0.859	0.859
$WGCCA_4$ (50, MLP)	**0.871**	**0.871**	**0.871**	**0.869**	**0.870**	**0.869**	**0.869**	**0.869**
$WGCCA_5$ (50, MLP)	0.870	0.870	0.870	0.868	0.869	0.868	0.868	0.868
$WGCCA_6$ (50, MLP)	0.868	0.868	0.868	0.866	0.867	0.866	0.866	0.866

views `SCoRe` (AV_2), Retweet network (NV_1) and Follower network (NV_3) for multi-class classification, with a F1-score (macro) of 0.687 (using SVM on embeddings of size = 50); and by combining the views – `SCoRe` (AV_2), Follower network (NV_3) and followee network (NV_4) for binary classification, with a F1-score (macro) of 0.859 (using MLP on embeddings of size = 50).

• When we take four views at a time ($WGCCA_4$), we achieve the best result by combining the views `SCoRe` (AV_2), Retweet network (NV_1), Quote network (NV_2) and Follower network (NV_3) for multi-class classification, with a F1-score (macro) of 0.690 (using SVM on embeddings of size = 50) - **which is also the best score achieved overall for multi-class classification**. For binary classification, combining the views `Tweet2Vec` (AV_1), `SCoRe` (AV_2), Follower network (NV_3) and Followee network (NV_4) gives us the best result, with a F1-score (macro) of 0.869 (using MLP on embeddings of size = 50) - **which is also the best result achieved overall for binary classification**.

• Considering five views at a time ($WGCCA_5$), we achieve the best result by combining the views `Tweet2Vec` (AV_1), `SCoRe` (AV_2), Retweet network (NV_1), Follower network (NV_3) and Followee network (NV_4) for both multi-class and binary classification, with a F1-score (macro) of 0.685 in the case of multi-class classification (using SVM on embeddings of size = 50), and 0.868 in the case of binary classification (using MLP on embeddings of size = 50).

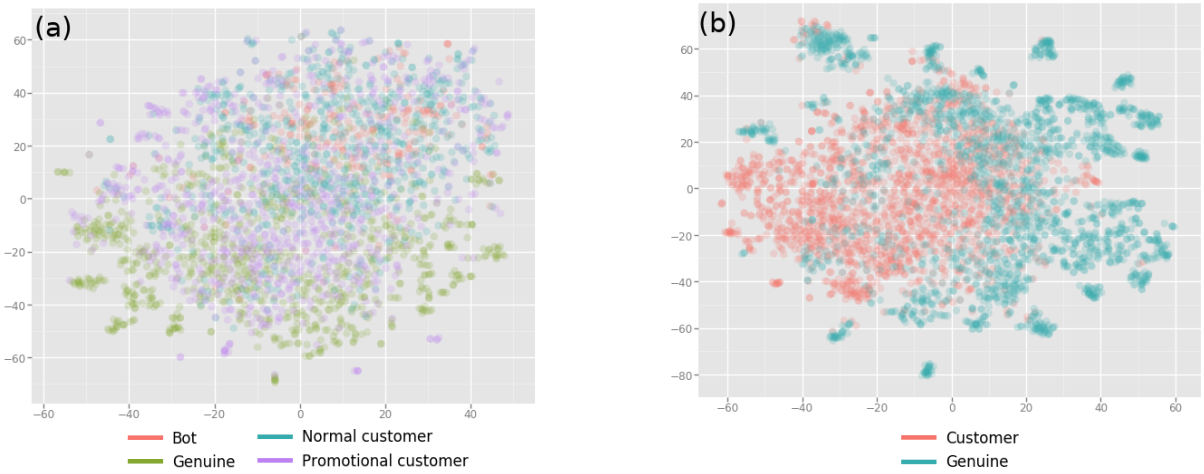


Figure 5.4: Qualitative analysis: t-SNE visualization of best performing user embeddings for (a) multi-class classification, and (b) binary classification.

• When we consider all six views ($WGCCA_6$), we achieve a F1-score (macro) of 0.681 for multi-class classification (using SVM on embeddings of size = 50) and 0.866 for binary classification (using MLP on embeddings of size = 50).

Tables 5.3 and 5.4 show the results for the multi-class classification and binary classification respectively in terms of the evaluation metrics. As shown in Table 5.3, `SCoRe` (our previous work) turns out to be the best among all the baselines. Our approach of detecting collusive users outperforms all the earlier baselines and `SCoRe`, with a relative improvement of 32.95% over the best previous baseline (`BotoM`), and 4.55% over `SCoRe`, in terms of F1-score (macro). The class-wise F1-score of the best performing classifier (SVM trained on embeddings of size = 50) in detecting each type of user is as follows: 0.74 (bot), 0.57 (promotional), 0.66 (normal), 0.64 (genuine). In case of binary classification, we observe in Table 5.4 that our approach outperforms the best baseline (`FakeAcc`) by 26.31% and `SCoRe` by 14.34%, in terms of F1-score (macro). The class-wise score of our best performing classifier (MLP trained on embeddings of size = 50) in detecting each type in case of binary classification is as follows: 0.85 (collusive) and 0.88 (genuine). Our approach of using WGCCA to generate a combined user embedding from the views also outperforms the alternate view fusion techniques - VF_{NN} and VF_{PCA}.

Fig. 5.4 shows the t-SNE visualizations of user embedding that gives us the best result for both multi-class and binary classification. The best result for multi-class classification is obtained when we use the weights: $(0, 1, 1, 0.25, 1, 0)$, corresponding to the views $(AV_1, AV_2, NV_1, NV_2, NV_3, NV_4)$, and generate combined user embeddings of size = 50, followed by training a SVM on the embeddings. The best result for binary classification is obtained when we use the weights: $(0.25, 0.25, 0, 0, 0.25, 0.25)$, corresponding to the same views, and generate combined user embeddings of size = 50, followed by training a MLP on the embeddings. We can see a separation of different types of users for both types of classifications in the figure. The visualization provides a qualitative way of evaluating our embedding method.

Further, Fig. 5.5 shows the variation in F1-score (macro) when taking different ratios of collusive users to genuine users, by training a SVM on the embedding which produced the best result for the entire dataset. This shows the robustness of our approach on skewed data (when the size of classes is imbalanced). Possible reasons for errors could be manifold - (i) the embedding method and classifier both could add noisy values in the data, or (ii) the model is unable to detect those users who retweet collectively

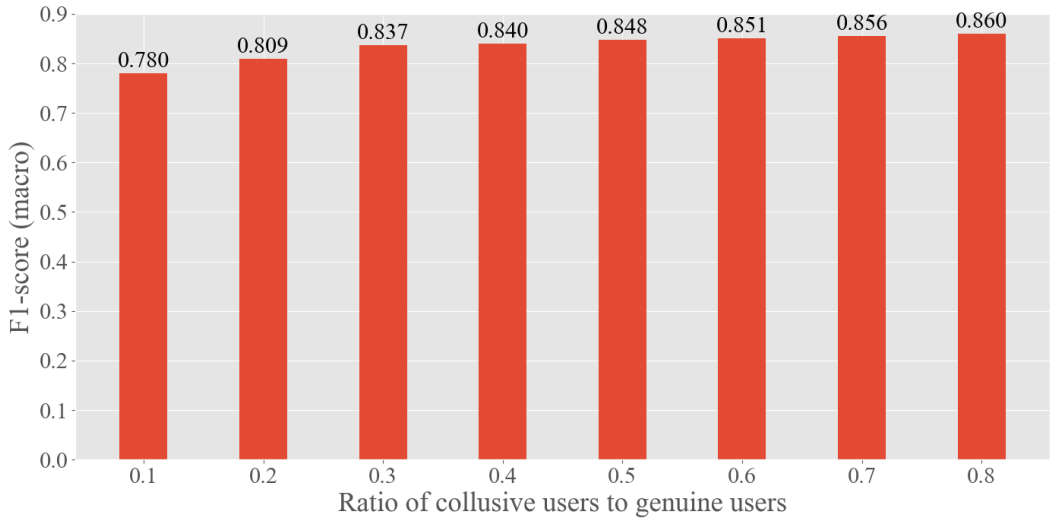


Figure 5.5: Variation of F1-score (macro) for different ratios of collusive users to genuine users.

much lesser than they retweet genuinely.

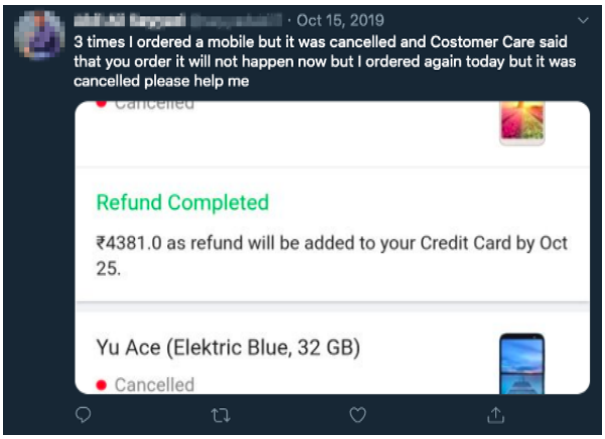
5.8 Case study

We have quantitatively established that detection and representation of collusive users require the introduction of a new methodology. We now study how effective is our method in distinguishing between collusive and genuine users.

Figure 5.6 shows screenshots of tweets made by three collusive users and a genuine user. In the very first glance, all of these tweets seem to be made by genuine users, because of a variety of reasons:

- Figure 5.6a is a tweet with a picture and a few hashtags. This seems like a genuine post; however, while analysing other tweets of this user, one can find out that they actively retweet blackmarket content (and hence have many blackmarket users in their retweet network). Thus, our multi-view approach is able to capture such behavior and flag the user as collusive, which the ground-truth annotation also supports.
- Figure 5.6b is a tweet, again with a picture, along with a complaint regarding some online service. Once again, this behaviour is extremely common by genuine users on Twitter, as they often resort to complaint resolution with the help of public support. However, this user has sent multiple of their tweets to the blackmarket service, making them an active user of the blackmarket services. They were flagged as collusive by our system.
- Figure 5.6c is a tweet about an opinion made about bitcoins based on a given link. None of the hashtags seem to be suspicious, and the use of textual enhancements like emoji's and punctuation is also in order. However, according to our ground-truth dataset, this user is an active participant of the blackmarket services and is also detected as one by our system. This is because this user actively makes use of the blackmarket services to further spread his or her own opinions, and also retweets content on blackmarket services on the topic of *blockchains*.
- Figure 5.6d is a tweet made by a genuine user. This is a promotional tweet, which gives an initial impression that it might be collusive. Although their network does not consist of collusive users and their retweet patterns do not reflect any blackmarket engagement as well. They are flagged by the system as genuine.

(a) Example tweet with a picture and a few hashtags similar to a genuine tweet.



(b) Example tweet with a picture along with a complaint regarding some online service.

(c) Example tweet about an opinion made about bitcoins based on a given link.

(d) Example tweet made by a genuine user, which looks like a promotional tweet.

Figure 5.6: Case study – Screenshots of tweets posted by collusive users ((a)-(c)) and a genuine user (d).

As we can see, collusive users can show stark similarities with genuine users. However, there are multiple factors that can mark them as collusive. These factors are mapped in our system, owing to the multi-view approach, which helps us effectively identify collusive and genuine users.

5.9 Conclusion and future work

In this work, we proposed several representations of collusive Twitter users using their attribute and network level features. We then used a multiview learning based approach (WGCCA) to combine these representations. We noticed that when we combine different user representations into a single user embedding, we are able to capture different aspects of a user's behavior on Twitter in a better way, as compared to considering the views individually. We showed the effectiveness of our approach over the baselines. The dataset we collected is also the first dataset of its kind. The dataset comprises of three types of collusive users: bots, promotional collusive users and normal collusive users, manually annotated by human annotators.

So far, we have explored the extrinsic collusive properties of users, which are analyzed directly from the multiview approach. In future, we wish to explore intrinsic properties of users and how these properties propagate to influence other users in their networks. These intrinsic properties are inherent traits in users and tweets that define their collusive behaviour, whereas extrinsic properties are their interactions with their networks. We also wish to explore the inter-dependency of such collusive users with the tweets that are a part of this network. Moreover, we would like to create a content-level

classification to study which type of tweets are submitted to blackmarket services and how these tweets are propagated. The other important direction would be to study the information diffusion of these tweets. We also plan to expand our dataset of collusive users to gain deeper knowledge about the behavioral traits of these users. The codes and datasets used in this work are available at https://github.com/uditarora/collusive-retweeters-TIST-2020.

6. Ranking to detect users involved in blackmarket-based collusive retweeting activities

Twitter's popularity has fostered the emergence of various illegal user activities – one such activity is to artificially bolster visibility of tweets by gaining large number of retweets within a short time span. The natural way to gain visibility is time-consuming. Therefore, users who want their tweets to get quick visibility try to explore shortcuts – one such shortcut is to approach the blackmarket services, and gain retweets for their own tweets by retweeting other customers' tweets. Thus the users intrinsically become a part of a collusive ecosystem controlled by these services.
In this work, we propose `CoReRank`, an unsupervised framework to detect *collusive users* (who are involved in producing artificial retweets), and *suspicious tweets* (which are submitted to the blackmarket services) simultaneously. `CoReRank` leverages the retweeting (or quoting) patterns of users, and measures two scores – the 'credibility' of a user and the 'merit' of a tweet. We propose a set of axioms to derive the interdependency between these two scores, and update them in a recursive manner. The formulation is further extended to handle the cold start problem. `CoReRank` is guaranteed to converge in a finite number of iterations and has linear time complexity. We also propose a semi-supervised version of `CoReRank` (called `CoReRank+`) which leverages a partial ground-truth labeling of users and tweets. Extensive experiments are conducted to show the superiority of `CoReRank` compared to six baselines on a novel dataset we collected and annotated. `CoReRank` beats the best unsupervised baseline method by 269% (20%) (relative) average precision and 300% (22.22%) (relative) average recall in detecting collusive (genuine) users. `CoReRank+` beats the best supervised baseline method by 33.18% AUC. `CoReRank` also detects suspicious tweets with 0.85 (0.60) average precision (recall). To our knowledge, **`CoReRank` is the first *unsupervised method* to detect collusive users and suspicious tweets *simultaneously* with *theoretical guarantees*.**

6.1 Introduction

With the engagement of a large number of users, Online Social Networks (OSNs) have become a major platform to achieve social reputation. The organic way to gain significant social reputation is a time consuming process. Therefore, users who join social media for promotion, big-product launches, etc. may not want to wait for such a long time to gain reputation; rather they may want to explore some shortcuts. This has led to the increasing trend of gaining rapid boost with the help of blackmarket services. These services provide support on all major social networks – Instagram, Facebook, Twitter, YouTube, Vimeo, Pinterest, SoundCloud, Vine etc., and assure customers that the services (followers/retweets/views/likes) they provide would appear to be genuine and extremely difficult to spot. Blackmarket services are categorized into two types based on the mode of service [6] – *premium* and *freemium*. Premium blackmarkets

provide services upon deposit of money. On the other hand, freemium services provide an additional option of unpaid services where customers themselves become a part of these services, participate in fake activities (following, retweeting other, etc.) and gain (virtual) credits. Hence, they become a part of the collusive echo-chamber controlled by these services. Here, we focus our attention on users involved in 'freemium retweeting services' and call them **"collusive users" - *users who retweet/quote tweets submitted to the blackmarket services and earn credits in return.*** We address the following problem – ***how can we design an efficient system to simultaneously detect users (based on their unusual retweeting pattern) and tweets (based on the credibility of the users who retweet them) involved in collusive blackmarket services?***

Background and Motivation: Current state-of-the-art algorithms mostly focus on detecting bots [57, 84] and users involved in synchronous fraudulent activities [1]. In our previous method `SCoRe`[3], we empirically showed that existing fake detection methods fail to identify collusive users as these users are not bots and their behavior is not synchronous. We conducted experiments only on 743 collusive retweeters collected from various freemium blackmarket services, and used standard supervised methods to classify collusive retweeters and genuine users. We also mentioned that detection of collusive users is challenging for two reasons – (i) These users neither resemble bots nor fake users as they express *a mix of organic and inorganic behavior*. Therefore, bot detection and fake user detection algorithms can not detect them. (ii) Collecting large scale labeled data of collusion users is extremely challenging. This necessitates the design of an unsupervised approach to detect collusive users.

Proposed Approach: In this work, we propose `CoReRank`, an unsupervised approach to simultaneously detect collusive users and suspicious tweets. We model the interactions between users and tweets in terms of retweets/quotes (collectively called as 'support') using a directed bipartite graph. We capture the interdependency between the *credibility of users* and the *merit of tweets* via a set of axioms. We further design recurrence formulations combining the graph, behavioral information and topical diversity of supported tweets to obtain the final score of the 'credibility' of users and the 'merit' of tweets. The cold start problem is also handled using Laplace smoothing. We further design `CoReRank+` which extends `CoReRank` to a semi-supervised setting (when few labeled data are available). We theoretically show that `CoReRank` is guaranteed to converge in a finite number of iterations, and it scales linearly with the number of edges present in the graph.

Experimental Results: We started by collecting users and tweets from two blackmarket services, and used them as seeds to expand the data to $10K$ users and 2.5 million tweets. Human annotators were employed to label unknown users as collusive/genuine. **This, to our knowledge, is the first labeled dataset of collusive/genuine users based on their retweeting activities**. We compare `CoReRank` with `SCoRe` [3], the only available system to detect collusive users involved in artificial retweeting activity. We also compare `CoReRank` with five other baselines designed to address similar type of problems (fake/bot detection, etc.). `CoReRank` outperforms the best unsupervised baseline by 269% and 300% for collusive user detection, and 20% and 22.22% for genuine user detection in terms of average precision and average recall (relative) respectively. `CoReRank+` also beats the best supervised method by 33.18% (relative) AUC. The added benefit of `CoReRank` is that it also identifies suspicious tweets with 0.85 precision and 0.60 recall. Empirical results further show that – (i) graph-based interdependency is the most effective component of our models, (ii) the running time of `CoReRank` is linear in the number of edges and much faster than any baseline.

Contribution: The overall contribution is six-fold:

- **Problem definition:** This work is addresses the problem of detecting blackmarket users involved in collusive retweeting activities.
- **Algorithm:** `CoReRank` is the first *unsupervised approach* to simultaneously detect collusive users as

well as suspicious tweets (submitted to the blackmarket services to gain retweets).

- **Theoretical guarantee:** `CoReRank` is guaranteed to converge in a finite number of iterations.
- **Effectiveness:** `CoReRank` outperforms six state-of-the-art methods by a significant margin.
- **Scalability:** `CoReRank` is fast, and scales linearly with the number of edges present in the constructed graph.
- **Dataset:** We collected and annotated a novel dataset of collusive users, which is the first labeled dataset of this kind.

Reproducibility: The code of `CoReRank` and the dataset are available at https://github.com/LCS2-IIITD/CoReRank-WSDM-2019.

6.2 Related work

We organize our related work into two parts: (i) general study on fraud detection in OSNs, and (ii) study on understanding blackmarket activities.

Fraud Detection in OSNs: A series of works investigated fraud detection from various aspects. [101] proposed a honeypot-based approach to detect spammers on Twitter and MySpace. They used profile-based features of the spammers to design supervised classifiers. [60] also used supervised classifiers with features based on tweet content and user social behavior. [157] detected criminal profiles on Twitter. A large number of papers designed methods to detect bots on Twitter. [85] analyzed the behavior of humans, bots, and cyborgs. They reported that humans have complex timing behavior while bots and cyborgs have a periodic behavior. [57] and [84] designed unsupervised and supervised approaches respectively to detect bots. [87, 134] focused on the detection of spam tweets based on tweet-related features. [158] studied fraud favoritism on retweets. Some other studies of fraud detection based on tweet content are [58, 135, 159, 160]. Multiple papers focused on detecting fraud using URLs embedded in tweets [101, 133] and blacklisted URLs [70, 135, 136]. [1] proposed `NDSync` to detect synchronized fraud activities (fake retweets) on Twitter. [63] studied the influence of fraudulent and genuine retweet threads and discovered patterns ('Triangles' and 'Homogeneity') followed by fraudulent users.

Study of Blackmarket Services: Compared to the study on general fraud detection, the effort so far has been limited to investigate fraudulent activities by blackmarket services. A detailed analysis of blackmarkets with the impact on multiple social networks was studied by [74]. [6] studied the multifaceted behavior of the blackmarket agencies. [54] studied the market size and market price of multiple Twitter follower markets. [75] used supervised approach to detect accounts of blackmarket services. [92] studied the characteristics of Twitter follower merchant markets. [76] proposed a method to detect followers who provide voluntary following services to make the profit. [52] used supervised approach for fake blackmarket follower detection.

Few studies attempted to analyze blackmarkets in other platforms such as live video-streaming, online recruitment, etc. [77] analyzed six different underground forums where users are involved in selling goods and services. [94] proposed an unsupervised approach to detect bot-generated broadcasts and views for famous broadcasting platforms (YouTube and Twitch). [143] used machine learning approach and textual analysis on a dataset containing real-life job ads to detect whether given employment or job advertisement is legitimate or fraudulent.

Major Differences with Existing Approaches: `CoReRank` combines both *graph structure* and *behavioral properties* in an *unsupervised manner* and returns *ranked lists of users and tweets simultaneously* based on *collusive retweeting activities*. `CoReRank` is also *guaranteed to converge* in a finite number

Table 6.1: Comparison of CoReRank and other baseline methods w.r.t different dimensions of an algorithm.

	[57]	[84]	[58]	[1]	[3]	[161]	Our
Address collusion phenomenon					✓		✓
Consider graph information		✓				✓	✓
Consider topic information							✓
Unsupervised approach	✓			✓		✓	✓
Return ranked list of users				✓			✓
Detect both collusive users and tweets							✓
Theoretical guarantees						✓	✓

of iterations with linear time complexity. Table 6.1 summarizes a comparison of CoReRank with other existing approaches. CoReRank is the only one which matches all specifications.

6.3 Data description

We solicited tweets from various freemium blackmarket services. We searched for these services by querying search engines with keywords such as 'Free Retweets', 'Retweet my Tweet', etc. We collected data from the following services (after taking proper IRB approval) - YouLikeHits[1] and Like4Like[2]. These services provide an 'earning area' where tweets submitted by the customers of the service are displayed so that other customers can gain credits by retweeting the tweets. We designed a web-scraper to extract tweets (T_b) as well as the customers (U_c) who submitted these tweets. We created three user sets: ground-truth genuine user set $\mathcal{S}_g$, ground-truth collusive user set $\mathcal{S}_c$, and unknown user set $\mathcal{S}_u$. We collected users (U_b) who retweeted/quoted the extracted tweets. Out of these users, we found (i) 4 verified Twitter users (U_b^v), and (ii) 329 users who were also a part of U_c. We added the former to $\mathcal{S}_g$ and latter to $\mathcal{S}_c$. The remaining 7451 users were added to $\mathcal{S}_u$. We further increased $\mathcal{S}_g$ by adding the followees of the verified users (U_f^v) with the assumption that verified users are more likely to follow genuine users. We also collected the tweets (T_f^v) that were retweeted/quoted by U_f^v. Finally, we collected all retweets and quotes (max. 3200) of users from their timeline. At this stage, the size of the ground-truth sets is as follows: $|\mathcal{S}_c| = 329$, $|\mathcal{S}_g| = 2667$ and $|\mathcal{S}_u| = 7451$. Users present in $\mathcal{S}_u$ were further annotated by human experts as collusive/genuine (see Section 6.5.2). The graph constructed from the dataset is described in Section 6.4.1.2.

6.4 Proposed methodology

In this section, we explain our efforts in formulating CoReRank. (which is motivated by [3, 162]).

6.4.1 CoReRank preliminaries

We start by constructing a graph comprising users and tweets as nodes. We hypothesize that users and tweets have intrinsic traits that often demonstrate their collusive nature and their credibility. Thus, users

[1] https://www.youlikehits.com/
[2] https://like4like.com/

and tweets can be allotted scores that define these traits. The reason to operate on the graph and not individual users/tweets (as done in [3]) is that these scores are interdependent on each other and the graph as a whole.

[**User Support**] A tweet t is considered to be *supported* (by retweeting or quoting) by a user u if u either retweeted or quoted t. This is given by $S(u,t)$ as follows -

$$S(u,t) = \begin{cases} w_q & \text{if } u \text{ quoted } t \\ w_r & \text{if } u \text{ retweeted } t \\ 0 & \text{otherwise} \end{cases}$$

Here, w_r (w_q) denote the weight of the edge when u retweeted (quoted) t. The relation between w_r and w_q can be defined by: $0 < w_r \leq w_q < 1$ as a quote is essentially a retweet augmented with further text or media, allowing the tweet to have more importance or weight than a simple retweet. In our model, we set w_r and w_q to 0.5 and 0.75 respectively. However, we show in Section 6.5 how the performance of our algorithm varies by changing the values of w_r and w_q.

[**Support Graph**] A bipartite support graph $G = (U, T, E)$ is a directed bipartite graph where U indicating the set of users forms the left partition, and T indicating the tweets supported by U forms the right partition. Edge $E_{(u,t)}$ connecting user $u \in U$ and tweet $t \in T$ indicates that u supported t with the edge weight $S(u,t)$ denoting the kind of support u extends to t.

6.4.1.1 Graph construction

The graph construction is divided into four distinct steps as follows (we use the same notations mentioned in Section 6.3).

(i) The users U_b who supported tweets T_b (submitted to Blackmarkets) form the left partition, and the submitted tweets T_b form the right partition of G. U_b and T_b are then connected by directed edges $E_{(U_b,T_b)}$; $S(u,t)$ denotes the edge weight connecting $u \in U_b$ and $t \in T_b$. We call this graph as the upper half G_u of the final graph G.

(ii) The left partition of the graph is further augmented with U_b^v, the verified users in U_b and U_f^v, the followees of U_b^v. The right partition is populated by augmenting T_f^v, the tweets supported by U_f^v. U_f^v and T_f^v are connected by directed edges $E_{(U_f^v,T_f^v)}$ with weights given by $S(.,.)$. We call this as the lower half G_l of the final graph G.

(iii) The next step is to look for possible connections $E_{(U_b,T_f^v)}$ between users in G_u and tweets in G_l. If a user $u \in U_b$ supported a tweet $t \in T_f^v$, a directed edge is added from u and t with weight $S(u,t)$.

(iv) Similarly, we search for possible connections $E_{(U_f^v,T_b)}$ between users in G_l and tweets in G_u. If a user $u \in U_f^v$ supported a tweet $t \in T_b$, a directed edge is added from u and t with weight $S(u,t)$.

After performing the above operation, we obtain the final support graph $G = (U, T, E)$, where $U = \{U_b, U_f^v\}$, $T = \{T_b, T_f^v\}$ and $E = \{E_{(U_b,T_b)}, E_{(U_f^v,T_f^v)}, E_{(U_b,T_f^v)}, E_{(U_f^v,T_b)}\}$. G turns out to be a connected graph. Table 8.1 shows the statistics of G.

Table 6.2: Statistics of the bipartite support graph G.

	Left partition	Right partition	Edges
G_u	$\lvert U_b\rvert = 7784$	$\lvert T_b\rvert = 1001$	$\lvert E_{(U_b,T_b)}\rvert = 55382$
G_l	$\lvert U_f^v\rvert = 2667$	$\lvert T_f^v\rvert = 2439319$	$\lvert E_{(U_f^v,T_f^v)}\rvert = 2862793$
G_u to G_l	# nodes in $U_b = 294$	# nodes in $T_f^v = 10512$	$\lvert E_{(U_b,T_f^v)}\rvert = 44466$
G_l to G_u	# nodes in $U_f^v = 10$	# nodes in $T_b = 14$	$\lvert E_{(U_f^v,T_b)}\rvert = 96$
G	$\lvert U\rvert = 10451$	$\lvert T\rvert = 2440320$	$\lvert E\rvert = 2962737$

6.4.1.2 Intrinsic traits of users and tweets

Users and Tweets have intrinsic traits on the basis of which they can be given a score to determine the 'credibility' (for users) and 'merit' (for tweets). Let $Out(u)$ be the set of tweets user u supported, and $In(t)$ be the set of users that support t. Below we provide a detailed discussion on the interpretation and relevance of these intrinsic properties.

Credibility of Users: The credibility of a user is an indication of how likely they are to support a tweet based on their genuine agreement with the content of the tweet. A highly credible user would only support those tweets which are about the *topics* that they frequently support. In contrast, a user with low credibility, who might be involved in collusive activities would support any tweet that they come across on a blackmarket service. This would highly diversify their timeline w.r.t the topics of supported tweets. Figure 6.1(b) supports this hypothesis. The credibility score of a user u is given by $C(u)$, where $C(u)$ ranges from 0 (very high collusive trait) to 1 (very high credibility trait).

Merit of Tweets: The merit of a tweet is an indication of the genuine organic support of users. A meritorious tweet would be supported by more credible users, indicating that it has a genuine support in the graph and is not earning support because of its submission to a blackmarket service. In contrast, a tweet with lower merit, even though it has a higher user support overall, would still lack the support of users with high credibility. The merit of a tweet t is given by $M(t)$, where $M(t)$ ranges from 0 (highly suspicious) to 1 (highly genuine).

6.4.2 `CoReRank` properties

The credibility of user u depends majorly on the merit of the tweets $Out(u)$ that u supported. Similarly, the merit of tweet t depends majorly on the credibility of the users $In(t)$ who supported t. The axioms mentioned below capture this interdependency.

[**Inter-support time**] Given a user u, let t_1 and t_2 be two consecutive tweets that u supported at T_1 and T_2, respectively. The time difference between t_1 and t_2 is thus given by $T_1 - T_2$. We construct a set of such time differences for *all* consecutive tweets that u supported, and form a set of inter-support times for u. This set is denoted by $IST(u)$.

[**Collusive users have very less inter-support times compared to genuine users**] Freemium blackmarket services require users to get 'points' to gain retweets and other privileges. In order to keep scoring, these users keep on retweeting random tweets, which leads to very less inter-support times. Hence, the average inter-support time of a collusive user, $avg(IST(u_1))$ is very less than that of a genuine user.

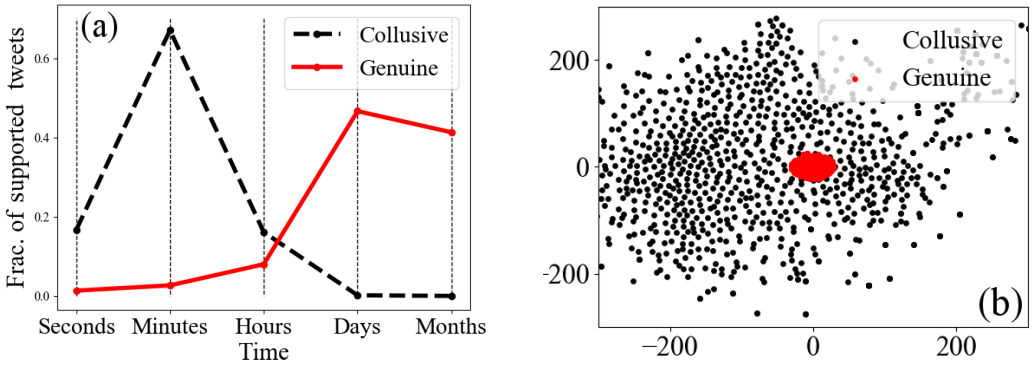


Figure 6.1: (a) Inter-support time for collusive and genuine users. Collusive users are very fast in supporting tweets. (b) Projection of tweets supported by one collusive user and one genuine user. We use t-SNE plot [2] to visualize the tweet space obtained from GloVe embedding (see Section 6.4.3.2 for details of embedding). Tweets supported by the genuine (collusive) user are on same (different) topic(s).

Formally,

$$\exists u_1, u_2 \in U, C(u_1) < C(u_2) \implies avg(IST(u_1)) < avg(IST(u_2))$$

Figure 6.1(a) validates Axiom 1. Collusive users support tweets in a succession, within a minute; whereas genuine users take days or months to support tweets.

[**Identically collusive support of tweets**] Two tweets, t_1 and t_2, are said to have identically collusive support if, $|In(t_1)| = |In(t_2)|$ and there exists a bijection f from $|In(t_1)|$ to $|In(t_2)|$ such that, $f(u) = u' \implies C(u) = C(u')\ \forall u \in In(t_1), u' \in In(t_2)$.

[**Among tweets with identically collusive support, a highly meritorious tweet receives higher support**] For two tweets, t_1 and t_2 with identically collusive support, if $S(u, t_1) \geq S(u', t_2)$, where $u \in In(t_1) \wedge u' \in In(t_2)$ such that $C(u) = C(u')$, then $M(t_1) \geq M(t_2)$.

[**A collusive user associated with blackmarket services demonstrates immense topical diversity**] As per Axiom 6.4.2, a collusive user tends to retweet every tweet that can help in receiving more credits. Thus, the tweets supported by that user tend to be very diverse in terms of the topic. Figure 6.1(b) supports Axiom 2. We show that the topics of the tweets supported by a collusive user are highly diverse (in the embedding space, points are scattered around the space), whereas the same for a genuine user is uniform (pointers are clustered).

6.4.3 `CoReRank` formulation

`CoReRank` is an unsupervised approach which considers a directed bipartite graph of users and tweets. We propose that users have unknown intrinsic scores that quantify how trustworthy they are, and tweets have unknown intrinsic scores that quantify its natural merit of being retweeted organically. Naturally, these scores are interdependent and unknown apriori. Here we start by describing how to obtain seed score which quantifies the intrinsic score of users and tweets. We also define topical diversity score of a user by calculating the inter-support similarity of the tweets s/he retweeted. Finally, we show how one can combine seed score, topical diversity and behavioral activities to jointly obtain the credibility of users and the merit of tweets.

6.4.3.1 Seed score

In order to compute the seed scores both for users and tweets that help `CoReRank` propagate further, we use `Birdnest` [161] (as suggested by [162]). `Birdnest` takes the following scores as inputs and uses Bayesian estimates to formulate how much the properties of a user or a tweet deviates from the rest.

- **User-specific Score**, is calculated by providing a vector that contains the inter-support times of the users. This array can be of variable length for each user and is provided as an input to `Birdnest`.
- **Tweet-specific Score**, is calculated with the aid of the length (word count) of the tweet. The length of a retweet (quote) is the length of the original tweet that was retweeted (the sum of the lengths of the original tweet and the quoted text). Each tweet is associated with a vector, whose each entry indicates the length of its retweet/quote. This vector is passed as an input to `Birdnest`. We hypothesize that the text length of a supported tweet is a measure of its behavioural activity [162].

The final outputs of `Birdnest` are the suspicion scores for users $S_U(u)$ and tweets $S_U(T)$. Finally, the seed scores, $\pi_U(u)$ and $\pi_T(t)$ for a user and a tweet, respectively are given by,

$$\pi_U(u) = 1 - S_U(u) \; \forall u \in U$$

$$\pi_T(t) = 1 - S_T(t) \; \forall t \in T$$

In some cases, it might happen that the user (tweet) provided (received) only a single support. In such cases, `Birdnest` does not provide us with a seed score. For these users and tweets, we have assigned the highest possible seed score of 1, giving them the benefit of doubt.

6.4.3.2 Inter-support similarity

To calculate the topical diversity score for a user, we first extract information from all the tweets present in our graph G. For each tweet of a user, we split it into words and derive their word representation in embedding space using GloVe embedding trained specifically with Tweets [163]. For this work, we use 100 dimensional pre-trained GloVe vectors. The final embedding of a tweet is obtained by combining the embedding of each word present in the tweet. We do not take into account the words for which the pre-trained word vectors are not found in GloVe. Finally, we measure $\tau_U(u)$, the inter-support similarity for user u by calculating the average cosine similarity of the embeddings corresponding to the tweets supported by u.

6.4.3.3 Recurrence formulation

We here propose a systematic approach to combine the following three signals into a recurrence framework – (i) graph-based interdependency between users and tweets, (ii) behavioral activities of users and tweet, and (iii) topical diversity of tweets supported by a user. We also present a way of handling cold start problem.

Graph-based Interdependency of Users and Tweets: As presented in the axioms mentioned previously, the credibility of a user is influenced by the merit of all the tweets that the user supported. Similarly, the merit of a tweet depends on the credibility of the users who supported it. We present below two mathematical formulations that incorporate interdependency among credibility and merit of users and tweets respectively.

For tweets,

$$M(t) = \frac{\sum_{u \in \text{In(t)}} \gamma_{1t} \cdot C(u) \cdot S(u,t)}{\gamma_{1t} + |\text{In(t)}|} \tag{6.1}$$

Similarly for users,

$$C(u) = \frac{\sum_{t \in \text{Out(u)}} \gamma_{1u} \cdot M(t) \cdot S(u,t)}{\gamma_{1u} + |\text{Out(u)}|} \tag{6.2}$$

Here, γ_{1u}, γ_{1t} are constants for users and tweets, respectively. Their values will be learned by parameter sweeping (Section 6.4.4).

Behavioral Activities: Apart from graph properties, behavioral properties are also included while calculating the credibility and merit of users and tweets. We modify Equations 6.1 and 6.2 to incorporate the seed scores explained in Section 6.4.3.1 of the users and tweets in the formulation of credibility and merit respectively.

$$M(t) = \frac{\sum_{u \in \text{In(t)}} \gamma_{1t} \cdot C(u) \cdot S(u,t) + \gamma_{2t} \cdot \pi_T(t)}{\gamma_{1t} + \gamma_{2t} + |\text{In(t)}|} \tag{6.3}$$

$$C(u) = \frac{\sum_{t \in \text{Out(u)}} \gamma_{1u} \cdot M(t) \cdot S(u,t) + \gamma_{2u} \cdot \pi_U(u)}{\gamma_{1u} + \gamma_{2u} + |\text{Out(u)}|} \tag{6.4}$$

Here, γ_{2t} and γ_{2u} are also constants for users and tweets respectively and will be learned by parameter sweeping (Section 6.4.4).

Topic-based Inter-support Similarity: Finally, we add $\tau_U(u)$, the inter-support similarity of user u into the credibility formulation. The lower the inter-support similarity, the higher the probability that the user is collusive. The modified version of Equation 6.3 is as follows:

$$C(u) = \frac{\sum_{t \in \text{Out(u)}} \gamma_{1u} \cdot M(t) \cdot S(u,t) + \gamma_{2u} \cdot \pi_U(u) + \gamma_{3u} \cdot \tau_U(u)}{\gamma_{1u} + \gamma_{2u} + \gamma_{3u} + |\text{Out(u)}|} \tag{6.5}$$

Here, γ_{3u} is a parameter for users. We learn this parameter by performing parameter sweeping as well (Section 6.4.4).

Handling Cold Start: For some users as well as tweets, multiple supports may not be available (as discussed in Section 6.4.3.1). Hence, getting seed scores for them is a problem. Also, such users and tweets can lead to biased rankings with high credibility or merit scores. We solve this problem using Laplace smoothing by assigning a default score μ_T to tweets and μ_U to users, weighed by parameters γ_{3t} and γ_{4u} respectively. The value of γ_{3t} and γ_{4u} decides how much importance is given to the default scores – high value implies less influence of the graph structure in the scores of users and tweets. Thus, the modified formulation of merit and credibility is given below.

$$M(t) = \frac{\sum_{u \in \text{In(t)}} \gamma_{1t} \cdot C(u) \cdot S(u,t) + \gamma_{2t} \cdot \pi_T(t) + \gamma_{3t} \cdot \mu_T}{\gamma_{1t} + \gamma_{2t} + \gamma_{3t} + |\text{In(t)}|} \tag{6.6}$$

$$C(u) = \frac{\sum_{t \in \text{Out(u)}} \gamma_{1u} \cdot M(t) \cdot S(u,t) + \gamma_{2u} \cdot \pi_U(u) + + \gamma_{3u} \cdot \tau_U(u) + \gamma_{4u} \cdot \mu_U}{\gamma_{1u} + \gamma_{2u} + \gamma_{3u} + \gamma_{4u} + |\text{Out(u)}|} \tag{6.7}$$

Here, γ_{3t} and γ_{4u} are parameters for tweets and users respectively, and will be learned by parameter sweeping (Section 6.4.4). The default scores are set as the average of the initial scores of all users and tweets.

Algorithm 2: `CoReRank` Algorithm

Input : $G(U, T, E), \gamma_{1t}, \gamma_{2t}, \gamma_{3t}, \gamma_{1u}, \gamma_{2u}, \gamma_{3u}, \gamma_{4u}$
Output : Credibility and Merit scores for all users and tweets

Calculate $\pi_U(u)\ \forall u \in U$ and $\pi_T(t)\ \forall t \in T$
Initialize $C(u)^0 = \pi_U(u)$ and $M(t)^0 = \pi_T(t)\ \forall u \in U, \forall t \in T$
Initialize $\mu_U = \frac{\sum_{u \in \text{U}} C(u)^0}{|U|}$ and $\mu_T = \frac{\sum_{t \in \text{T}} M(t)^0}{|T|}$
$k = 0$
error = maximum possible integer value
while $error > \epsilon$ **do**
    $k = k + 1$
    $\tilde{C}^{k-1}(u) = \texttt{norm}(C^{k-1}(u)) \forall u \in U$ such that $\tilde{C}^{k-1}(u) \in [0, 1]$
    Update the merit of tweets using Equation 6: $\forall t \in T$,
    $M^k(t) = \frac{\sum_{u \in \text{In(t)}} \gamma_{1t} \cdot \tilde{C}^{k-1}(u) \cdot S(u,t) + \gamma_{2t} \cdot \pi_T(t) + \gamma_{3t} \cdot \mu_T}{\gamma_{1t} + \gamma_{2t} + \gamma_{3t} + |\text{In(t)}|}$
    Update the credibility of users using Equation 7: $\forall u \in U$,
    $C^k(u) = \frac{\sum_{t \in \text{Out(u)}} \gamma_{1u} \cdot M^k(t) \cdot S(u,t) + \gamma_{2u} \cdot \pi_U(u) + \gamma_{3u} \cdot \tau_U(u) + \gamma_{4u} \cdot \mu_U}{\gamma_{1u} + \gamma_{2u} + \gamma_{3u} + \gamma_{4u} + |\text{Out(u)}|}$
    error = $\max(\max_{u \in U} |C^k(u) - C^{k-1}(u)|, \max_{t \in T} |M^k(t) - M^{k-1}(t)|)$
return $C^k(u)\ \forall u \in U$ and $M^k(t)\ \forall t \in T$

6.4.4 The `CoReRank` algorithm

We now briefly describe the `CoReRank` algorithm (see Algorithm 1 for the pseudo-code). Given $G(U, T, E)$, the support graph of users and tweets, `CoReRank` takes all constants – γ_{1t}, γ_{2t}, γ_{3t}, γ_{1u}, γ_{2u}, γ_{3u}, γ_{4u} as parameters. At first we calculate the seed scores for all the users and tweets, using the `Birdnest` algorithm. For users and tweets for which the seed score could not be found, it is initialized to the highest value, i.e., 1. In the first iteration, the scores for all the users and tweets are initialized to their seed scores. Next, we calculate the cold start constants for users and tweets, i.e., μ_U and μ_T respectively. At the beginning of each iteration, we normalise the credibility scores for users using *Min-Max Normalization*. Following this, we keep computing scores using Equations 6.6 and 6.7. Convergence is achieved when the maximum of the maximum of differences of scores between $(t+1)^{th}$ and t^{th} iterations, i.e., $error$ is less than a very small value, ϵ (set as 10^{-6}).

The convergence factor ϵ is chosen using a grid search in $\{10^{-6}, 10^{-4}, 0.01, 0.1, 0.5, 1\}$. We use *parameter sweep* to tune other parameters – γ_{1t}, γ_{2t}, γ_{3t}, γ_{1u}, γ_{2u}, γ_{3u} and γ_{4u}. In this method, possible combinations of all the parameters from a given list of values are considered. The possible values for γ_{1t}, γ_{2t}, γ_{3t}, γ_{1u}, γ_{2u}, γ_{4u} are taken in the range $[0, 1]$ with a step of 0.3, while γ_{3u} is chosen from $\{0, 1, 2, 3\}$. Maximum accuracy is achieved with the following parameter setting: $\gamma_{1t} = 0.6$, $\gamma_{2t} = 0.6$, $\gamma_{3t} = 0.3$, $\gamma_{1u} = 0.6$, $\gamma_{2u} = 0.6$, $\gamma_{3u} = 3$, $\gamma_{4u} = 0.3$.

Output: The outputs of `CoReRank` are ranked lists of users and tweets based on $C(u)$ and $M(t)$ respectively. Ranking in descending (ascending) order of $C(u)$ will place the genuine (collusive) users at the top of the ranking (same for the tweets based on $M(t)$).

6.4.5 `CoReRank+`: A semi-supervised version

Oftentimes, we have partial knowledge about the labels of some users (verified, blackmarket customers, etc.) and tweets. We can leverage such prior information and incorporate them to our formulation in a semi-supervised fashion. We provide each user u with a label score, $\alpha_U(u)$ which can be defined as follows -

$$\alpha_U(u) = \begin{cases} \alpha_U^c & \text{if } u \text{ is a collusive user} \\ \alpha_U^g & \text{if } u \text{ is a genuine user} \\ 0 & \text{if } u \text{ has no pre-defined label} \end{cases}$$

Similarly, the label score for a tweet t, $\alpha_T(t)$ can be defined as follows -

$$\alpha_T(t) = \begin{cases} \alpha_T^c & \text{if } t \text{ is a labeled as suspicious} \\ 0 & \text{if } t \text{ has no pre-defined label} \end{cases}$$

We set the values of these constants in such a manner that the genuine users are awarded with a high positive label score and the customers of blackmarket services are given a high negative label score. In our experiment, we set these constants as follows: $\alpha_U^c = -100$, $\alpha_U^g = 100$ and $\alpha_T^c = -100$.

The final equations of credibility and merit after incorporating the labels are given below -

$$M(t) = \frac{\sum_{u \in \text{In(t)}} \gamma_{1t} \cdot C(u) \cdot S(u,t) + \gamma_{2t} \cdot \pi_T(t) + \gamma_{3t} \cdot \mu_T + \alpha_T(t)}{\gamma_{1t} + \gamma_{2t} + \gamma_{3t} + |\text{In(t)}|} \tag{6.8}$$

$$C(u) = \frac{\sum_{t \in \text{Out(u)}} \gamma_{1u} \cdot M(t) \cdot S(u,t) + \gamma_{2u} \cdot \pi_U(u) + + \gamma_{3u} \cdot \tau_U(u) + \gamma_{4u} \cdot \mu_U + \alpha_U(u)}{\gamma_{1u} + \gamma_{2u} + \gamma_{3u} + \gamma_{4u} + |\text{Out(u)}|} \tag{6.9}$$

We design a modified algorithm, called `CoReRank+` by replacing the equations mentioned in Lines 10 and 12 of Algorithm 1 by Equations 6.8 and 6.9.

6.4.6 Theoretical guarantee

In this section, we provide the theoretical guarantee to show that `CoReRank` will converge for a given set of inputs. Before we begin, let $C^\infty(u)$ and $M^\infty(t)$ be the final scores of user u and tweet t respectively. Then we have the following results:

[**Lemma 1**] For any given tweet, t, the difference between their final score and their score after the first iteration of `CoReRank` cannot exceed $\frac{3}{4}$. Formally, it means that $|M^\infty(t) - M^1(t)| \leq \frac{3}{4}$. Similarly, for users the upper bound is $\frac{3}{4}$, i.e., $|C^\infty(u) - C^1(u)| \leq \frac{3}{4}$

[**Theorem of convergence**] Between successive iterations, the difference in the scores of the users and tweets is bounded. For any user $u \in U$, $|C^{\infty}(u) - C^{k}(u)| \leq \left(\frac{3}{4}\right)^{k}$. Thus, as the algorithm proceeds through more and more iterations, the value of k keeps on increasing and the difference in score from the final score keeps on decreasing. Similarly for a tweet $t \in T$, $|M^{\infty}(t) - M^{k}(t)| \leq \left(\frac{3}{4}\right)^{k-1}$.

Proof Sketch: We will prove this using induction on k.
Base Cases ($k = 1$): This is vacuously true from Lemma 6.4.6.
Induction Hypothesis: Assume that for any $n \leq k$, we have, $|C^{\infty}(u) - C^{k}(u)| \leq \left(\frac{3}{4}\right)^{k}$ and $|M^{\infty}(t) - M^{k}(t)| \leq \left(\frac{3}{4}\right)^{k-1}$, $\forall u \in U, \forall t \in T$.
Induction Step ($k = n + 1$): For $k = n + 1$, we have,

$$|M^{\infty}(t) - M^{n+1}(t)| \leq \gamma_{1t} \frac{\sum_{u \in \mathrm{In(t)}} |(\tilde{C}^{\infty}(u) - \tilde{C}^{n}(u))| \cdot |S(u,t)|}{\gamma_{1t} + \gamma_{2t} + \gamma_{3t} + |\mathrm{In(t)}|}$$

As $|(\tilde{C}^{\infty}(u) - \tilde{C}^{n}(u))| \leq |C^{\infty}(u) - C^{n}(u)| \leq \left(\frac{3}{4}\right)^{n}$,

$\implies |M^{\infty}(t) - M^{n+1}(t)| \leq \frac{\gamma_{1t} \cdot \left(\frac{3}{4}\right)^{n} \cdot |\mathrm{In(t)}| \cdot |S(u,t)|}{\gamma_{1t} + \gamma_{2t} + \gamma_{3t} + |\mathrm{In(t)}|} \leq \left(\frac{3}{4}\right)^{n}$

Also, $|C^{\infty}(u) - C^{n+1}(u)| \leq \frac{\sum_{t \in \mathrm{Out(u)}} |\gamma_{1u}| \cdot |(M^{\infty}(t) - M^{n+1}(t))| \cdot |S(u,t)|}{\gamma_{1u} + \gamma_{2u} + \gamma_{3u} + \gamma_{4u} + |\mathrm{Out(u)}|}$
$\leq \frac{|\gamma_{1u}| \cdot \left(\frac{3}{4}\right)^{n} \cdot |S(u,t)|}{\gamma_{1u} + \gamma_{2u} + \gamma_{3u} + \gamma_{4u} + |\mathrm{Out(u)}|}$. As $S(u,t) \leq \frac{3}{4}$, $|C^{\infty}(u) - C^{n+1}(u)|$
$\leq \frac{|\gamma_{1u}| \cdot \left(\frac{3}{4}\right)^{n+1}}{\gamma_{1u} + \gamma_{2u} + \gamma_{3u} + \gamma_{4u} + |\mathrm{Out(u)}|} \leq \left(\frac{3}{4}\right)^{n+1}$

[**Bound on iterations**] There exists a bound on the number of iterations until `CoReRank` converges. This bound is governed by the precision to which the score is calculated before convergence is declared, i.e., ϵ. The number of iterations till convergence is at most $2 + \lceil \frac{\log\left(\frac{\epsilon}{2}\right)}{\log\left(\frac{3}{4}\right)} \rceil$ **Proof:** Let $k = 2 + \lceil \frac{\log\left(\frac{\epsilon}{2}\right)}{\log\left(\frac{3}{4}\right)} \rceil$. By Theorem 6.4.6, after $k + 1$ iterations, $\forall t \in \mathcal{T}$, $|M^{\infty}(t) - M^{k+1}(t)| \leq \frac{3}{4}^{k} \leq \frac{3}{4}^{\log_{\frac{3}{4}}(\frac{\epsilon}{2})} = \frac{\epsilon}{2}$. Similarly, $M^{\infty}(t) - M^{k+2}(t)| \leq \frac{\epsilon}{2} \cdot \frac{3}{4} \leq \frac{\epsilon}{2}$. Thus,

$$|M^{k+1}(t) - M^{k+2}(t)| = |M^{k+1}(t) - M^{\infty}(t) + M^{\infty}(t) - M^{k+2}(t)|$$

As $|x + y| \leq |x| + |y|$,

$\implies |M^{k+1}(t) - M^{k+2}(t)| \leq |M^{k+1}(t) - M^{\infty}(t)| + |M^{\infty}(t) - M^{k+2}(t)| \leq 2 \cdot \frac{\epsilon}{2} = \epsilon$

Similarly for credibility, $|C^{k+1}(u) - C^{k+2}(u)| \leq \epsilon \forall u \in \mathcal{U}$.

Thus, by Line 6 of Algorithm 1 it will take $k + 2$ iterations to converge. Hence, a low value of ϵ would require a larger number of iterations for the algorithm to converge.

Time Complexity of `CoReRank`: In each iteration, `CoReRank` updates the scores of users and tweets in constant time. Thus, the complexity of each iteration is $\mathrm{O}(|\mathrm{E}| + |\mathrm{V}|)$. Since $|E| >> |V|$ in $G(U,T)$, $\mathrm{O}(|\mathrm{E}| + |\mathrm{V}|) \approx \mathrm{O}(|\mathrm{E}|)$. As explained in Theorem 6.4.6, `CoReRank` converges in a constant number of iterations. Let n be the product of the number of iterations till convergence and the number of runs of `CoReRank`. Thus, the time complexity of algorithm is $\mathrm{O}(\mathrm{n}|\mathrm{E}|)$, which is linear in the number of edges of $G(U,T)$ (as supported empirically in Figure 6.5(d)).

6.5 Experimental results

This section starts by briefly explaining the baselines and human annotation process, followed by the detailed performance analysis of the competing methods.

6.5.1 Baseline methods

As mentioned before, `CoReRank` addresses the problem of detecting collusive users involved in fake retweeting activities. We choose six state-of-the-art (supervised and unsupervised) methods as baselines, which are similar to the problem we address here. (i) **Baseline I** (`BotoM`): [57] measured bot score for each user and produced a ranked list of users. (ii) **Baseline II** (`NDSync`): [1] detected synchronous retweet fraudulent activities by calculating a user-level suspiciousness score which combines the suspiciousness score for each retweet thread by projecting them into a multi-dimensional feature space. (iii) **Baseline III** (`Birdnest`): [161] detected fraudulent reviewers by combining the rating and temporal information and generating a likelihood-based suspiciousness metric for users and reviews. We adopted `Birdnest` to rank users and tweets.

The **supervised approaches** are as follows. (i) **Baseline IV** (`SpamBot`): [84] detected spam bots using a set of content-based and graph-based features. The proposed set of features is used in Naive Bayes classifier to classify a user into collusive or genuine. (ii) **Baseline V** (`FakeAcc`): [58] used a classification method to detect fake accounts on Twitter based on minimum weighted feature set calculated over 22 features. The feature set is then applied on several classifiers among which SVM performed the best. (iii) **Baseline VI** (`SCoRe`): Our previous work [3] is the closest baseline of our method which identified blackmarket customers by running SVM using a set of 64 features.

6.5.2 Annotating unknown users

We asked three human annotators[3] to label $7,451$ unknown users $\mathcal{S}_u$ (mentioned in Section 6.3) into *collusive* or *genuine*. Annotators were given the definition of collusive users and Twitter Terms of Service. They were also given complete freedom to search for more information related to the (collusive) users on the web and apply their intuitions. The following guidelines were also given to them:

(1) Collusive users are members of blackmarket services who retweet/quote the tweets submitted to blackmarket services to earn credits in return.

(2) To earn credits from the blackmarket services, collusive users tend to show more aggressive behavior in terms of retweeting activity. Moreover, collusive users are less likely to retweet content of their friends.

(3) Generally, tweets submitted to the blackmarket services are related to promotions of tweets, accounts or services [82].

(4) Users created by the blackmarket services will only be involved in retweeting other tweets rather than publishing their own tweets. The annotators were asked to check the retweet statistics such as number of retweets, inter-retweet times, etc.

We found high inter-annotator agreement (Fleiss' kappa coefficient of 0.75). We finally considered

[3] Annotators were experts in social media, and their age ranged between 25-35.

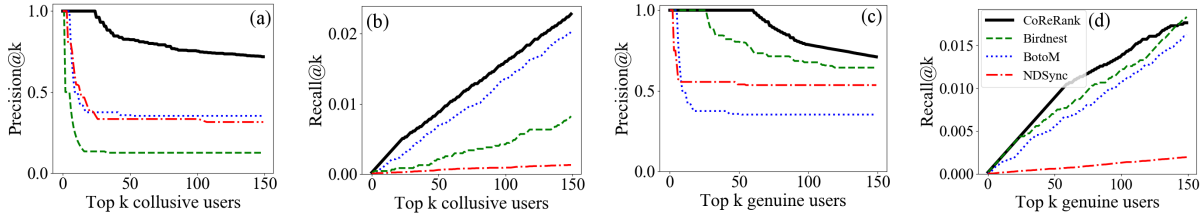


Figure 6.2: Change in performance of the competing unsupervised methods with the increase of k (the number of results returned) for detecting both (a-b) collusive and (c-d) genuine users.

4732 users as collusive and 2719 users as genuine (for which at least two annotators agreed). The newly labeled sets were augmented with the sets of ground-truth users $\mathcal{S}_c$ and $\mathcal{S}_g$ (as described in Section 6.3).

6.5.3 Comparative evaluation

We observe the performance of the competing unsupervised methods with the increase of k, number of results returned. Figure 6.2 shows that till $k = 30$ ($k = 60$), precision of `CoReRank` is almost 1 for collusive (genuine) user detection; even after this, the decrease in performance is significantly less for `CoReRank` compared to other methods. The recall curve corresponding to `CoReRank` also increases steadily with k compared to other methods.

Table 6.3 shows the performance of unsupervised and (semi-) supervised methods separately. We observe that `CoReRank` beats the best baseline (`Birdnest`) with 269% high average precision and 300% high average recall for collusive user detection. For genuine user detection, its improvement is 20% and 22.22% higher than `Birdnest` based on average precision and average recall respectively. The performance of (semi-) supervised methods is reported separately after averaging over 10-fold cross validation. For `CoReRank+`, we choose 0.202 as the threshold of $C(u)$; users whose corresponding $C(u)$ values are higher than (lower than or equal to) the threshold are considered as genuine (collusive). We plot the non-cumulative distribution of $C(u)$ and choose the threshold based on sudden change in the slope of the curve in Fig. 6.4 which we explain below. We observe that `CoReRank+` beats the best baseline (`SCoRe`) by 33.18% in terms of AUC (similar pattern is observed for other evaluation metrics).

Metric	BotoM	NDSync	Birdnest	CoReRank
AP (Collusive)	0.212	0.218	0.221	**0.817**
AR (Collusive)	0.006	0.010	0.003	**0.012**
AP (Genuine)	0.394	0.532	0.727	**0.875**
AR (Genuine)	0.005	0.009	0.009	**0.011**
Metric	**SpamBot**	**FakeAcc**	**SCoRe**	**CoReRank+**
AUC	0.573	0.578	0.696	**0.927**
P (Collusive)	0.611	0.589	0.724	**0.933**
R (Collusive)	0.336	0.362	0.727	**0.887**
P (Genuine)	0.547	0.559	0.603	**0.910**
R (Genuine)	0.790	0.762	0.599	**0.947**

Table 6.3: Performance of the competing methods. We show the accuracy separately for unsupervised (in terms of Average Precision (AP) and Average Recall (AR)) and (semi-) supervised (in terms of ROC-AUC, Precision (P) and Recall (R), averaged over 10-fold cross validation) methods.

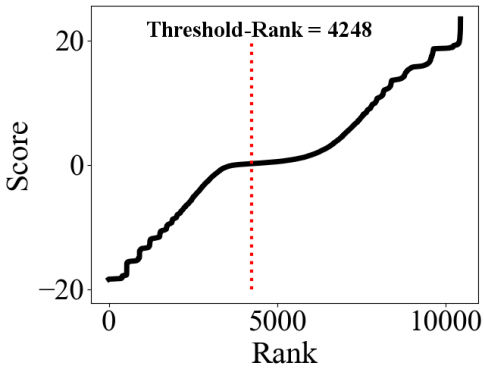


Table 6.4: Non-cumulative distribution of credibility scores vs. rank of users after algorithm

In order to evaluate `CoReRank+`, we create a thresholding technique that allows us to compare it's results with the available ground truth labels. In Figure 6.4, we plotted the non-cumulative distribution of credibility scores for all users ($C(u) \forall u \in U$) against their allotted ranks. At $x = 4248$, we encounter the **Sharpest Turning Point** of the curve. Corresponding to $x = 4248$, $C(u) = 0.202$. A threshold margin is then created at $x = 4248$. All the users whose rank lies before 4248 are termed as `collusive` and all

those whose rank is after 4248 are termed as `genuine`.

The possible reason for misclassifcation in the CoReRank graph is that edges are more genuine than their collusive retweets and that spurious signal propagates to its neighbors as well. For such a collusive user, topical diversity score would also be lower.

Robustness Analysis: We also report the performance of the competing (semi-)supervised methods with the increase of training size to show the robustness of the methods w.r.t the size of the training set. We vary the training data from 10% to 90%. Figure 6.5(c) shows the AUC on test sets, averaged over 50 random iterations of training data. We observe that `CoReRank+` always outperforms others for all training percentage. This shows the efficiency of our framework even when a small amount of training data is available.

6.5.4 Parameter selection

We conducted Parameter Selection by the method of Parameter Sweep. In Parameter Sweep, all possible combinations of the parameters are checked to find out the most optimal combination that yields the best precision.

Selecting graph parameters: The weights of the edges of the graph, w_r and w_q are taken in the range $(0, 1)$ at a step of 0.25. Also, the condition $0 < w_r \leq w_q < 1$ needs to be satisfied. Thus, all the possible combinations of (w_r, w_q) are ordered in the following fashion- $(0.25, 0.25)$, $(0.25, 0.5)$, $(0.25, 0.75)$, $(0.5, 0.5)$, $(0.5, 0.75)$ and $(0.75, 0.75)$.

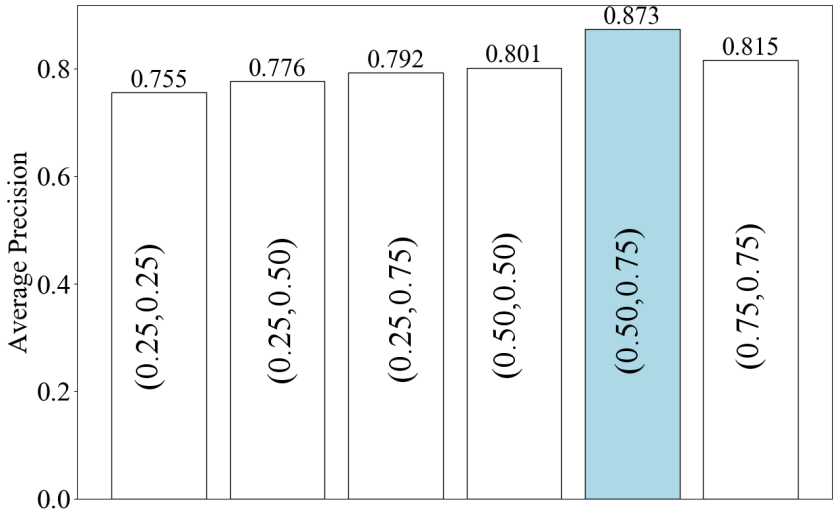


Figure 6.3: Average precision of CoReRank vs. Index of ordered combinations of edge weights

Figure 6.3 shows how the precision of the algorithm varies with the different ordered combinations of the weights of the edges of the graph. The highest precision is observed at $w_r = 0.50$ and $w_q = 0.75$, which is the optimal parameter combination fixed for our entire experiment.

Selecting recurrence parameters: CoReRank has 7 parameters - $\gamma_{1t}, \gamma_{2t}, \gamma_{3t}, \gamma_{1u}, \gamma_{2u}, \gamma_{3u}, \gamma_{4u}$. $\gamma_{1t}, \gamma_{2t}, \gamma_{3t}, \gamma_{1u}, \gamma_{2u}, \gamma_{4u}$ are taken in the range $[0, 1]$ with a step of 0.3, while γ_{3u} is chosen from $\{0, 1, 2, 3\}$.

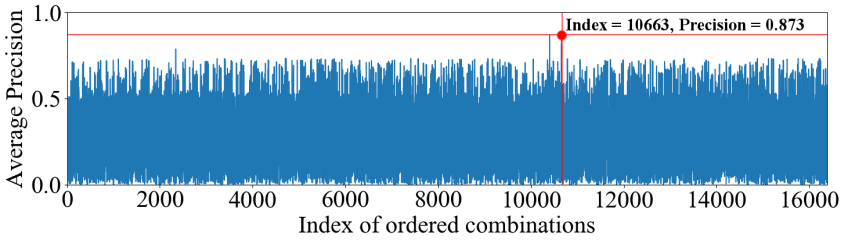


Figure 6.4: Average precision of CoReRank vs. Index of ordered combinations of parameters

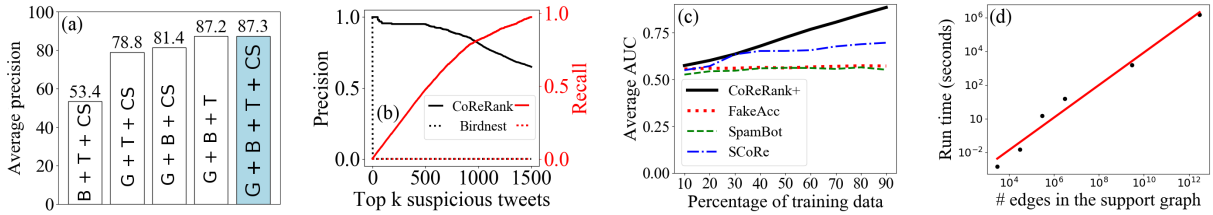


Figure 6.5: (a) Importance of different components of the recurrence formulation – graph (G), behavior (B), topic (T) and cold start (CS). (b) Precision and recall of `CoReRank` and `Birdnest` in detecting suspicious tweets. (c) Average AUC with different percentage of training data. (d) Scalability analysis of `CoReRank`.

Figure 6.4 demonstrates how the precision changes due to the affect of different combinations of the parameters. The `x-axis` denotes the combinations of the parameters. The combinations are ordered, in order to facilitate the plotting of the curve. The `y-axis` denotes the precision of the algorithm. As it can be noticed, the precision of the algorithm reaches a global maxima at index 10663. The combination that is represented by this index 10663 is selected as our optimal parameter combination throughout all the experiments. This combination is as follows - $\gamma_{1t} = 0.6$, $\gamma_{2t} = 0.6$, $\gamma_{3t} = 0.3$, $\gamma_{1u} = 0.6$, $\gamma_{2u} = 0.6$, $\gamma_{3u} = 3$ and $\gamma_{4u} = 0.3$.

6.5.5 Importance of different components

In order to understand the importance of different components in the recurrence formulation of `CoReRank`, we drop each component in isolation (set its corresponding coefficient as 0) and measure the accuracy. Figure 6.5(a) shows maximum decrease in accuracy (38%) after removing graph (G) component, followed by behavior (9%), topic (6.75%) and cold start (0.11%). However, removing none of the components increases the accuracy, indicating that all the components are important.

6.5.6 Suspicious tweet detection

One of the advantages of `CoReRank` is that apart from ranking users based on their collusive activities, it can also rank tweets based on the merit score, which none of the baselines (except `Birdnest`) can. We take 1001 tweets which we collected from the blackmarket services as ground-truth suspicious tweets[4]. Figure 6.5(b) shows the precision and recall of `CoReRank` and `Birdnest` with the number of tweets returned (k). `CoReRank` achieves 0.85 average precision and 0.60 average recall, whereas both values for `Birdnest` is 0. In the ranked list that `Birdnest` returns, the first suspicious tweet appears at 3485^{th} position.

[4]We did not show the results for genuine tweet detection as we were not sure about the ground-truth genuine tweets.

6.5.7 Scalability analysis

Theorem 6.4.6 already showed that the running time of CoReRank scales linearly in the number of edges in the bipartite support graph. To show this empirically, we take the complete bipartite graph and keep adding edges (within the range of $10^4 - 10^{12}$) randomly without changing the number of nodes. Figure 6.5(d) shows that the running time increases linearly with the number of edges. On the dataset we collected, the average running time over 50 iterations is 170 seconds, which is much faster than BotoM (14400 seconds), NDSync (3000 seconds) and Birdnest (1200 seconds)[5]. We implemented CoReRank in Python using the Pandas library [164] for efficiency. All experiments were executed on a 1.8 GHz Intel Core i5 Macbook Air, 8 GB DDR3 RAM, running macOS High Sierra v10.13.3.

6.6 Conclusion

We presented CoReRank, the first unsupervised framework to simultaneously detect collusive users and suspicious tweets, controlled by blackmarket retweeting services. We showed the effectiveness of our framework over six other baselines on our own (manually curated and annotated) dataset. We also provided theoretical guarantees to show the convergence of CoReRank. CoReRank outperforms the best unsupervised baseline by 269% and 300% for collusive user detection, and 20% and 22.22% for genuine user detection in terms of average precision and average recall (relative) respectively. CoReRank+ also beats the best supervised method by 33.18% (relative) AUC. The added benefit of CoReRank is that it also identifies suspicious tweets with 0.85 precision and 0.60 recall. Empirical results further show that – (i) graph-based interdependency is the most effective component of our models, (ii) the running time of CoReRank is linear in the number of edges and much faster than any baseline. The dataset we collected is also the first dataset of this kind. CoReRank can further be improve with tweet2vec [165] or even Transformer-based methods can be used for generating vector representation of a tweet. A recent large-scale language model BERTweet [166] can also be used; however, it is only pre-trained for English tweets. We also made the code and dataset available for the purpose of reproducibility.

[5]We did not report the running time of supervised methods as it may not be appropriate to compare two different classes of methods.

Part III

Collusion on other Twitter appraisals and online media platforms

7. Detecting collusive users involved in blackmarket following services on Twitter

The popularity of Twitter has fostered the emergence of various illegal user activities - one such activity is to artificially bolster the social reputation of Twitter profiles by gaining a large number of followers within a short time span. Many users want to gain followers to increase the visibility and reach of their profiles to wide audiences. This has provoked several blackmarket services to garner huge attention by providing illegitimate followers via the network of agreeable and compromised accounts in a collusive manner. Their activity is difficult to detect as the blackmarket services shape their behavior in such a way that users who are part of these services disguise themselves as genuine users.
In this work, we propose DECIFE, a framework to detect collusive users involved in producing 'following' activities through blackmarket services with the intention to gain collusive followers in return. We first construct a heterogeneous user-tweet-topic network to leverage the follower/followee relationships and linguistic properties of a user. The heterogeneous network is then decomposed to form four different subgraphs that capture the semantic relations between the users. An attention-based subgraph aggregation network is proposed to learn and combine the node representations from each subgraph. The combined representation is finally passed on to a hypersphere learning objective to detect collusive users. Comprehensive experiments on our curated dataset are conducted to validate the effectiveness of DECIFE by comparing it with other state-of-the-art approaches. To our knowledge, this is the first attempt to detect collusive users involved in blackmarket 'following services' on Twitter.

7.1 Introduction

Are you a Twitter user? Do you want to boost your Twitter profile (by increasing the follower count) within a limited time without getting suspended by Twitter? Several online services are ready to assist you. What you need to do is simple – pay them, and they will provide you followers; most of these followers would be legitimate Twitter users. If you can not afford to pay money, then you can opt for another option – just become a part of these services, start following their customers and earn credits; these credits can be used further to gain your own followers. Do not worry about being flagged by Twitter policy as these services are so smart in their following mechanism that they can easily deceive Twitter into thinking that their activity is legitimate. This is the philosophy behind many online blackmarket following services.

Twitter is arguably the most popular Online Social Network (OSN) for mass communication. It is also one of the key platforms for digital campaigning, social networking and opinion dissemination. The popularity of Twitter has attracted with it new online markets that help its users to make their profiles attractive by increasing followers, retweets, likes, replies, etc. In this context, Stringhini et al. [167] coined the term "Twitter followers market" to characterize those syndicates catering to people willing to pay for a

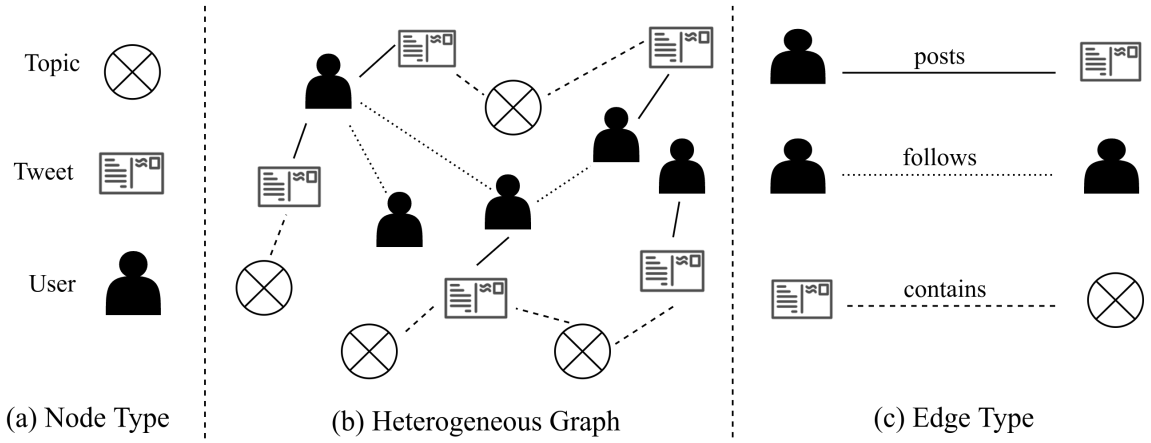


Figure 7.1: A heterogeneous network for modeling collusive users and their interactions. (a) Three types of nodes. (b) User-tweet-topic heterogeneous network. (c) Three types of edges involved in the network. The detailed construction of the heterogeneous network can be found in Section 7.4.1.

quick increase of their followers. Many people try to rapidly gain fame by exploiting this mechanism – they buy followers from these online markets. However, the coordinators of these services manage the activities of users to maintain the integrity and avoid societal, privacy and security problems. Controlling such type of illegitimate following activities has thus become one of the major challenges. There exist several blackmarket agencies which have created thriving and intelligent ecosystems of producing illicit followers. Users can gain such followers by paying money (*premium services*) or for free by following customers of those services (*freemium services*) [168]. In this work, we focus our attention on the latter case, and call this type of users "**collusive users**" – ***users who gain followers from blackmarket services***. Our previous works [3, 5, 169] on collusive entities in online media reported that collusive users are not 'fake' – they are not bots, the handles of these Twitter profiles are normal human beings, and their following activities are not mechanically controlled by any predefined policy. Rather, collusive users exhibit *a hybrid following behavior* – on one hand, similar to non-collusive users, they organically follow other users due to similar topical interest; on the other hand, similar to fake users, they inorganically follow other blackmarket customers only to gain credits. Designing a system to identify collusive users would be useful for Twitter managers and social network analysts to figure out how and what extent the profile of a user has turned out to be popular due to the support of such blackmarket services.

In this work, we propose `DECIFE`, a novel heterogeneous graph attention network for detecting collusive users who are involved in producing fake followers. We first create a heterogeneous network (c.f. Figure 7.1) based on the relationship and linguistic properties of users. We then decompose the network into four different subgraphs to capture semantic relations between users. Finally, we exploit a hierarchical aggregation network to learn node representations that are passed on to a hypersphere learning objective to detect collusive users. To evaluate our method, we collected data of collusive users from a popular blackmarket service by designing a customized web scraper. The entire data collection was carried out after taking proper IRB (institutional review board) approval from our institute. We show the effectiveness of our proposed `DECIFE` model by comparing it with three baseline methods – FakeFol$_{ss}$[170], FakeFol$_{s}$[170], FolMarket [171] and three individual components of `DECIFE` considered in isolation (ablation study).

In summary, the major contributions of the work are four-fold:

C1. We deal with a **novel problem** of detecting collusive users in Twitter where normal users get involved in illegal following activities through blackmarket services for boosting their online profiles. *To the best of our knowledge, no existing work has investigated the problem of 'collusive user detection involved in blackmarket following services' on Twitter.*

C2. We introduce DECIFE, a **novel framework** to detect collusive users on Twitter. It uses a hierarchical subgraph aggregation framework to leverage the follower/followee relationships and linguistic properties of users.

C3. We prepare **a new dataset** of collusive users who are involved in blackmarket following services on Twitter. *This, to our knowledge, is the first dataset of this kind.*

C4. We conduct extensive experiments on the curated dataset of collusive users to show the **superiority of our method** over state-of-the-art approaches.

Reproducibility: To encourage reproducible research, we have made the codes and the anonymized version of the dataset publicly available at https://anonymous.4open.science/r/DECIFE-140E.

7.2 Related work

We discuss the related literature by dividing the existing work into two parts – (i) detection of fake followers in OSNs, and (ii) study of blackmarket services in OSNs.

7.2.1 Detection of fake followers in OSNs

Most of the approaches to detect fake followers identify a set of features and use machine learning techniques. Wiltshire, in her blog[1], discussed the population of fake followers and found that there is an increase of 1-3% fake followers for a Twitter account every couple of months. [172] mentioned that there are two reasons to buy fake followers: increase visibility and push advertisements. A tool, called "Fake Followers Check"[2] was developed to detect fake followers on Twitter based on the ratio of friends and followers, usage of repeated spam phrases, count of retweets, etc. [172] proposed a machine learning approach to detect fake followers using multiple features and different machine learning classifiers. [173] used centrality-based graph features to detect fake Twitter followers. [174] found retweeters who can be used to spread the message effectively among different group of people. [175] proposed fBox, an algorithm to detect suspicious friend and follower links on a large who-follows-whom Twitter dataset. [176] developed a classifier for fake follower detection using profile, timeline and relationship based features. [177] proposed CatchSync to detect suspicious nodes (followers and botnets) exhibiting synchronized behavior on Twitter social network. [178] studied the dynamics of unfollow behavior in Twitter based on online relationships of Korean-speaking Twitter users. [179] identified users with increased follower count using unsupervised local neighborhood detection method. [180] proposed supervised spammer detection method with social interaction to detect spammers on twitter based on content and social interaction. [181] detected campaign promoters by mapping the problem to relational classification and solved it using typed Markov Random Fields. [182] discussed how social bots which interact with humans and get unnoticed have risen in the present scenario. [183] distinguished fake followers from the legitimate users using several discriminative features. [184] used a network-based strategy to identify fake followers. [185] focused on the detection of zombie followers in Sina Weibo.

[1] https://www.gshiftlabs.com/social-media-blog/the-fake-followers-epidemic
[2] https://bit.ly/2KzMRdd

7.2.2 Study of blackmarket services in OSNs

Blackmarket services have gained substantial attention recently because of the techniques they use to provide services to the customers. [167] was the first to analyze the Twitter follower markets based on the market size and market price. A detailed analysis of blackmarket services is presented in [186, 187] with the impact on multiple OSNs. Most of the prior studies showed how fake followers in social media help in promoting different agenda [167, 171, 188]. [189] showed how collusion networks collect 'OAuth' access tokens from colluding members and abuse them to provide fake likes or comments to their members. [51] proposed an automated approach to detect collusive behavior in question-answering systems. [190] studied models for detecting Instagram posts that gained interaction through collusive networks. [171] also discovered an oligopoly structure of merchants involved in blackmarket services. [168] studied multiple types of blackmarket agencies and analyzed a honeypot fraudster ecosystem to provide insights about multifaceted behaviour of different fraudsters. [191] analyzed structure of social networks present on six different underground forums to understand the social dynamics of e-crime markets. [192] studied various blackmarkets and developed a classifier to detect fraudulent accounts sold by these marketplaces. [193] studied the behavioral characteristics of Twitter follower market merchants based on user and content based features. [76] detected 'volowers' (followers who provide voluntary following services) who make profit in the follower markets. Recently, there are some preliminary works of collusive user detection on Twitter and YouTube. Our previous works [3, 5, 169] proposed techniques to detect collusive users involved in blackmarket-based retweeting service on Twitter. [7] proposed `CollATe`, an end-to-end framework to detect collusive entities on YouTube fostered by various blackmarket services. We encourage the readers to go through our comprehensive survey [194] on collusive activities in different online media platforms.

Differences with Previous Studies: The fundamental differences between the studies discussed above on fake follower detection and the collusive user detection are two-fold: (i) unlike fake followers who are mostly bots or whose activities are mechanically controlled by predefined policies, collusive users are normal human beings, and they themselves control their accounts. Therefore, unlike fake users who mostly show "synchronous behavior" [177], *collusive users are asynchronous in nature* [3]. (ii) Unlike fake followers, collusive users exhibit a hybrid following behavior – in one hand, being a normal user, they follow other users organically due to similar topical interest; on the other hand, being a collusive user, they randomly follow other blackmarket customers inorganically to gain credits [5]. *To the best of our knowledge, ours is the first work to detect* ***collusive users who gain artificial followers*** *from blackmarkets.*

7.3 Background and dataset

Approaching blackmarket services is one of the quickest ways to boost the impact of users on social media. These blackmarket services offer promotional services on multiple online media such as OSNs (e.g., likes on Facebook; followers, retweets, likes on Twitter; followers on Instagram), subscription-sharing platforms (e.g., views, subscribers, likes on Youtube), music-sharing platforms (e.g., plays, followers, likes, reposts, comments on SoundCloud, fans on ReverbNation), business and employment-oriented platforms (e.g., followers, connections, endorsements on LinkedIn), etc. We identified the blackmarket services providing collusive follower appraisals for Twitter by querying on search engines with keywords such as "buy free followers", "get me followers", "get followers quickly". To collect collusive users, we selected YouLikeHits[3], a popular credit-based freemium blackmarket service. Customers on navigating

[3]https://www.youlikehits.com/twitter2.php

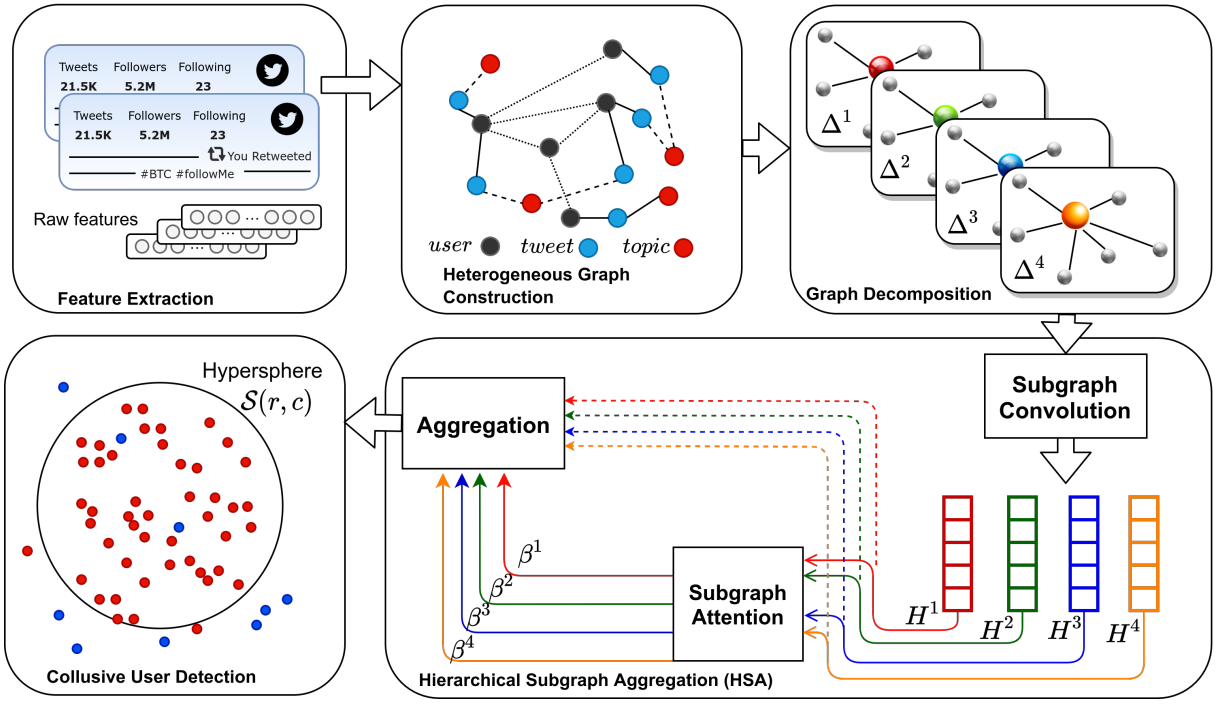


Figure 7.2: Architecture of our proposed DECIFE model for collusive user detection on Twitter.

to these services can see a dashboard (earning area) of other customers who are also using that service. YouLikeHits provide the customers with an initial credit of 50 which can be utilized to use various facilities offered by the service. After taking proper IRB approval from our institute, we developed a web scraper that used Selenium[4] to start a headless web browser navigating to the URL of the blackmarket service. We used popular Python packages such as Requests, BeautifulSoup etc. to parse the Twitter user IDs who submitted their profile for collusive follower appraisals. We further used the Tweepy library[5] to collect the metadata and timeline of the Twitter users. Note that we also collect a set of non-collusive users who surely have not participated in any kind of blackmarket-driven activities to gain artificial appraisals. This set of users is only collected for the test phase of our experiment and is not the part of original dataset.

7.4 DECIFE: Our proposed model

The objective of collusive user detection task is to learn how to automatically identify Twitter users who submitted their accounts to blackmarket services to gain collusive followers. Formally, given a set of collusive users $u_i = \{u_1, u_2, \dots\}$, connected via relationships $\Delta_m = \{\Delta_1, \Delta_2, \dots\}$ in a heterogeneous network $\mathcal{G}$, DECIFE learns a hypersphere boundary $\mathcal{S}(r, c)$ (with radius r and center c) around the collusive users to identify whether a new user is a collusive user or not. In this section, we present the architecture of DECIFE. It consists of four major components: *network construction*, *feature extraction*, *hierarchical subgraph aggregation* and *collusive user detection*. In this section, we introduce each of these components in detail. Figure 7.2 shows the schematic architecture of DECIFE.

[4] https://www.seleniumhq.org/
[5] https://github.com/tweepy/tweepy

7.4.1 Network construction

Here, we show the construction of the heterogeneous network and subsequent subgraphs for modeling the interactions of Twitter users.

Heterogeneous Network: To model the Twitter network, we construct a heterogeneous network [195] as a directed network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where each node $v \in \mathcal{V}$ and each edge $e \in \mathcal{E}$ are related with their node-type mapping functions $\rho(v) : \mathcal{V} \rightarrow \mathcal{X}$ and edge-type mapping functions $\varphi(e) : \mathcal{E} \rightarrow \mathcal{Y}$, respectively. $\mathcal{X} = \{\text{user}(U), \text{tweet}(T), \text{topic}(O)\}$ denote the node-types and $\mathcal{Y} = \{\text{follows}(r), \text{posts}(p), \text{contains}(c)\}$ denote the edge-types such that $|\mathcal{X}| + |\mathcal{Y}| > 2$ (a heterogeneous network contains at least 2 node-types or edge-types). An illustration of this network is shown in Figure 7.1. Every Twitter user $u_i \in U$ is represented with a feature vector x_i in $\mathcal{G}$ and is connected with its corresponding tweets $t_{ij} \in T$ and to users she follows or is followed by. Every tweet t_{ij} of a user u_i is connected to a topic node $o_k \in O$ via one-to-one mapping, where k is the set of topics.

Network Decomposition: The heterogeneous network is decomposed into subgraphs based on multiple relationships. A Relationship Δ between a labelled[6] user-node u_i and another labelled user-node u_j is a sequence of connected relations $\mathcal{M}_1 \rightleftharpoons \mathcal{M}_2 \cdots \rightleftharpoons \mathcal{M}_n$ in $\mathcal{G}$. Each relation $\mathcal{M}$ from source node $v_1 \in \mathcal{V}$ to target node $v_2 \in \mathcal{V}$ with edge $e \in \mathcal{E}$ is denoted as $\langle \mathcal{X}(v_1), \mathcal{Y}(e), \mathcal{X}(v_2) \rangle$. Note that (v_1, v_2) can belong to any node-type connected via edge e.

To capture different aspects of a user's behavior, we use four types of relationships, $\Delta_m, m = \{1, 2, 3, 4\}$ which are detailed below:

C1. Δ_1: To describe a connection between users in the 'following' blackmarket services, we formulate a relationship having a **common follower**.

$$\text{user}_1 \xrightarrow{follows} \text{user}_x \xleftarrow{follows} \text{user}_2$$

C2. Δ_2: Users in a blackmarket service tend to participate in a *barter system* where they follow other users to gain credits that can later be used to obtain followers of their accounts. As an example, a **transition user** (user_x) signed up on the service, **follows users** (user_1) to gain credits which can be availed **to gain followers** (user_2) in return. To exploit this characteristic, we formulate a relationship between users in a 1-hop manner.

$$\text{user}_1 \xleftarrow{follows} \text{user}_x \xleftarrow{follows} \text{user}_2$$

C3. Δ_3: To describe a **direct connection** between two users, we formulate a relationship in a 0-hop manner.

$$\text{user}_1 \xrightarrow{follows} \text{user}_2$$

C4. Δ_4: Twitter users involved in the promotion of campaigns/websites/events try to gain popularity from a particular target audience by gaining more retweets/followers to their tweets/accounts. To capture the connection between users tweeting/retweeting on the **same topic**, we formulate a relationship based on their common topic of interest. We detail this strategy of connecting users based on topical interest in Section 7.5.2.

$$\text{user}_1 \xrightarrow{posts} \text{tweet}_1 \xrightarrow{contains} \text{topic}_1 \xleftarrow{contains} \text{tweet}_2 \xleftarrow{posts} \text{user}_2$$

[6]We refer to a labelled user as a user whose ground-truth label is known. In our case, it refers to a collusive user which we collected from the blackmarket service.

Table 7.1: User metadata features used by DECIFE.

Feature	Description	Type
Follower count	Number of followers of a user	Real
Friend count	Number of friends of a user	Real
Status count	Number of tweets posted and retweeted by a user	Real
Favorite count	Number of status liked by a user	Real
Description presence	If the account description is present	Binary
Description length	Number of characters present in the user description	Real
URL	If a URL is present in the user description	Binary
Location	If the location is present in the user info	Binary
Profile image	If a profile image is present in the user account	Binary
Background image	If a background image is present in the user account	Binary
Account age	Number of days elapsed since the account was created	Real
Account entropy	Skewness in user's activity, $\sum c_i \log(c_i)$, where c_i is the year-wise count of tweets by the user	Real
Emojis	Avg. number of emojis per tweet posted by a user	Real
Retweets	Ratio of retweet count to tweet+retweet count	Real
URLs	Avg. number of URLs per tweet by a user	Real
User mentions	Avg. number of user mentions per tweet by a user	Real
Words	Avg. number of words in a tweet by a user	Real
Hashtags	Avg. number of hashtags in a tweet by a user	Real

7.4.2 Features extraction

In a Twitter user network, the behaviours and attributes of an account are directly linked with the user's characteristics. Taking this into account, we consider 18 features which are directly extracted or calculated from the user metadata. We describe each feature, its corresponding description and type in Table 7.1. We then concatenate these features to form an 18-dimension user-metadata feature vector that is used to train our DECIFE model. The user-metadata feature vectors, along with subgraph information, are fused at 2-levels using Hierarchical-Subgraph Aggregation (HSA) network to obtain final user representations.

7.4.3 Hierarchical Subgraph Aggregation (HSA)

We employ a two-level hierarchical subgraph aggregation mechanism to detect whether a user is collusive or not. In the first step, a weighted subgraph convolution network is used to obtain hidden representations of the labelled users based on the connections of a user node u_i with its associated users $u_j, j \in \mathcal{N}_i^m$ ($\mathcal{N}_i^m$ being the set of neighbors of users u_i connected via relationship Δ_m). Next, the subgraph attention network learns the importance of different edge-types to aggregate information among all nodes in the network.

7.4.3.1 Subgraph convolution network

Considering that a pair of users in a subgraph can be connected via multiple edges (e.g., two users posting multiple different tweets on the same topic), we propose a weighted subgraph convolution network with the following layer propagation rule:

$$h_i^m = \sigma(b^m + \sum_{j \in \mathcal{N}_i} \frac{e_{ij}^m}{c_{ij}^m} x_j w^m) \tag{7.1}$$

where e_{ij}^m is the edge-weight coefficient for the edge e_{ij} connected via relationship Δ_m (c.f. Table 7.3), and $\mathcal{N}_i$ is the set of neighbors of node i. Further, $c_{ij}^m = \sqrt{|\mathcal{N}_i||\mathcal{N}_j|}$ is the normalization term, and $|\mathcal{N}_i|$

is the degree of node i. Here, x_j is the input feature vector of node j, and w^m is the learnable weight parameter for the subgraph m. The weight coefficient in subgraph convolution network makes it superior for learning the connectivity information between the user-nodes than a conventional graph convolution network.

In generalised-matrix form, the equation can be written as:

$$H^m = \sigma((D^m)^{-1} A^m K^m X W^m) \tag{7.2}$$

where A^m is the adjacency matrix, K^m is the edge-weight matrix, $D^m_{ii} = \sum_{j \in \mathcal{N}_i} A^m_{ij}$ is the diagonal matrix, and W^m is the learnable layer-wise weight matrix of a subgraph having relationship Δ_m.

Therefore, we obtain 4 groups of subgraph-specific node-embeddings H^m for each given relationship $\Delta_m, m = \{1, 2, 3, 4\}$. These are then used to compute importance among subgraphs using Subgraph Attention Network as described below.

7.4.3.2 Subgraph attention network

Subgraph convolution network learns a node representation by leveraging the importance between users only from a single type of relationship. However, attending to multiple relationships at once can reveal the importance of different edge-types to a node pair and aggregate rich semantic information from them. The subgraph-specific node-embeddings H^m obtained from subgraph convolution network are projected using a multilayer feed-forward neural network to undergo a non-linear transformation.

$$\tilde{H}^m = W_1 \cdot H^m_i + b_1 \tag{7.3}$$

$$w^m = \frac{1}{n} \sum_{i}^{n} W_2(\tanh(\tilde{H}^m)) + b_2 \tag{7.4}$$

where W_1, W_2 are the weight matrices and b_1, b_2 are the bias vectors.

The weight coefficient β^m of each subgraph is calculated by normalizing the importance w^m using a softmax function.

$$\beta^m = \frac{\exp(w^m)}{\sum_{k=1}^{n} \exp(w_k)} \tag{7.5}$$

The information from both the levels can be aggregated to obtain the final embeddings as follows:

$$Z = \sum_{m=1}^{3} \beta^m \cdot H^m \tag{7.6}$$

To improve the stability of the training process, we extend this to multiple heads $d = \{d_1, d_2, \ldots, d_l\}$ by passing Z^{l-1} as an input through the Hierarchical Subgraph Aggregation Network,

$$Z^l = HSA(Z^{l-1}) \tag{7.7}$$

The final embedding Z^l of each user-node is used to learn a hypersphere boundary around the collusive distribution.

7.4.4 Collusive user detection

Given a training set $\{Z_n = Z_1, Z_2, .., Z_n\} \in \mathbf{R}^{n \times d}$ with embeddings of dimension d belonging to n collusive users $u_i \in U$ derived by hierarchical aggregation, our DECIFE model is trained to learn a hypersphere boundary that encloses the network representations around majority of the collusive data. The objective is to obtain a minimum volume hypersphere $S(r, c)$ with radius $r > 0$ and center $c \in \mathcal{F}_k$ using a mapping function $\phi_k : Z \rightarrow \mathcal{F}_k$ to map the feature space Z into Hilbert space $\mathcal{F}_k$. The loss $\mathcal{L}$ is similar to the DeepSVDD [196] and is defined as follows:

$$\mathcal{L} = \ r^2 + \frac{1}{\mu n} \sum_{i=1}^{n} \xi_i \quad s.t. \quad ||\phi_k(x_i) - c||^2_{\mathcal{F}_k} \leq r^2 + \xi_i, \quad \xi_i \geq 0 \ \forall \ i \tag{7.8}$$

The slack variables $\mu \in (0, 1]$ regulates the trade-off between penalties ξ_i and the volume of sphere $S(r, c)$, and ξ_i allows a smooth boundary of the sphere. The initial radius is set as 0, and the center is obtained through the first forward pass to our network model.

$$c_0 = \frac{1}{n} \sum \texttt{DECIFE}(\mathcal{G}) \tag{7.9}$$

During validation and testing, a point is marked as collusive user if it falls inside the hypersphere S such that $||\phi_k(x_i) - c||^2_{\mathcal{F}_k} > r^2$.

7.5 Experimental setup

We conduct an extensive evaluation of DECIFE on our collected dataset (mentioned in Section 7.3). In this section, we start by briefly describing the baseline methods, followed by detailed evaluation.

7.5.1 Baseline methods

We consider three state-of-the-art methods – first two focus on fake follower detection and the third one focuses on Twitter blackmarket follower services. However, since these methods are not focused towards a network-based detection approach, we conduct an ablation study by considering one relationship at a time in our model. We also add these methods for comparing with DECIFE, which is a combination of all the relationships.

FakeFol$_{ss}$: Castellini et al. [170] detected fake Twitter followers in a *semi-supervised* fashion by running denoising autoencoder. Their model is trained on the data of only one class (5% of collusive users).

FakeFol$_{s}$: Castellini et al. [170] further introduced a supervised approach to compare it with semi-supervised approach mentioned in FakeFol$_{ss}$. They used a bunch of features related to user profile and user activity, and showed Random Forest to be the best classifier.

FolMarket: Aggarwal et al. [171] characterized underground blackmarket services which provide Twitter followers. They explored three units of blackmarket services – merchants, customers and phony followers. They also built a supervised classifier (SVM with RBF kernel) to classify non-collusive users and phony followers.

Variants of DECIFE: We also derive multiple variants of DECIFE to comprehensively compare and analyze the performances of each component. These variants are **DECIFE**$_{\Delta_1}$, **DECIFE**$_{\Delta_2}$, **DECIFE**$_{\Delta_3}$ and **DECIFE**$_{\Delta_4}$. Note that Δ_i refers to relationships mentioned in Section 7.4.1.

Table 7.2: Dataset statistics.

Statistic	Value
# of train users	10,863
# of test users	426
# of tweets	6,627,277
# of topics	1,000

Table 7.3: Subgraph statistics.

Relationship Δ	# edges (i, j)	Edge-weight (e_{ij})
Δ_1	6,408,903	# common followers
Δ_2	7,206,330	# common transition users
Δ_3	64,669	1
Δ_4	88,069,870	$\sum_{k \in K} min((o_k)^i, (o_k)^j)$

7.5.2 Implementation details

7.5.2.1 Sentence embeddings:

Since Twitter is a global social-media platform and the intention of the collusive users is to earn followers regardless of the community, language identification of all tweets using Langid [197] revealed a mix of languages in the user timelines. Therefore, to capture the semantics of the tweets in different languages, we use Language Agnostic Sentence Embedding representations (LASER) [198] to build cross-lingual features of the users' tweets. For each user with δ historic tweets, we obtain a $\delta \times 1024$ dimensional embeddings. These multilingual embeddings are finally passed through a cosine-similarity based K-means clustering algorithm[7] for topic identification of each tweet.

7.5.2.2 Parameter settings:

We implement our model using Python 3.8.3, PyTorch 1.4.0 and Deep Graph Library [199]. The hyperparameters are tuned based on the validation set and results are reported on the test set. The penalty parameter μ is set to 0.2. The network parameters are randomly initialized with a fixed seed for all experiments and optimized using Adam optimizer. We set the learning rate to 0.6 and weight decay regularization with $\lambda = 0.0005$. For the subgraph convolution network, hidden size $H^m = 32$ is used. Further, the hidden dimension $\tilde{H}^m$ for subgraph attention network is set to 128, and 2 heads are used to train the HSA model.

7.5.2.3 Data statistics:

We start by collecting a real-world Twitter user dataset for the detection of collusive users. A total of 11,076 users involved in the "following" activity from blackmarket service are identified and labelled as collusive users. Among these, 10,863 users are randomly used for training `DECIFE`, and the remaining 213 users are used for testing. The entire data collection process from the blackmarket service is mentioned in Section 7.3. Along with this, we take into account 213 users which are labelled as non-collusive. **We would again like to emphasize on the fact that we chose only those Twitter users as non-collusive who are guaranteed not to be a customer of any blackmarket service. The set of non-collusive users is only used at test time to measure the performance of `DECIFE`.**

Further, the followers, followees and tweets of all users are extracted using the Twitter API, which are used for building the heterogeneous network. The heterogeneous network contains around 7.9 million user nodes, 6.6 million tweet nodes and 1000 topic nodes. This network is decomposed into four subgraphs, each having $11,289$ users as nodes connected via different relationships Δ. The results are reported after 10-fold cross-validation. Detailed statistics of the dataset and the subgraphs are shown in Tables 7.2 and 7.3, respectively.

[7] https://github.com/facebookresearch/faiss

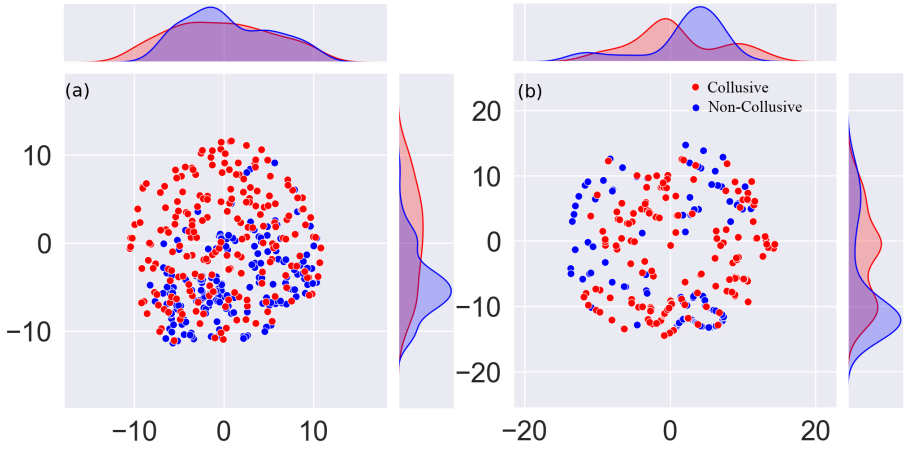


Figure 7.3: TSNE visualized features of collusive (red) and non-collusive (blue) users from the test dataset. The density estimate plots along the axis depict the probability distribution of the users in both sets as (a) raw features, and (b) fully-trained model.

7.6 Experimental results and analysis

7.6.1 Performance comparison

In order to evaluate the performance of DECIFE against the baseline methods, we use three popular evaluation metrics: AUC-ROC, AUC-PR and F1-score. Table 7.4 summarizes the comparative analysis of the competing methods. In general, DECIFE outperforms all the baselines. Among the baselines, FakeFol$_s$ turns out to be the best in terms of AUC-ROC and F1-score. Among different variants of DECIFE, we observe that DECIFE$_{\Delta_2}$ outperforms all the other variants and our proposed model in terms of F1-score. The reason behind this is the correct identification of a higher number of non-collusive users (true positives) in comparison to the collusive users (true negatives), which is not the primary objective of our predictive modeling task[8]. Note that for each of the evaluation metrics, we average the final values over 10 runs. As can be seen in Table 7.4, our proposed DECIFE model consistently achieves the best performance over others by a significant margin. We observe that the performances of the baseline methods are poor, indicating that they fail to generalize for collusive user detection as they are primarily targeted towards fake user detection. On the contrary, DECIFE leverages the benefit of multiple relationships that prove the effectiveness of heterogeneous network to integrate multi-type information.

Table 7.4: Performance comparison of the competing models.

Method	AUC-ROC	AUC-PR	F1-score
FakeFol$_{ss}$	0.417	0.455	0.537
FakeFol$_s$	0.617	0.568	0.683
FolMarket	0.500	0.476	0.645
DECIFE$_{\Delta_1}$	0.427	0.423	0.606
DECIFE$_{\Delta_2}$	0.689	0.563	**0.802**
DECIFE$_{\Delta_3}$	0.818	0.833	0.745
DECIFE$_{\Delta_4}$	0.771	0.745	0.688
DECIFE	**0.895**	**0.854**	**0.786**

The TSNE visualizations (c.f. Figure 7.3) of the test set embeddings for collusive and non-collusive users indicate that raw features cannot be used for detecting collusive users, whereas in the trained model, the learned hypersphere clearly encloses and distinguishes the collusive users. Moreover, the Kernel Density Estimate (KDE)

[8]93.4% of the non-collusive users and 60.5% collusive users are predicted correctly by DECIFE$_{\Delta_2}$.

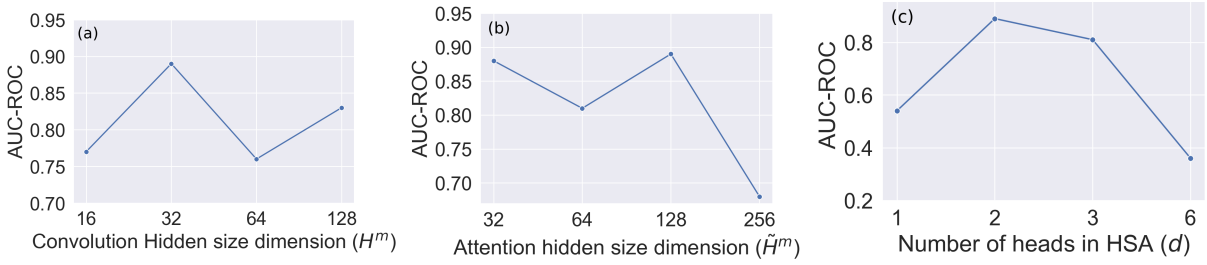


Figure 7.4: Parameter sensitivity of DECIFE w.r.t (a) convolution hidden size dimension, (b) attention hidden size dimension, and (c) the number of heads.

plots in Figure 7.3 depict the probability distribution of users in both sets. The probability density of collusive users in the trained model is high at points where density of non-collusive users is low, which is not the case in raw features. This suggests that the collusive users can be easily partitioned in the embedding space using the final trained embeddings by our converged DECIFE model.

7.6.2 Sensitivity of parameters

We also investigate the sensitivity of parameters and report the results (AUC-ROC) of our proposed model with various parameters in Figure 7.4. We consider the following three parameters:

- *Convolution hidden size dimension* (H^m): We first test the effect of the dimension of hidden layer in the subgraph convolution network. The results are shown in Figure 7.4(a). We can observe that our model achieves the best performance when H^m is set to 32. However, the performance is very sensitive with the change in the dimension, suggesting that the convolution hidden size H^m plays a crucial role in comparison to other parameters.
- *Attention hidden size dimension* ($\tilde{H}^m$): We explore our experimental results by varying the size of the attention vector ($\tilde{H}^m$) during projection. The result is shown in Figure 7.4(b). We can find that the performance of DECIFE generally slightly varies upto the hidden dimension size 128. For higher dimensions, the model struggles to converge and overfits as the 32-dimensional embedding is projected into a 256-dimensional embedding.
- *Number of heads* (d): In order to check the impact of our proposed aggregation framework, we measure the performance of DECIFE with the number of heads d. The result is shown in Figure 7.4(c). Based on the results, we observe that multi-head HSA model ($d > 1$) performs better than single head ($d = 1$). But as the complexity increases ($d = 6$), the model starts to overfit, and the performance is significantly impacted. Also, we find that the training process becomes more stable with the growth in the number of model heads.

7.6.3 Qualitative analysis

We qualitatively examine all 25 false negatives produced by our DECIFE model in the testing phase of our dataset. In a Twitter network of collusive users, the priority is to minimize social damage; we, thereby, review the collusive users which are falsely detected as non-collusive by our model. We find that 18 of these users (~72%) have less than 25 followers, thereby forming a sparse connectivity in our proposed heterogeneous graph. This also makes them unable to learn the connectivity information between the nodes and thereby limiting the performance of DECIFE to identify these users. For the remaining 7 users, we could not find enough conclusive evidence from our collected data to interpret their incorrect detection by our proposed model. We would also like to mention that after manual investigation, 4 out of these 7 users do not exists currently on Twitter. We observed that DECIFE is successfully able to detect collusive users but misclassifies some of the genuine users as collusive. Upon deeper investigation, we found that the misclassified genuine users have a high follower to followee ratio showing characteristics similar to the collusive users.

Figure 7.5: Wordcloud of the hashtags used by collusive users (true negatives)

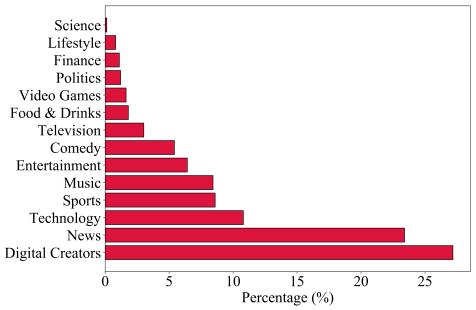


Figure 7.6: Category distribution of tweets posted by collusive users

Table 7.5: Example of tweets posted by collusive users.

Tweet	Category
Boost Your Google Ranking Fast with 250 High Quality Backlinks #fiverr #backlinks #traffic #website	Freelancing
OMG, I have 1,000 followers! Thank you. Think I will make it to 10,000? #followme	Blackmarkets
This is the first spinner that brings free BTC. Spin & Earn. #btcspinnerio #btcspinner #bitcoin #btc	Digital asset
This twitter site is all about #sports #gambling, sports #handicapping. He give my free picks on #Soccer, #NFL-Foot	Sports
Find The Latest Airdrop $ Bounty $ Award. $2000 Free Tokens Are Airdropping Now! Hunt now	Digital asset
Get up to 50GB of free space for all your photos, videos, docs, and music	Technology
Check it out! socialspider will do viral youtube promotion usa or worldwide for $5 on #Fiverr	Freelancing
#CyberMiles is a new blockchain protocol being developed to revolutionize how digital commerce and online marketplace	Digital asset
Yo Bro my Music is poping right now. Everysong i drop on soundcloud touches 1000 in a Month. Can you drop my songs on your Youtube.	Entertainment
Grab them while they last, 100 free DAT Tokens here #datumnetwork #ico #cryptocurrency	Digital asset
Get $1million $CPPG tokens. Airdrop bounty Crypto cryptocurrency	Digital asset

In order to understand the behaviour of the correctly identified collusive users (true negatives[9]), we highlight interesting quantitative covariates among these accounts.

Current user status: At the time of conducting this research, we found that only 2% of detected collusive users which are collected from the blackmarket services are suspended/deleted by Twitter. This clearly shows how these users are successfully able to evade the existing fraudulent account detection techniques deployed by Twitter.

Popular hashtags: We analyze a total of 119,486 tweets posted by the collusive users detected using DECIFE covering their entire tweeting activity in our dataset. Figure 7.5 shows the wordcloud of the hashtags used in tweets after removing two-letter words and common stopwords. Here, the font size corresponds to the frequency of the text. Unsurprisingly, we observe keywords related to the blackmarket services ('followme'), entertainment ('music', 'soundcloud', 'edm', 'youtube', 'spotify'), freelance platforms('fiverr'), digital asset ('cryptocurrency', 'bitcoin', 'blockchain'), etc. This shows how collusive entities could have manipulated the social growth of these entities. We believe these keywords also provide new directions for future collusive entity detection. We list out a few example tweets with these keywords in Table 7.5.

Tweet categories: It is often interesting to see which categories of tweets are posted by fraudulent users. We follow a zero-shot learning approach mentioned in [200] to identify the categories present in the tweets posted by the detected collusive users. We frame it as a zero-shot topic-classification problem where the task is to map a sentence (tweet in our case) to a class (category in our case) that is not observed during training. We map the tweets into

[9]Note that collusive class is the negative class in our experiment.

one of the 14 Interactive Advertising Bureau (IAB) categories[10] used by the supply partners within the Twitter Audience Platform (TAP). We show the category distribution of tweets posted by the detected collusive users in Figure 7.6. It can be clearly seen that almost 50% of the tweets are from two categories: *Digital creators* and *News*. A possible reason for this is the recent popularity of these categories on Twitter, allowing tweets from these categories to connect directly with their audience, discover emerging content and trends, and build communities using two monetization programs[11]: *Amplify Pre-roll* and *Amplify Sponsorships*.

7.7 Conclusion

The prevalence of collusion over social media has become a serious problem. In this work, we addressed the problem of detecting collusive users involved in blackmarket services to produce illegitimate followers for Twitter users. We proposed DECIFE, a framework that considers the follower/followee relationships and linguistic properties of tweets for detecting collusive users in Twitter. Compared with previous methods, our approach focuses on multiple relationships between the users and can leverage information more efficiently. Using extensive experiments, we demonstrated the effectiveness and superiority of DECIFE with an attention mechanism on our curated dataset over the state-of-the-art methods. To the best of our knowledge, this work is the first attempt using heterogeneous network to detect collusive followers on Twitter. The dataset we collected is also the first dataset of its kind. For future work, in addition to the metadata and topical properties, we plan to capture the temporal properties of these users that can help us improve the performance of the predictive modeling task.

[10] https://developer.twitter.com/en/docs/twitter-ads-api/campaign-management/api-reference/iab-categories

[11] https://media.twitter.com/en/articles/products/2018/media-studio/monetization.html

8. Detecting and analyzing collusive entities on YouTube

YouTube sells advertisements on the posted videos, which in turn enables the content creators to monetize their videos. As an unintended consequence, this has enabled various illegal activities such as artificially boosting of views, likes, comments, and subscriptions. We refer to such *videos* (gaining likes and comments artificially) and *channels* (gaining subscriptions artificially) as "collusive entities". Detecting such collusive entities is an important yet challenging task. Existing solutions mostly deal with the problem of spotting fake views, spam comments, fake content, etc., and oftentimes ignore how such fake activities emerge via collusion. Here, we collect a large dataset consisting of two types of collusive entities on YouTube – *videos* submitted to gain collusive likes and comment requests, and *channels* submitted to gain collusive subscriptions.
We begin by providing an in-depth analysis of collusive entities on YouTube fostered by various *blackmarket services*. Following this, we propose models to detect three types of collusive YouTube entities – videos seeking collusive likes, channels seeking collusive subscriptions, and videos seeking collusive comments. The third type of entity is associated with temporal information. To detect videos and channels for collusive likes and subscriptions respectively, we utilize one-class classifiers trained on our curated collusive entities and a set of novel features. The SVM-based model shows significant performance with a true positive rate of 0.911 and 0.910 for detecting collusive videos and collusive channels respectively. To detect videos seeking collusive comments, we propose `CollATe`, a novel end-to-end neural architecture that leverages time-series information of posted comments along with static metadata of videos. `CollATe` is composed of three components – metadata feature extractor (which derives metadata-based features from videos), anomaly feature extractor (which utilizes the comment time-series data to detect sudden changes in the commenting activity), and comment feature extractor (which utilizes the text of the comments posted during collusion and computes a similarity score between the comments). Extensive experiments show the effectiveness of `CollATe` (with a true positive rate of 0.905) over the baselines.

8.1 Introduction

Online media is increasingly becoming the most effective medium for sharing ideas, thoughts, and information. The primary reasons behind the large user engagement are the ease of accessibility and ever-evolving attractive sharing facility. The content shared in online media usually includes personal data, documents, photos, and videos. Generally, videos are used to convey meaningful information in a shorter duration, providing unparalleled advantages to content consumers. YouTube, an online video-sharing platform, allows users to upload, share, or live-stream videos on the Internet. Free service, measurable analytics, availability of multiple genres, and access to broad audiences have made YouTube the most popular platform for both content creators and content consumers. The popularity of a YouTube video is generally determined by the number of likes or comments it receives over a period of time. Similarly, the popularity of a YouTube channel is measured by the number of subscribers. The natural way to gain traffic to a channel/video and make it noticeable is a time-consuming process for content creators. Apart from creating attractive content, oftentimes, the content creators become desperate to figure out effective shortcuts to gain quick popularity of their content and channels. This may lead them to choose unethical and inorganic ways with the aid of blackmarket services to gain popularity within a short duration. The primary objective of an artificial boosting mechanism is to convince the YouTube ranking algorithm to prefer a video/channel over its competitors.

In recent years, YouTube has started a *YouTube Partner Programme*[1] where content creators can make money from the advertisements and other revenue streams. However, the channel has to meet minimum two requirements – *at least* $1,000$ *subscribers* and *at least* $4,000$ *hours of watch time within the past 12 months*. Furthermore, content creators who are new to YouTube and have managed to attract only a small audience do not get significant engagement despite having fantastic content. It leads them to invest to the blackmarket services to reach out to more audiences and increase their revenues in a faster and effective way. The blackmarket-driven artificial engagement is strictly against the YouTube Terms of Service and may lead to a permanent ban on the user accounts. To counter this, blackmarket services provide their facilities in such a way that collusive entities (channels and videos) are evaded from being detected by the in-house fake detection algorithms deployed by YouTube. Interestingly, in our dataset, we found 7 such collusive channels, which are marked as *verified* by YouTube! This clearly shows that YouTube is unable to detect such fraudulent entities, thereby creating an inadequate social space for the entire YouTube population.

Throughout this study, we use the following nomenclature. A YouTube video is said to be *collusive* if likes or comments of the video are artificially inflated with the help of blackmarket services. Similarly, a collusive YouTube channel is the one which receives artificial subscriptions from blackmarket services. We use *collusive entities* to refer to both collusive videos and channels.

The current work provides a large-scale investigation and detection of collusive entities on YouTube. We create a unique dataset of collusive entities collected from YouLikeHits (a credit-based freemium blackmarket service). *This, to our knowledge, is the first labeled dataset of YouTube collusive entities*. We start our analysis by examining videos submitted to blackmarket services for collusive appraisals (likes and comments) based on two perspectives – propagation dynamics and video metadata. We then analyze the collusive channels based on location, channel metadata, and network properties. In the collusive channel network, the structural properties of the giant component show that it is a small-world. Further, we are interested in detecting whether a new entity is collusive or not. Since we collected the collusive data directly from the blackmarket websites, we are sure that the collected data does not have any noisy labels. In such cases, instead of collecting a large dataset representing the entire YouTube population, we rather focus on designing sophisticated methods that can leverage the characteristics of entities in one class (collusive in our case) and learn their representations for the prediction task. Note that we were unable to extract any temporal information related to like and subscription activities for videos and channels respectively, due to the restrictions and limitations of the YouTube API.

We propose three models to detect three types of collusive entities. The first and the second models utilize one-class classifiers trained only on the collusive entities using video metadata and channel features to detect collusive videos and channels respectively. The third model attempts to detect videos seeking collusive comments. In addition to the video metadata, it uses textual and temporal information of each comment. The temporal information indicates the aggressive patterns of blackmarket users (as these users aggressively post comments on collusive YouTube videos in order to gain credits). In addition to this, the semantic representation of each comment enables us to learn the similarity pattern of these users in posting comments and also to evade the existing fake detection strategies. During the collection of this dataset, we realized that many videos have a limited number of comments despite being posted for collusive comments to the blackmarket services. We discarded such videos as the comments were possibly deleted, which might result in noisy information in the labeled data. Finally, we end up collecting a relatively smaller dataset of videos seeking collusive comments compared to those videos/channels seeking collusive likes/subscriptions. This further raises another challenge of learning generalized and robust representation. Thus, by incorporating these properties of blackmarket users and taking into account the issues mentioned above, we propose `CollATe`, a denoising autoencoder model to detect videos submitted in blackmarket services for collusive comments. `CollATe` consists of three components – metadata feature extractor, anomaly feature extractor, and comment feature extractor to learn feature representation of videos.

The first and second models achieve significant accuracy with a true positive rate of 0.911 and 0.910 respectively. The third model, `CollATe` achieves a true positive rate of 0.905, outperforming seven baselines. Altogether this work sheds light on how collusion happens on YouTube and focuses on the detection of collusive entities to make YouTube an adequate video sharing platform for the content creators and content consumers. We believe this could be used to curb the adverse effects of collusion in gaining artificial social growth. We summarize the contributions of this work as follows:

[1] https://support.google.com/youtube/answer/72851hl=en

- To the best of our knowledge, we are the first to investigate YouTube videos and channels submitted to blackmarket services for collusive appraisals.
- We prepare four unique datasets of collusive entities on YouTube: (i) videos submitted to YouTube for collusive likes, (ii) videos submitted to YouTube for collusive comments, (iii) channels submitted to YouTube for collusive subscriptions, and (iv) the network of collusive YouTube channels. The dataset is attained using YouTube API and custom-designed scrapers that can be easily extended and used to collect large YouTube data in an effective way. We believe that these four datasets would help researchers analyze blackmarket-driven collusive activities happening on YouTube and develop tools to detect them.
- We analyze the YouTube videos for collusive likes and comments from two perspectives – propagation dynamics and video metadata. We also analyze collusive YouTube channels based on their location, channel metadata, and network properties. The giant component of the collusive channel network turns out to be a small-world.
- We utilize one-class classification models to detect videos and channels submitted to blackmarkets for collusive likes and subscriptions. We also propose `CollATe`, an end-to-end neural framework to detect YouTube videos seeking collusive comments.

Reproducibility: Codes and datasets are available at the following link: https://github.com/LCS2-IIITD/CollATe.

Organization of the study: We now discuss the organization of this work. Section 8.2 discusses related work for the study. Section 8.3 presents a detailed study of the blackmarket services and motivates the research problem of collusive entities in YouTube. Section 8.4 discusses the data collection strategy. In Section 8.5, we deeply analyze collusive YouTube videos and channels. Section 8.6 introduces the proposed frameworks to detect collusive entities. We present the experimental results in Section 8.7. Section 8.8 shows the important implications of the collusive entities detected using our models. We conclude the study with future directions in Section 8.9.

8.2 Related work

We divide the relevant related work into two parts: (i) fraud/spam detection in online media, and (ii) studies on blackmarket services.

8.2.1 Fraud/spam detection in online media

Plenty of studies focused on the detection of fraudulent activities in a wide range of platforms. In recent years, fraudulent activities are most common in major online media sites such as Facebook [14, 15, 16], Instagram [17, 18] and Twitter [19, 20, 21, 22, 23, 24, 25, 26]. [27] presented a review on existing fake and fraud detection strategies in online media platforms. A number of fake comment detection strategies [28, 29] on YouTube were also proposed in recent years. Most of these studies rely solely on the textual data of the comment. Nevertheless, there is no prior work that considers the temporal properties of comments, which in the case of collusion, is the most important factor, the reason being collusive users perform appraisal operations aggressively in order to gain credits rapidly. [30] proposed `LEAS`, an algorithm to detect fake engagements in video sharing platforms using a temporal engagement graph between users and video objects. [31] proposed a set of tools to detect view fraud in online video portals. [32] investigated the problem of fake views caused by robots in video sharing platforms.

In the field of fraud detection in online advertising, [33] proposed an advertising network model to discover coalitions between pairs of fraudsters in e-commerce platforms. [34] designed an automated approach for ad networks to detect click-spam attacks. [35] analyzed disinformation and crowd manipulation tactics on YouTube by analyzing video metadata. [36] examined intensive groups among YouTube commenter networks by constructing a two-level optimization problem for maximizing local degree centrality and global modularity measures. [37] conducted a longitudinal analysis of the promotion of conspiracy videos on YouTube. [38] studied the problem of injection attacks on the recommendation systems (fake co-visitations) of YouTube. A number of studies have been conducted on detecting spam on video streaming platforms. [39] proposed `Tubespam`, a novel classification model for comment spam filtering on YouTube. [29] studied the performance of five state-of-the-art text feature selection

methods for spam filtering on YouTube using Naive Bayes and Decision Tree. [40] detected video spammers on YouTube based on the EdgeRank algorithm to decide which post/stories should appear in each user's news feed. [41] proposed a spam detection system for YouTube using a set of spam-related attributes from videos. [42] detected forum spammers on YouTube based on the mining comment activity log of a user and extracting patterns indicating spam behavior. [43] detected spam comments on YouTube by showing the effectiveness of using character n-grams instead of word n-grams to improve the accuracy of the classification model.

More recently, YouTube has become the most important tool for live streamers. In our dataset, we found around 30% of the collusive channels involved in streaming live videos. [44, 45] discussed the detection of copyright infringement on YouTube live videos. [44] developed a crowd-sourced based copyright infringement detection (CCID) scheme from live chat messages on YouTube and Twitch to identify original copyright content from the owner. [45] presented an end-to-end supervised detection framework to combat copyright infringement in live video streams using the live chat messages from the audiences. However, these methods tend to combat the problem of fake, fraud, and spam detection in video-sharing platforms but are not applicable for collusive entity detection.

8.2.2 Studies on blackmarket services

Despite the fact that a plethora of studies exist on detecting fraud/spam activities in online media, there has been relatively less work on investigating blackmarket services providing collusive appraisals. [46] focused on how manipulators create disinformation by fake engagement activities on YouTube. [47] provided a broad overview of the blackmarket services providing fake YouTube views. The authors reported that one of the blackmarket services, named Devumi had earned more than $1.2 million in around 3 years of service by selling 196 million views.

Studies that are more related to the current work include our previous studies [3], [5], [169] and [4], which investigated the problem of blackmarket-based collusive activities in Twitter. [6] studied multiple types of blackmarket link fraud behaviors in Twitter by analyzing the connectivity patterns of fake followers via the egonet and boomerang networks. In our first work, we proposed `SCoRe`[3], a supervised method to detect collusive retweeters affiliated to blackmarket services in Twitter. We also showed the differences between fake and collusive activities based on the synchronicity of retweet behaviors. We then extended [5] our previous work [3] to show the differences between the working principles of premium and freemium blackmarket services. After that, we proposed `CoReRank`[4], an unsupervised method to detect collusive users and suspicious tweets by leveraging the user's retweeting and quoting patterns. We also [201] proposed `HawkesEye`, a framework to detect fake retweeters using Hawkes process and topic modeling on tweets. [48] proposed a multi-task learning approach to detect tweets submitted to freemium blackmarket services. [49] proposed `DeFrauder`, an unsupervised method to spot online fraudulent collusive groups in review websites. [50] proposed a review graph model to detect spammers in online review stores. [51] proposed an automated approach to detect collusion behavior in online question-answering systems. Other studies identified fake followers on Twitter [9, 52, 53, 54]. [55] and [56] are some of those who used network-centric properties to detect fake followers. `Fake Follower Check`[2] is one such tool to detect fake followers based on profile-centric and behavioral features of Twitter users.

To the best of our knowledge, there has been no major attempt to detect collusive entities on video sharing platforms such as YouTube. Our current effort provides a deeper understanding of the collusive activities on YouTube and focuses on designing automated approaches to detect these activities.

8.3 Background

8.3.1 Blackmarket services

Blackmarket services help online media users in gaining appraisals inorganically for their content. They offer services related to online social networks (Facebook: likes, follow, share; Instagram: likes, follow; Twitter: retweets,

[2] https://1tnr.short.gy/socialbakers

likes, follow), recruitment platforms (LinkedIn: endorsement, recommendations, connections), video sharing platforms (YouTube: video views, video comments, channel subscriptions; Vimeo: video views, video comments), etc. The inorganic appraisals help in artificial boosting of online media content, thereby creating an inadequate social space. Online media entities such as big companies, advertising firms seeking active participation in their promotional campaigns target these websites to expedite their reach to their target audiences.

The blackmarket services are divided into two types based on the mode of service [202]:

- **Premium services:** These services charge customers for the facilities they provide. Customers have to register themselves and opt for one of their plans to gain appraisals.
- **Freemium services:** These services are free of cost and work like a barter system. The primary goal of freemium services is to let their customers familiarize with free services and convince them to subscribe to the premium plans. Most of the freemium services are *credit-based*, where each customer receives virtual credits by appraising the content of other customers.

In this work, we focus our analysis on *YouLikeHits*[3], a credit-based freemium service which helps content creators on YouTube escalate their subscribers, views, comments and likes.

8.3.2 Collusion on YouTube platform

YouTube is a video sharing platform where users can upload videos by creating YouTube channels. The platform is free of cost and is operated by two types of users:
(i) *Content creators:* Users who upload videos to their channel.
(ii) *Content consumers:* Users who watch videos, interact with videos in the form of likes/comments, or subscribe to channels.

When a content creator uploads a video, content consumer can perform the following actions – *like* the video, *dislike* the video, *comment* on the video, *share* the video, *save* the video to wishlist, and *subscribe* the content creator's channel. The popularity of a YouTube video is measured by the number of appraisals it receives from the content consumers. Thus the content creators need to ensure that their content receives high appraisals from the consumers. Moreover, with the advent of the concept of *Monetization on YouTube*[4], content creators have begun to attract audiences by uploading videos aimed towards specific genres such as teaching, entertainment, business, etc. This may further motivate them to choose artificial ways to gain quick popularity in their content. Currently, the earning potential of a channel/video is solely driven by the number of subscribers/views. When a video is posted on YouTube, it is shared with the YouTube community of similar channels as recommendations. With millions of videos posted every second, a majority of the videos go unnoticed to the target audience. The organic way of gaining appraisals is a tedious task, which leads content creators to opt for an alternative way by means of blackmarket services.

8.3.2.0.1 How collusion happens on YouTube? Collusion on YouTube happens when a video or a channel is posted in blackmarket services for appraisals. In this work, we refer to a user who submits the content to the blackmarket service as a *collusive user*. In a freemium blackmarket service, collusive users receive credit points upon performing appraisals on the content of other collusive users. In the case of YouTube, the majority of the blackmarket services request the collusive users to submit only the video or channel URL. Collusive users can contribute to the artificial boosting of YouTube videos and channels in several ways: (i) viewing other videos, (ii) posting a comment on videos[5], (iii) posting likes on a video, and (iv) subscribing to a channel.

Fig. 8.1 shows the peaks observed in collusive videos (in red) for collusive comment appraisals detected using our proposed approach, `CollATe`. However, we do not see any such peaks in other random videos (more details can be found in Section 8.6.3.3). The reason is that the collusive users tend to perform collusive appraisals aggressively to gain credit points, which they can use later to add new content. This aggressive nature results in

[3] https://www.youlikehits.com/
[4] https://www.youtube.com/account_monetization?nv=1
[5] Note that blackmarket services may ban a collusive user upon identifying a spammy comment posted by the user.

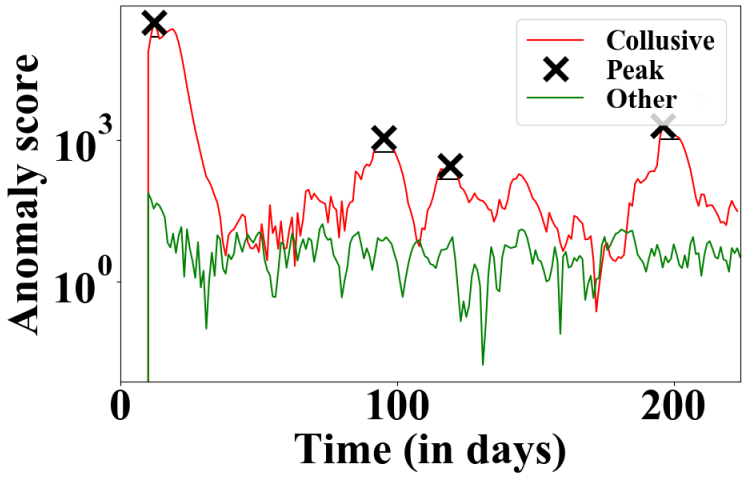


Figure 8.1: An example of anomalous pattern in videos detected by `CollATe` for collusive (in red) and other videos (in green). The x-axis displays the time span (in days) starting from when the first comment was posted and y-axis determines an anomaly score calculated by `CollATe`. Here, the anomaly score is the Mahalanobis distance computed using Equation 8.2 as mentioned in Section 8.6.3.1. The peak width (horizontal black line) corresponding to every peak indicates the duration of the peak.

peaks in the inter-arrival time of appraisals of the collusive entities; however, normal users do not exhibit such behavior.

8.3.2.0.2 Why only YouTube? In this article, our focus is to detect collusive entities only on YouTube. The primary reason of conducting this study in one platform is due to the challenge in collecting a large dataset of videos and channels hosted on other platforms and submitted to blackmarket services for collusive appraisals. While conducting this study, we checked a few freemium blackmarket services which provide collusive appraisals to video-sharing platforms such as Twitch and Vimeo. During the collection of information from the blackmarket services, we observed that a very less data is available for these platforms. Additionally, unlike YouTube, the APIs of these platforms have certain limitations and do not provide enough public information which can be used for detailed analysis of the collected entities.

8.3.2.0.3 Challenges in collusive entity detection. Detecting collusive entities is a difficult task [3, 5]. There exist blackmarket services which have created intelligent mechanisms to produce collusive appraisals. We list down the unique challenges of the collusive entity detection task below:

C1. Collusive accounts are not full-time bots, spam-accounts or fake accounts used only for appraisal. Thus, existing studies focusing on bot, spam and fake account detection are not able to capture their behavior as shown in our previous studies [3, 5].

C2. Collusive accounts do not show a completely genuine behavior. Some of their appraised content would be endorsed not because of their interest in the content, but merely to comply with the barter system in blackmarket services. This implies that collusive users show an amalgamation of both organic and inorganic behavior – being genuine users, they organically appraise the content of their interest; being a blackmarket members, they also inorganically appraise the content posted in blackmarket services.

C3. Limited contextual information is available on short texts present in collusive entities such as comments, replies, etc., which makes it difficult for deep neural networks to generate appropriate representations.

Table 8.1: Summary of the dataset. Here, the column *# unique CC* refers to the number of unique content creators of videos submitted for collusive likes and comments. *Entities* refers to *videos* or *channels*; *actions* refers to *like/comment* for videos and *subscription* for channels.

Type	# entities	# deleted	# verified	# unique CC	Max actions	Min actions	Avg. actions
V_l	45572	15662	342	28702	3151770	0	1333
V_c	25106	1060	86	11752	1008428	0	120
V_s	7847	0	7	–	12935205	0	5378

8.4 Dataset description

8.4.1 Data collection

The major challenge is to collect a large set of YouTube videos and channels submitted to the blackmarket services for collusive appraisals and a contrasting set of videos. We started our data collection by designing multiple web scrapers for the following purposes – (i) scraping data from blackmarket website (YouLikeHits) and (ii) scraping video related data (i.e., description, comments) of YouTube videos. Both the scrapers performed their operation independently. We used Joblib[6] library to utilize multiple cores in the deployed server. We also extensively used YouTube API[7] for the following purposes – (i) extracting subscribers of YouTube channels, and (ii) extracting the exact time at which the comments are posted.

We collected the information of collusive videos from YouLikeHits, a blackmarket service that provides three types of collusive appraisals – (i) likes to YouTube videos, (ii) comments to YouTube videos, and (iii) subscriptions to YouTube channels. We queried multiple search engines with keywords such as 'free YouTube likes', 'free YouTube comments'. Interestingly, apart from links to the websites providing artificial YouTube views and subscriptions, we found a large number of blackmarket websites pointing to other online media platforms such as Twitter, Facebook, Instagram. It indicates the popularity of blackmarket websites to achieve artificial social status in a much rapid way among online media users. We developed a scraper to retrieve YouTube entities (videos/channels) involved in collusive appraisals on YouLikeHits. Interestingly, we observe that some of the collusive videos are propagated by verified YouTube channels. We found 342 (*resp.* 86) videos submitted to YouLikeHits for collusive likes (*resp.* comments). We also found 7 verified YouTube channels, which are registered in YouLikeHits for collusive subscriptions. While scraping, we found that only 25% of videos for collusive likes and 4.05% of videos for collusive comments are deleted by YouTube's current fraud detection system. All the above observations show that YouTube is unable to detect these entities effectively using its in-house fraud detection mechanism. These insights further motivated us to design efficient methods to detect YouTube entities that are involved in gaining collusive appraisals with the help of blackmarket services.

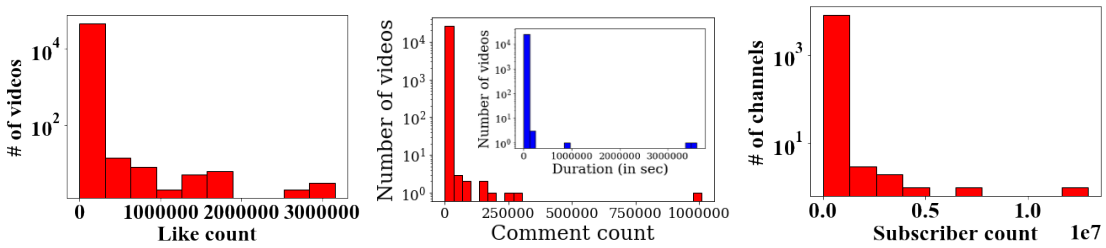
We extracted video metadata and comments of all the videos using our custom-designed scrapers and YouTube API. Finally, we divided the dataset into three unique sets – V_l (videos submitted to YouLikeHits for collusive likes), V_c (videos submitted to YouLikeHits for collusive comments) and V_s (channels submitted to YouLikeHits for collusive subscriptions). The entire data statistics is showed in Table 8.1 and Fig. 8.2. Note that we did not find any videos which are common across V_l and V_c.

8.4.2 Data privacy

To our knowledge, this work is the first effort that aims to analyze YouTube videos and channels submitted to blackmarket services for collusive appraisals. We emphasize that we will not release the sensitive information (i.e., video ids and uploader details) when we make the dataset public. The entire data collection process was performed after taking proper Institutional Review Board (IRB) approval from our institute.

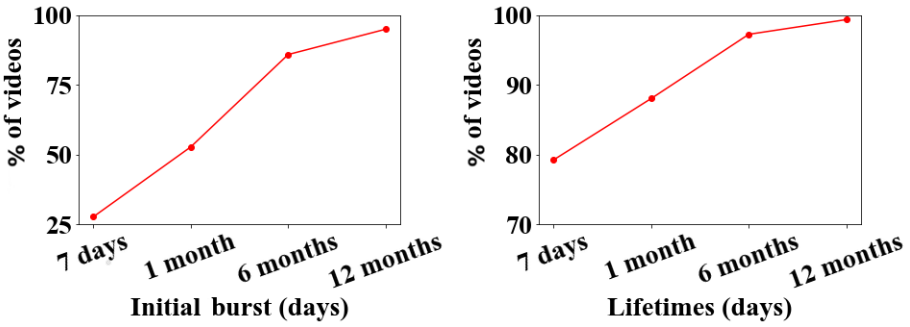
[6]https://joblib.readthedocs.io/en/latest/

[7]https://developers.google.com/youtube/v3/



(a) (b) (c)

Figure 8.2: Distribution of (a) likes of YouTube videos, (b) comments of YouTube videos, and (c) subscribers of YouTube channels in our dataset.



(a) (b)

Figure 8.3: Distribution of temporal properties characterizing propagation dynamics – (a) initial delays and (b) lifetimes of YouTube collusive videos.

8.5 Analyzing collusive YouTube entities

We analyze the collusive videos from two aspects - (i) propagation dynamics, and (ii) video metadata. In the first part of this section, we will present the propagation dynamics of collusive videos using two metrics – *initial burst* and *lifetimes*. In the second part, we will show the analysis of the collusive YouTube channels based on location, channel metadata and network structure.

8.5.1 Videos submitted for collusive comments/likes

For videos submitted for collusive comments, we extract the video metadata and video comments (full text of each comment and timestamp at which the comment was posted). Note that due to the restrictions of YouTube API, we are unable to provide detailed insights of the videos submitted to blackmarket websites for collusive likes. The API only allows retrieving the total count of likes and dislikes.

8.5.1.1 Propagation dynamics of artificial boosting

Here we focus on the temporal properties of collusive videos based on two features – *initial burst* and *lifetimes*. As the timestamps of occurrence of like activities are not available with us due to API restrictions, we perform the analysis only on videos submitted to blackmarkets for collusive comments.
(i) **Initial burst:** We consider the initial burst as the first time when there is a peak in the arrival rate of comments in a video. Section 8.6.3.1 outlines the peak detection technique. Through this analysis, we characterize how rapidly a video receives collusive appraisals. Fig. 8.3(a) shows that around 30% of the videos have an initial burst of artificial boosting (first peak) within 7 days, and around 50% of the videos have an initial burst within the first one month

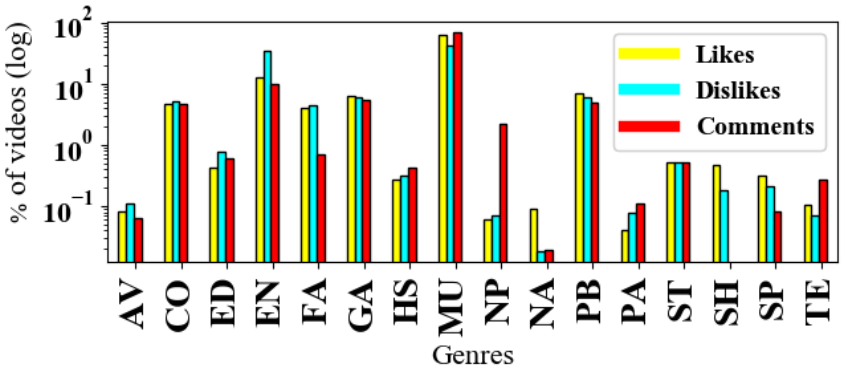


Figure 8.4: Genre-wise distribution of likes, dislikes and comments of collusive videos. Full forms of the labels in the x-axis are mentioned in Section 8.5.1.2.

(a) (b)

Figure 8.5: Wordcloud of titles of videos submitted for (a) collusive comments and (b) collusive likes.

from the date of posting.

(ii) **Lifetimes:** Here we consider the lifetime of the collusive activity over a video. We calculate the delay between the burst of the first peak and the fall of the last peak. Fig. 8.3(b) shows that about 80% of the videos have lifetimes within 7 days. This illustrates how these videos gain rapid attention through blackmarket services.

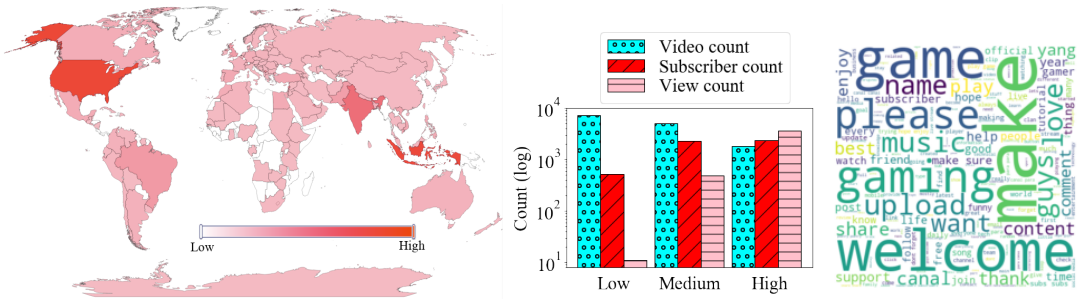
8.5.1.2 Metadata of collusive videos

Here we analyze the metadata of collusive videos.

(i) Video genres: We observe the distribution of genre of videos submitted to blackmarket services. The common genres on YouTube are – Gaming (GA), Entertainment (EN), Travel & Events (TE), Film & Animation (FA), Music (MU), People & Blogs (PB), Autos & Vehicles (AV), Education (ED), Comedy (CO), News & Politics (NP), Howto & Style (HS), Science & Technology (ST), Sports (SP), Pets & Animals (PA) and Nonprofits & Activism (NA). We plot the genre-wise distribution of likes, dislikes and comments of collusive videos in Fig. 8.4. The bars for likes and dislikes are drawn from the video metadata for collusive likes, and bars for comments are drawn from the video metadata for collusive comments. As expected, we observe 'Music' to be the most popular genre for videos submitted for collusive likes (63.05%) and comments (69.81%).

(ii) Wordcloud of video title: We show the wordcloud generated from the title of the videos submitted to blackmarkets for collusive likes and comments in Fig. 8.5. For clarity, we remove the two-letter words and common stopwords. Here the font size corresponds to the frequency of the text. We clearly observe the presence of similar keywords such as promotional keywords like 'free', 'best', 'top', etc. in both the cases. With the presence of these keywords, it is evident that videos for collusive like/comment appraisals focus on target-specific keywords for quick promotion.

(iii) Uploader authenticity: We also study the authenticity of the uploader of videos for collusive comments and likes. Surprisingly, we observe that verified users (marked by YouTube) are also involved in gaining collusive appraisals via blackmarket services (see Table 8.1).



(a) (b) (c)

Figure 8.6: (a) Country-wise distribution of collusive YouTube channels, (b) distribution of YouTube channels based on video, subscriber and views, and (c) wordcloud of channel titles.

8.5.2 Channels submitted for collusive subscription requests

In our analysis thus far, we focused only on YouTube video submitted to blackmarket services for collusive likes and comments. In this section, we examine YouTube channels submitted to blackmarket services for collusive subscriptions.

(i) Country-wise distribution: Fig. 8.6(a) shows the world map plot for the country-wise distribution of collusive YouTube channels (in red color), indicating the count of channels submitted to blackmarket services for collusive subscriptions. Out of $7,847$ collusive channels, we find $3,804$ channels with no countries mentioned in the profile. In the remaining set, USA tops the list with 20.11% of collusive YouTube channels followed by Indonesia with 19.58%.

(ii) Metadata of collusive YouTube channels: We show the video, view and subscriber count of collusive YouTube channels. For better visualization of the distributions, we create custom range of the counts – Low ($1 < count < 100$), Medium ($100 <= count <= 1000$) and High (> 1000). Fig. 8.6(b) shows the distribution of video, subscriber and view count for YouTube channels submitted for collusive subscriptions. We observe that video count and subscriber count are comparable across all ranges; however, there are too many channels with high view count registered in blackmarkets for collusive subscriptions. Fig. 8.6(c) shows the wordcloud aggregated over channel titles. We eliminate common stop words and two-letter word. Here also, the font size corresponds to the frequency of the text. We observe that collusive channels have keywords such as 'game', 'tutorial', 'music', etc., corresponding to personal interests, which help them improve the outreach by connecting with the viewers and subscribers within the same domains.

(iii) Network observations: We study the network structure of collusive channels using social network analysis tools. First, we create an undirected network – nodes are the collusive YouTube channels, and an edge represents common subscribers among two channels. We use the YouTube API to collect the subscribers of each channel. Out of $7,847$ channels, we observe only $2,721$ channels with public subscriptions (i.e., end-users can see what channels they are subscribed to). This forms the node-set in the network. We obtain $2,650$ edges among these nodes. We obtain 168 different connected components from this graph. We consider the final graph to be the maximum connected component (giant component) with $1,320$ nodes and $1,396$ edges for our analysis. We show various network properties (see Table 8.2) to better understand the structure of the giant component. We notice that a large fraction of nodes (48.51%) belong to the largest connected component, which perhaps indicates the barter system in freemium services. The average shortest path length of the network is 9.037, which is of the order of the comparable random network of the same size and average degree. Note that the comparable random network is created using the same number of nodes and edges as that of the collusive YouTube channel graph but with random edge connections. The average clustering coefficient of the giant component in the collusive channel network is 0.0023, which is an order of magnitude higher than that of the comparable random network whose clustering coefficient is 0.0016. These two structural properties of the giant component indicate that the network of collusive YouTube channels is a small-world [203].

Table 8.2: Statistics of collusive YouTube channel network.

# nodes	# edges	AD*	Diameter	APL**	Density
1320	1396	2.115	19	9.037	0.0016

* - Average degree. ** - Average path length.

Table 8.3: Notations and denotations.

Notation	Denotation
TS	Time sequence vector of comments
$c^{(i)}$	Cumulative comment count at each timestamp
e	Error vector
$\mathcal{N}(\mu, \eta)$	Gaussian distribution using the error vectors
$a(c)$	Anomaly score calculated using Mahalanobis distance
η	Comment similarity score
P	Number of peaks
W	Number of windows
q	Query comment
C	Window comments
$g(q)$	Embedding for the query comments
$g(C)$	Embedding for the window comments
$x \in X$	Input data point x in input space X
V	Intermediate decoded space
Z	Intermediate encoding
$y \in Y$	Output label y in labels space Y
$\hat{x}$	Noisy or corrupted input data point
τ	Encoder present in the autoencoder
ψ	Decoder present in the autoencoder

Table 8.4: Abbreviations used throughout the study.

Abbreviation	Description
MFE	Metadata Feature Extractor
AFE	Anomaly Feature Extractor
CFE	Comment Feature Extractor
DAC	Denoising Autoencoder Classifier
$ARIMA$	Auto Regressive Integrated Moving Average
WMD	Word Mover Distance

8.6 Detecting collusive entities

In the previous section, we have discussed how collusive videos and channels have an adverse effect on YouTube social space. Therefore, the question we would like to answer is – *can we automatically detect if an entity on YouTube is submitted for collusive appraisals*?

In this section, we discuss the methodology for the detection of videos and channels submitted to blackmarket services for collusive likes, comments, and subscriptions. First, we will show how we detect videos for collusive likes, and channels for collusive subscriptions. We pose this problem as a one-class classification problem as the set of genuine videos is unknown to us[8]. We utilize several one-class classifiers with the proposed sets of static features for the detection. Secondly, we develop a novel end-to-end framework, named `CollATe` that leverages video metadata, anomalous activities and comment similarity for detecting videos for collusive comments. Table 8.3 summarizes the notations with their denotations and Table 8.4 summarizes the abbreviation and description used in the model.

8.6.0.0.1 Reason behind proposing three collusive entity detection tasks. YouTube has three types of appraisals: *likes, comments, subscriptions* for two types of entities: *videos* and *channels*. Likes and comments are for videos, and subscriptions are for channels. In case of collusion, the blackmarket services provide appraisals in such a way that one kind of appraisal is not related to other e.g., if a customer requests for likes on a YouTube video, the video will only receive likes and not any other appraisal. This is also seen in case of Twitter [5] where there are two types of appraisals, *followers* and *retweets*, for two types of entities, *users* and *tweets*, respectively. As each

[8]One can argue that YouTube verified channels/videos might constitute the genuine set. We did not consider them because of the following reasons. (i) In our dataset, we found 7 verified channels which were submitted to the blackmarkets. Therefore, we were unsure whether all verified entities are really genuine or not. (ii) Verified channels/videos are usually very popular, receiving huge appraisals from the viewers. Therefore, they may not look like a normal, random YouTube entity and may lead to bias in the dataset.

appraisal is independent of other appraisals, we proposed three detection tasks and did not combine them to just one task.

8.6.0.0.2 Reason behind proposing two one-class models and one binary classification model. The reason behind not framing the first two models as a binary classification task is the unavailability of textual and temporal information for these two models. In literature, it has been seen that collusive accounts show a hybrid behavior – they sometimes behave like a genuine accounts and exhibit organic activities; at the same time, they inorganically appraise the content of other accounts to gain credits from blackmarket services. This made us to hypothesize that collusive accounts tend to have very diverse topics of interest as they appraise content without their genuine interest (textual information) and show sudden changes in the commenting activity (temporal information). In our case, the appraisal present in the third task (i.e., collusive comments for YouTube videos) is an interactive-based appraisal where a user posts a textual comment on a specific time for a video. However, appraisals present in the first two tasks, i.e., collusive likes for YouTube videos and collusive subscriptions for YouTube channels, are tap-based appraisals where a user simply taps on the like/subscribe button. The only way to collect temporal information for these appraisals is using YouTube Live Streaming API which has a quota limitation and makes it infeasible in a real-time scenario where data comes in streaming fashion. Moreover, tap-based appraisals do not contain any textual information. Thus, we proposed the first two models as one-class classification models and the last model as a binary classification model.

8.6.1 Features for detecting videos submitted for collusive likes

In this section, we present novel features to characterize videos submitted to blackmarket services for collusive like requests. In Section 8.7.2, we show how these features affect the performance of different one-class classification models.

• **Activeness (α):** We observe that collusive videos generally receive more likes with a relatively small number of views. We compute the *Activeness* as the ratio of the total number of likes to the total number of views gained by the video. It shows the high engagement of content consumer on the basis of likes and views. Fig. 8.7(a) shows the distribution of activeness for collusive videos.

• **Favorability (β):** We notice that collusive videos receive less dislikes even for a large number of views[9]. We compute the *Favorability* as the ratio of dislikes to the sum of likes and dislikes gained by the video. It shows the likelihood of likes gained by the video. Fig. 8.7(b) shows the distribution of favorability for collusive videos.

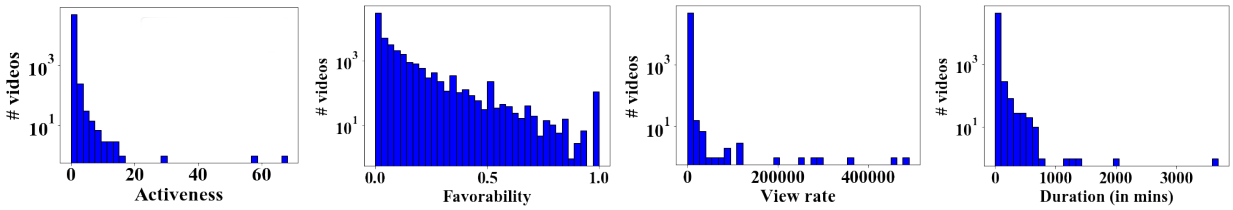
• **View-rate (γ):** The collusive users may focus on increasing the likes of the videos. However, with the growth in likes, view count will automatically increase (the reverse is not true though). This unintended growth in views also helps the content creator to gain complementary collusive views. We compute the *view-rate* as the ratio of the total number of views gained by the video to the total number of days since its submission to YouTubey. Fig. 8.7(c) shows the distribution of view-rate for collusive videos.

• **Video duration (δ):** We observe that YouTube videos for collusive comment requests are fairly short length videos with an average duration of 6 minutes. We consider the length of the video (in seconds) as one of the features for the collusive entity detection. Fig. 8.7(d) shows the distribution of video duration for collusive videos.

The final metadata feature extractor v_e has the following form:

$$v_e = (\alpha, \beta, \gamma, \delta) \tag{8.1}$$

[9]Note that the number of dislikes is not complementary of the number of likes, as users who view a video may not make any action (like/dislike). Therefore, it is not necessary that the sum of likes and dislikes of a video is the number of views.



(a) (b) (c) (d)

Figure 8.7: Exploratory analysis of features used to characterise videos submitted for collusive likes.

8.6.2 Features for detecting channels submitted for collusive subscriptions

In this section, we present the novel features to characterize channels submitted to blackmarket services for collusive subscriptions. Note that due to YouTube API restrictions, we are unable to collect granular details about the YouTube channels.

• **Hidden subscriber count:** It is the number of subscribers the channel receives which are hidden from its profile.

• **Video count:** It is the count of the total number of videos present in the channel.

• **Subscriber count:** It is the count of the total number of subscribers the channel has received.

• **View count:** It is the total number of views the channel has received.

• **Comment count:** It is the total number of comment the channel has received.

8.6.3 Detecting videos submitted for collusive comments

In this section, we present novel features to characterize videos submitted to blackmarket services for collusive comments. In Section 8.7.2, we will show the importance of these features. It is known that not all the comments posted on a video are collusive in nature. This makes it hard to identify collusive comments even for human experts. Our dataset contains rich social information about YouTube videos such as uploader details, video metadata, channel details, raw comments text, and comment timestamp. The goal of our proposed model is to transform the data into useful features and identify the collusive videos.

`CollATe` comprises of three components: (i) metadata feature extractor (v_m), (ii) anomaly feature extractor (v_a), and (iii) comment feature extractor (v_c) as shown in Fig. 8.8. Once v_m, v_a and v_c are derived, they are concatenated to create the final video representation. The collusive video detector takes the learned representations as input to predict if a video is collusive. The details of the metadata feature extractor is mentioned in Section 8.6.1 (we use the same metadata feature extractor except view-rate (γ)). The reason behind not choosing view-rate is that a popular YouTube video is likely to have similar view-rate to the collusive one. Two other components are mentioned below.

8.6.3.1 Anomaly feature extractor

The goal of the anomaly feature extractor is to detect sudden changes in the commenting activity over a video. Such activities generally continue for a shorter time duration and then stop suddenly. We would like to detect such activities for a given video to derive useful features. The anomaly feature extractor takes sequential time-series data of comments as input. To extract the anomalies (peaks) from the time-series data, we employ Gated Recurrent Units (GRU) as the core module of the anomaly feature extractor. We adapt the idea of stacking recurrent layers for anomaly detection from Malhotra et. al [202].

Overview: Consider a time sequence vector $TS = \{c^{(1)}, c^{(2)}c^{(3)}, \ldots c^{(n)}\}$ where every point $c^{(t)} \in R^m$ in

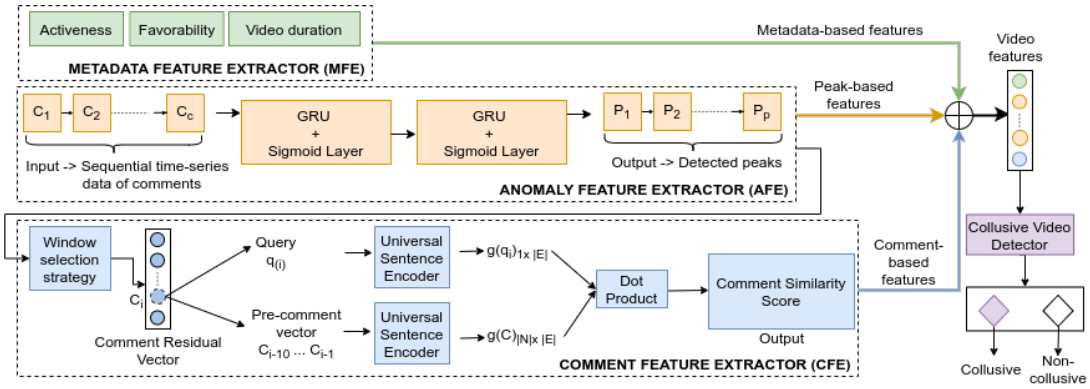


Figure 8.8: The architecture of `CollATe`. The green colored network is the metadata feature extractor, the orange colored network represents the anomaly feature extractor, and comment feature extractor is blue colored.

the time sequence is a m-dimensional vector $\{c_1^{(t)}, c_2^{(t)}, c_3^{(t)}, \ldots c_m^{(t)}\}$, denoting the cumulative comment count at each timestamp for a given video. Using the time sequence vector TS, we train a stacked GRU network. This model learns to predict the next l values for d of the input variables such that $1 \leq d \leq m$. We take m units in the input layer and $d \times l$ units in the output layer. The hidden layer GRU units are fully connected through recurrent connections. We stack the GRU layers in a manner that every unit in the lower GRU hidden layer is fully connected with every unit in the hidden layer above it through feed-forward connections.

We utilize the predicted values for computing the prediction error distribution using which we detect the unusual comment activities (peaks). The error vector $e^{(t)}$ is defined as $e^{(t)} = [e_{11}^{(t)}, \ldots, e_{1l}^{(t)}, \ldots, e_{d1}^{(t)}, \ldots, e_{dl}^{(t)}]$, where $e_{ij}^{(t)}$ is computed by taking the difference between the actual value of $c_i^{(t)}$ and its predicted value at time $t - j$. We fit a multivariate Gaussian distribution $\mathcal{N}(\mu, \Sigma)$ using the error vectors, where μ and Σ are estimated using Maximum Likelihood Estimation.

To label the observation as either collusive or non-collusive, we calculate the anomaly score $a(c)$, which is defined as the Mahalanobis distance between the computed error vector $e^{(t)}$ and the distribution $\mathcal{N}$.

$$a(c) = (c - \mu)^T \Sigma^{-1} (c - \mu) \tag{8.2}$$

Finally, we concatenate all the observations for a given video time sequence and feed the concatenated sequence to the *find_peaks* module of Scipy[10] to retrieve the peak width and peak height. Using the same data, we get two useful features that we pass into the collusive video detector.

(i) Peak count (ϕ): It is the count of the of peaks detected for a video.

(ii) Average peak area (ω): The average peak area is calculated as the average of the overall area covered by each peak. After exploratory data analysis, we observe a very clear distinction between the distribution of peak width and height of collusive and other random videos. The width and height of each peak is calculated using the *peak_width* module of Scipy[11].

Figs. 8.9(a) shows the peak count and average peak area for collusive videos.

The final anomaly feature extractor v_a has the following form:

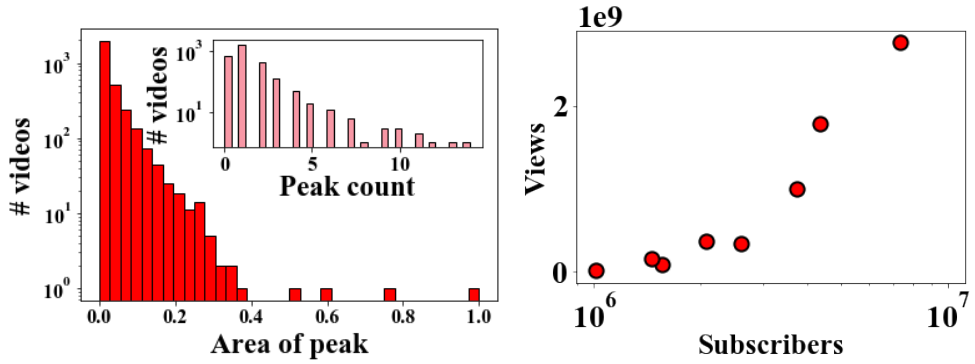
$$v_a = (\phi, \omega) \tag{8.3}$$

8.6.3.1.1 Training details:

As the idea here is to detect unusual commenting behaviour, we train GRU on $\boldsymbol{V_o}$ (more details on $\boldsymbol{V_o}$ can be found in Section 8.6.3.3) which is a set of random videos. Once trained, the model

[10] https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.find_peaks.html

[11] https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.peak_widths.html



(a) (b)

Figure 8.9: (a) Exploratory analysis of the features used for anomaly detector in case of collusive videos. (b) Linear pattern between subscribers and views of YouTube channels i.e., channels with more subscriptions tend to lead to more views.

will not be able to generate patterns similar to collusive videos resulting in higher valued error vector or in simple terms – an anomalous activity. In our experiments, we noticed that LSTM cell tends to generate multiple peaks for single activity while GRU was able to mitigate this issue well. For the same, we choose to employ GRU cells in place of LSTM cells [202]. Moreover, we could also reduce the model training time substantially using the GRU cells in place of the LSTM cells.

8.6.3.2 Comment feature extractor

YouTube users generally comment on videos if either they like/dislike the video or the domain of interest is the same. On the other hand, the collusive users are majorly interested in uplifting their credits just by posting a large number of comments. We observe that they tend to borrow the content from the recently posted comments, make marginal changes and post the same. We hypothesize that the comments posted during artificial boosting should have high textual similarity between the collusive comments.

To this end, we measure the similarity of closely related comments posted during the boosting timeline. We extract full comment texts of all the videos using the YouTube API. The data collection strategy is detailed in Section 8.4.1.

Window selection strategy: Computing a similarity score by utilizing all the previously posted comments for a given query comment might seem intuitive but is a computationally heavy task. For the same, we first retrieve the comments posted during peak time. Moreover, we propose to use a fixed-size (w) moving window and roll it over the set of retrieved comments. We define the last comment in each set as query comment (q_i) and the other comments as the window comments (C_i). The best performing window size (w) from the experiments was found to be of size 10.

Comment similarity score (η): To calculate comment similarity score, we first encode both the query comment q_i and the window comments C_i using Universal Sentence Encoder (USE) [204][12]. As we know that collusive users can belong to different geographical places and may not use the same language, we chose to work with the multilingual USE model[13] to retrieve comment embeddings. In the literature, USE based model has been able to achieve state-of-the-art for SemEval-2017 Task [205] on Semantic Textual Similarity Multilingual, and Cross-lingual Focused Evaluation. Currently, the multilingual model supports 16 different languages and has shown strong performance in cross-lingual text retrieval. The input to the model can be variable length text in any of the

[12]Models like BERT can be used to generate embeddings of comments. However, multilingual BERT (SentenceBERT) was not available during that the time of our study. We used USE (Universal Sentence Encoder) infor our study as it includes training on multiple tasks across languages (which is largely present in our YouTube comment data), and the compute time is linear in the length of the input sequence.

[13]https://tfhub.dev/google/universal-sentence-encoder-multilingual/1

supported languages and the output is a 512-dimensional vector. We transform each query comment into a fix-length embedding vector $g(q_i)$ and the window comments into set of vectors $g(C_i)$ using USE.

It has been observed that the same video was posted multiple times for the collusive comments. It may also happen that the collusive user ran out of credit points due to which the collusive video is no longer shown on the blackmarket website until the user again starts earning some credit points. In such cases, our peak detection strategy is likely to detect multiple peaks (p) for the videos (V). Moreover, a peak can have a large number of comments. We define the comment similarity score (η) for a given video as follows:

$$\eta = \frac{\sum_{i=1}^{P} P_{s(i)}}{P} \tag{8.4}$$

$$P_{s(i)} = \frac{\sum_{j=1}^{W} \max_{j:w_j \in P}(g(q_j) \cdot g(C_j)^T)}{W} \tag{8.5}$$

where $P_{s(i)}$ denotes comment similarity score for ith peak, P denotes the number of peaks, W denotes the number of windows, and $g(q_j)$ and $g(C_j)$ denote the jth query embedding and jth window embedding respectively. We choose the maximum of the dot product of query embedding and the sentence embedding under the assumption that one comment is derived from only one other comment.

We use the derived score η along with the total comment count (t_c) to create v_c. The final comment feature extractor v_c has the following form:

$$v_c = (\eta, t_c) \tag{8.6}$$

We then concatenate the learned representations from the metadata feature extractor v_e, anomaly detector v_a and comment feature extractor v_c to form the video feature representation denoted as $v = v_e \oplus v_a \oplus v_c$.

8.6.3.3 Collusive video detector

Here we introduce the collusive video detector, as shown in Fig. 8.10. The primary purpose of the detector is to learn the distribution from the collusive data and identify similar ones. Although this task looks very similar to the tasks mentioned in Sections 8.6.1 and 8.6.2, it actually uses a different kind of features by incorporating the temporal representation of the comments. To investigate this, we manually collected a small set of YouTube channels that are not posted on YouLikeHits. We denote this set of videos by $\boldsymbol{V_o}$. Here we only considered collecting the other set of videos ($\boldsymbol{V_o}$) from the top 5000 channels because YouTube channels with more subscriptions tend to lead to more views, thus resembling the collusive channels. Although it is not possible to find out view-timestamp from the YouTube API for any given video, we can still expect the overall behavior of $\boldsymbol{V_o}$ to be similar to Figure 8.9(b) as a user-driven blackmarket service is analogous to a channel in a few ways such as both benefit from more subscribers, the subscribers are loyal and tend to watch/like/comment on the videos posted. We selected a playlist from each channel at random. From each playlist, we randomly selected *k* (where, *k*=3 in our case) videos. As the overall data was biased towards some specific genres, we removed some videos at random to maintain generality in data. The presence of such similar patterns in both the classes increases the difficulty of the classification task even further but will make the classifier more relevant for real-world collusion detection. Here, $\boldsymbol{V_o}$ contains videos having similar comment growth patterns and thus can be considered a noisy/adversarial set in terms of comment growth patterns. Note that it does not represent the entire YouTube population. For the same, we report the true positive rate (TPR) of the model with respect to the set of collusive videos.

Training a fully connected network that directly optimizes only the supervised objective by gradient descent also does not work very well in such cases. What works better is to initially use a local unsupervised criterion to pre-train each layer, with the goal of learning to produce a useful higher-level representation from the lower-level representation output by the previous layer. From this initial point, gradient descent on the supervised objective leads to better solutions in terms of generalization performance [206, 207].

We propose to use Denoising Autoencoder Classifier (DAC), which uses autoencoders, a deep unsupervised learning method that improves the generalization of supervised learning on limited labeled data. In particular,

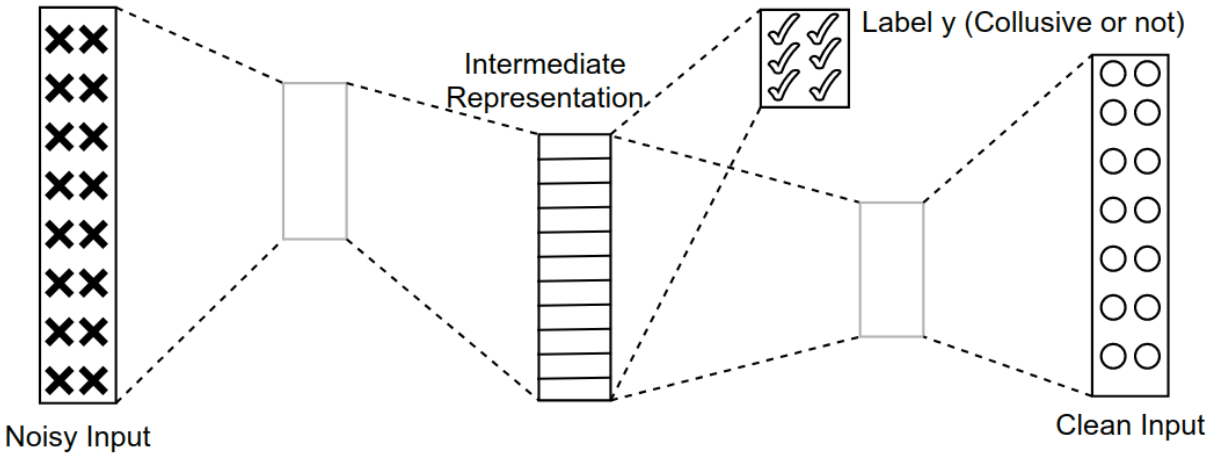


Figure 8.10: The architecture of Denoising Autoencoder Classifier.

we investigate a case where the input signal is noisy and we explore the multi-task learning with two tasks: a classification output, coupled with a reconstruction output of a Denoising Autoencoder. To establish the effectiveness of this approach, we also present a comparison of this approach against different baselines, including a multi-layer perceptron (MLP) model as a replacement of the Denoising Autoencoder (more details in Table 8.6).

Throughout this section, we define the input space as *X*, input data point $x \in X$, label space as *Y*, output label $y \in Y$, intermediate decoded space *V*, intermediate encoding $z \in Z$, noisy or corrupted input data point $\hat{x} \in X$.

At a high level, DAC consists of two structural components: the denoising autoencoder, and the dense layer as the classifier. Denoising autoencoder has a very similar structure to the autoencoders, except rather than reducing only the reconstruction loss over the input data, it maps noisy inputs to original clean inputs. Autoencoders consist of two components: an encoder $(\tau : X \implies Z)$ and a decoder $(\psi : Z \implies X)$ where τ and ψ are inversely related. They first map an input $x \in X \in \mathbb{R}^d$ to an intermediate representation $z \in Z \in \mathbb{R}^{d'}$ of reduced dimensionality $(d' < d)$ via τ. This intermediate representation is then reconstructed back to a point x' in the input space, and the difference between x and x' is measured using a loss function L.

On the contrary, in case of denoising autoencoder, a noisy representation of x, $\hat{x}$, is fed to the network yielding $\hat{x}'$, and then $\hat{x}'$ is compared to the original clean input data point x. The classifier utilizes the intermediate representation z of the denoising autoencoder. After encoding noisy input, the classification task is added that contributes to the encoder weights independently from the decoder. This task consists of a fully-connected layer that predicts the label $y' \in Y$. Finally, both the classification loss and reconstruction loss propagate backward and contribute to encoder weights.

Training details: We consider a fully-connected autoencoder with two hidden layers containing 128 units each for all the experiments. We use mean squared error as the loss function difference between x and $\hat{x}'$. To generate corruption in training data, we randomly corrupt the input data by $\pm 10\%$ of its original value. We use the entire unlabeled training set, i.e., 1384 videos – 756 collusive and 628 random videos, to train the denoising autoencoder for 25 epochs and minimize mean squared error (MSE) from the reconstruction output. During this stage, we do not update the weights for edges between the encoded layer and classification output. The denoising autoencoder classifier is then trained to minimize the categorical cross-entropy loss on the labeled training samples for 150 epochs. Note that we choose the optimal value of each parameter based on the hyperparameter search. We implement our model using Python 3.8.3 and TensorFlow 2.3.1. The model has a total of $18k$ trainable parameters and is less than $100KB$ in size.

8.7 Experimental results

8.7.1 Baselines for detecting videos submitted for collusive comments

We validate the performance and robustness of `CollATe` by comparing with seven baselines. Since there is no existing research dealing with the same problem present in this work, we develop a few of our own baselines and adopt some of the existing studies to our problem setting. We report the performance of all the competing methods along with the detailed analysis of the performance and robustness.

- **Extractor models:** `CollATe` uses the combination of metadata, anomaly and comment feature extractors. Thus, we propose two baselines: (i) considering only one extractor at a time (B_1), and (ii) considering two extractors at a time (B_2). This also provides the ablation study of our method.
- **Variant of proposed model (B_3):** As shown in Fig. 8.8, comment feature extractor and anomaly feature extractor can be modified to obtain a variant of `CollATe`. We modify both the components to obtain different variants of the proposed model as described below:
  LSTM-based Anomaly Extractor: We replace LSTM units in place of GRU for detecting the anomalous comments. We observe that this variant tends to generate multiple peaks for the videos with only one-time collusive comments. This change not only adds the noisy comments but also deteriorates the overall performance significantly as the comment feature extractor is dependent on this stage.
  WMD-based Comment Similarity Score: Word Mover's Distance (WMD) [208] is a popular metric for calculating semantic similarity at a document level. We utilize standard pre-trained Word2Vec embeddings from Gensim library[14] to compute the Word Mover's Distance d_{ij}. We transform the distance d_{ij} into a similarity score s_{ij} as below:
  $$s_{ij} = \frac{1}{1 + d_{ij}} \tag{8.7}$$
  where i denotes the comment index, and j denotes the window index. Rest of the details for computing η remains the same as mentioned in Section 8.6.3.
  There are two major differences between this model and `CollATe` – (i) word-level vs. sentence-level embeddings, and (ii) monolingual vs. multilingual support. We discuss the impact of these differences through our experiment results in the next section.
- **DetectPV (B_4):** We use the method proposed by Bulakh et al.[209] as our fourth baseline. It uses a supervised learning approach to identify fraudulently promoted videos by extracting features from the video metadata. We wanted to check if video metadata-based fraudulent video detection techniques would be useful to detect collusive videos on YouTube.
- **ARIMA (B_5):** Since `CollATe` uses temporal information of comments posted on videos to detect peaks and their associated properties, one may argue that a time series based anomaly detection method may be able to detect collusive videos. To this end, we consider the method proposed by [210] as another baseline. ARIMA is an auto-regressive integrated moving average model that uses the combination of auto-regression and moving average to detect anomalies on time-series data.
- **CNN (B_6):** Feature interaction-based prediction models such as convolutional neural network (CNN) models have been used extensively for various image processing tasks [211, 212]. In our case, we use 1-D CNN layer, followed by dense layers to form a fully-connected neural network. The model learns to extract and map features from the data to the output labels.
- **WND (B_7):** We use the Wide & Deep Factorization (WND) method proposed in Guo et al.[213] as another baseline. The model consists of a wide part (to memorize the past behavior) and deep part (to embed into lower dimension) and utilizes deep neural network and factorization machines to address the interactions among the features.

[14] We tried with glove-twitter-200 and word2vec-google-news-300 embeddings and found the latter to perform better.

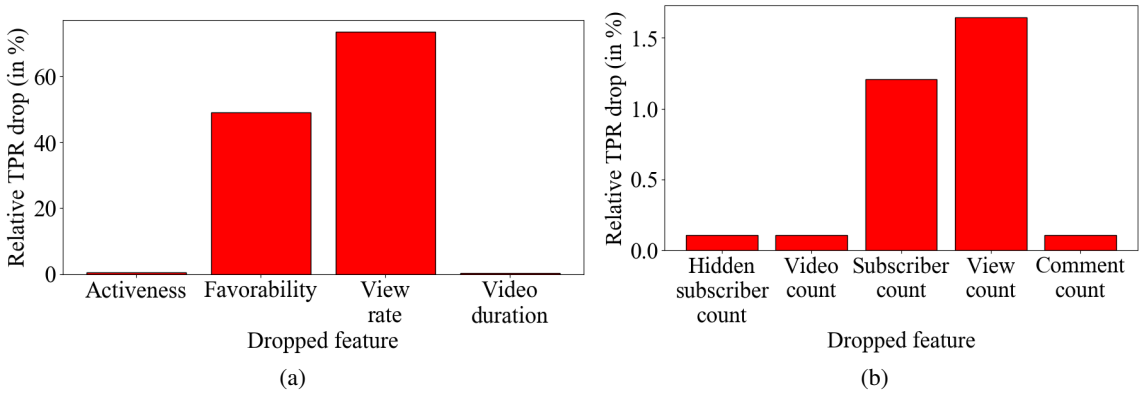


Figure 8.11: Feature importance considering (n-1) features at a time, i.e., dropping each feature in isolation (a) for Task 1 and (b) Task 2.

Table 8.5: Performance of one-class classification models on two tasks. Task 1: videos submitted for collusive likes; Task 2: channels submitted for collusive subscriptions.

Model	Task	True positive rate
One-class SVM	Task 1	**0.911**
	Task 2	**0.910**
Isolation forests	Task 1	0.902
	Task 2	0.906
Minimum covariance determinant	Task 1	0.900
	Task 2	0.901
Local outlier factor	Task 1	0.899
	Task 2	0.894

8.7.2 Prediction results

Tasks 1 and 2: Identify the videos (channels) submitted to blackmarket services for collusive likes (subscriptions).
Due to the restriction of the YouTube Data API, we are unable to access the timestamp of like/dislike activity. This resists us to create any time-centric features for collusive like appraisals. Similarly, we are unable to get detailed information about YouTube channels from the API. For our prediction task, we use one-class classification models with the features mentioned in Sections 8.6.1 and 8.6.2. The models are trained on one class, which in our case is the collusive class. Note that we only report the *true positive rate* (TPR) for our prediction task as we are only interested in the proportion of actual positives that are correctly identified by the model. We perform the prediction task with the following state-of-the-art one-class classifiers: one-class SVM [214], isolation forests [215], minimum covariance determinant [216], and local outlier factor [217]. All the scores are reported after 5-fold cross validation. Note that the test set remains same across all the competing methods.

Table 8.5 reports the performance of the one-class classification models for detecting videos submitted for collusive likes (Task 1) and channels submitted for collusive subscriptions (Task 2). In the former task, we observe the best accuracy of the model (with true positive rate of 0.911) with one-class SVM. To analyze the influence of each feature, we perform experiments, taking $(n-1)$ features at a time, i.e., dropping each feature in isolation. The most important feature turns out to be *view rate* based on quantified relative importance. In the latter task, we once again observe that one-class SVM performs the best (with true positive rate of 0.910). *View count* turns out to be the best feature for this task. Fig. 8.11 shows the feature importance for Task 1 and Task 2 with the relative TPR drop percentage when we drop each feature in isolation.

Task 3: Identify the videos submitted to blackmarket services for collusive comments.

Our third task is to determine whether a given video was submitted to blackmarket services for collusive comments. We use the approach explained in Section 8.6.3 to detect if a video is collusive. Table 8.6 shows that DAC consistently achieves better accuracy compared to its baselines. In Table 8.6, we mark the best performing extractor for each of the baselines, B_1, B_2 and B_3 in bold.

Here, CFE-USE denotes the usage of Universal Sentence Encoder while CFE-WMD denotes the usage of Word Mover's distance for calculating sentence similarity. CFE-USE (*comment similarity* and *comment count*) with DAC is individually the best performing extractor for baseline B_1. Although we expect AFE to perform better than at least MFE, due to the nature of the non-collusive video data, it does not. These results support our argument about the higher similarity in the distribution for both collusive and non-collusive datasets. For the same, we take into account multiple extractors that not only improvise the performance but also increase the robustness. B_2 (MFE + CFE-USE) with DAC outperforms other permutations as it captures the most useful representation for classification.

B_3 (*CFE-WMD*) seems to perform better than a few baselines but is unable to outperform `CollATe`. This is expected because WMD-based similarity score calculation depends on word-level embeddings rather than sentence-level embeddings. Moreover, we also find that the feature importance is evenly distributed in `CollATe` compared to B_3. [218] also suggested that the weight vector of a robust classifier should be distributed as evenly as possible. Finally, we report the baseline performance of B_4 (DetectPV), B_5 (ARIMA), B_6 (CNN) and B_7 (WND). We observe that the overall performance of `CollATe` is better than the baselines.

Table 8.6: Performance comparison of `CollATe` with baselines for detecting videos submitted for collusive comments.

Method	**TPR**	**FPR**	**Accuracy**	**AUC**
B_1 (AFE)	0.715	0.455	0.638	0.629
B_1 (CFE-USE) (MLP)	0.789	0.410	0.718	0.689
B_1 (MFE)	0.825	0.444	0.702	0.690
B_1 **(CFE-USE) (DAC)**	**0.866**	0.157	0.845	0.834
B_2 (MFE + CFE-USE) (MLP)	0.792	0.395	0.704	0.698
B_2 (MFE + AFE)	0.795	0.412	0.701	0.692
B_2 (AFE + CFE-USE) (DAC)	0.875	0.177	0.839	0.848
B_2 **(MFE + CFE-USE) (DAC)**	0.881	0.179	0.853	0.851
B_3 **(AFE + MFE + CFE-WMD) (DAC)**	**0.866**	0.182	0.844	0.841
B_4 (DetectPV)	0.829	0.232	0.802	0.798
B_5 (ARIMA)	0.768	0.429	0.677	0.669
B_6 (CNN)	0.679	0.491	0.609	0.594
B_7 (WND)	0.890	0.250	0.825	0.820
`CollATe`	**0.905**	0.194	**0.860**	**0.855**

8.8 Interesting observations

In this section, we detail the important implications of collusive entity detection task. To this end, we consider only the top 10% of the collusive entities detected (true positive) using our models to present our analysis. For tasks 1 and 2, the top 10% is generated using the scoring function[15] of our best-performing one-classification model. For task 3, the top 10% is generated using the softmax values present in the last layer of `CollATe`. We use the term *highly collusive* to refer to the collusive entities present in the top 10% in each of the cases.

OBSERVATION 1 (**Highly collusive videos have high video ratings):** We define rating as the ratio of likes to the sum of likes and dislikes gained by the video – the video will have rating 1 if there are no dislikes. We observe

[15] https://scikit-learn.org/stable/modules/generated/sklearn.svm.OneClassSVM.html#sklearn.svm.OneClassSVM.score_samples

that highly collusive videos have very high ratings with an average rating of 0.931. This shows how blackmarket services have been able to gain collusive appraisals in an effective way.

OBSERVATION 2 (**Highly collusive videos have very short video duration):** We note that highly collusive videos have a very short duration, with average video length of only ~4 minutes. As confirmed by YouTube[16], it itself promotes videos that keep people on YouTube for a long period of time. Thus, videos with short duration do not get the proper audience naturally, thereby making the authors choose blackmarket services to gain artificial appraisals in a quicker way.

OBSERVATION 3 (**Gaining collusive likes does not guarantee to gain collusive comments):** We observe that the videos submitted for collusive likes do not have many comments in them, with the average number of comments being only 12.4. This also corroborates with the fact that we do not have any intersection between the sets V_l and V_c. We can ensure that collusive like requests and collusive comment requests are two completely independent activities on the blackmarket platforms.

OBSERVATION 4 (**Highly collusive channels are popular channels with a large number of videos):** We note that highly collusive channels are popular YouTube channels with average subscribers count of $39,477$, average view count of $8,762,188$, and average videos count of 111. The reason is that getting new subscribers for those channels is an extremely difficult task, which makes them choose artificial ways of gaining new subscribers by means of the blackmarket services.

OBSERVATION 5 (**Highly collusive videos have a moderate inter-arrival rate of comments):** We study the inter-arrival rate of comments in the highly collusive videos. Surprisingly, we observe a mean inter-arrival rate of ~5 hours for each comment. The possible reason behind such high value of inter-arrival rate is due to the expiration of credits for the collusive comments in the blackmarket services.

OBSERVATION 6 (**Highly collusive videos have a moderate inter-arrival rate of comments):** We study the inter-arrival rate of comments in the highly collusive videos. Surprisingly, we observe a mean inter-arrival rate of ~5 hours for each comment. The possible reason behind such high value of inter-arrival rate is due to the expiration of credits for the collusive comments in the blackmarket services.

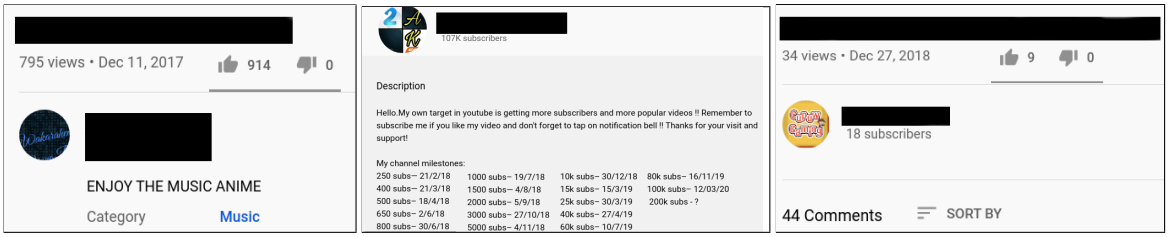
OBSERVATION 7: (**'People & Blogs' is the most popular genre for highly collusive channels**) We observe 'People & Blogs' to be the most popular genre (33.53%) for highly collusive channels. This genre usually contains YouTube bloggers who upload original content, and share it all with friends, family, and the world on YouTube. This is due to the reason that blogging in YouTube is currently considered as the most profitable way to earn money[17]. Thus, to gain quick popularity and recognition in the community, these users join blackmarket services. The second most popular genre for highly collusive channels is 'Music' (22.60%).

OBSERVATION 8: (**Highly collusive videos are family-friendly and non-paid unlisted videos**) We observe that highly collusive videos are family-friendly and non-paid videos. The possible reason behind not submitting violent and mature YouTube videos to blackmarket services may be due to the reason that YouTube does not allow age-restricted videos to be monetized. The possible reason behind not submitting paid videos to blackmarket services is that the monetization of YouTube videos is determined by the level of engagement (likes/comments) a video generates.

Fig. 8.12 shows the snapshot of some collusive entities detected using our models. Fig. 8.12(a) shows a collusive YouTube video where the number of views is much lesser than the number of likes. This clearly gives us the indication that the YouTuber is buying artificial likes from the blackmarket websites. Fig. 8.12(b) shows a collusive YouTube channel where the description clearly states that the author is looking for quick popularity. The YouTube is maintaining a timeline of the increase in the subscriber count of the channel. Also, the channel description contains promotional keywords such as 'gaining more subscribers'. Fig. 8.12(c) shows a collusive YouTube video where the video has more number of comments than the views and likes. This also gives us the indication that the YouTuber is buying artificial comments from the blackmarket websites.

[16] https://youtube-creators.googleblog.com/2012/08/youtube-now-why-we-focus-on-watch-time.html

[17] https://medium.com/@KeywordsHeaven/blogging-vs-youtube-72803bb3dacf



(a) (b) (c)

Figure 8.12: Example of (a) collusive video (for likes), (b) collusive channel (for subscriptions), and (c) collusive video (for comments) detected by our models. Sensitive information are blurred.

8.9 Conclusion and future work

With the increase in the involvement of content creators and content consumers on YouTube, social growth has become the most important metric for popularity. To achieve rapid social growth, a large number of content creators go against the stream by registering their videos/channels to blackmarket services for collusive appraisals. Although many studies have been carried out to detect fraud and fake activities on multiple online media platforms, identification of collusive entities remains a relatively important unexplored area of research. The major contributions of this work are manifold. (i) We collected a large dataset of YouTube videos submitted to blackmarkets for collusive likes, comments and subscriptions. *To the best of our knowledge, this is the first dataset of this kind.* (ii) We analyzed the collusive videos from two perspectives: propagation dynamics and video metadata. The YouTube channels are analyzed based on their location, video metadata and network properties. (iii) In order to detect videos submitted in blackmarket services for collusive likes and channels submitted for collusive subscriptions, we utilize one-class classification models trained only on the collusive data. The SVM-based model achieves 0.911 TPR using video metadata features and 0.910 TPR using channel features to detect videos and channels submitted to blackmarket services for collusive likes and subscriptions respectively. (iv) We then proposed `CollATe`, a system that combines three feature extractors (metadata, anomaly and comment) to learn representations of a video. `CollATe` makes use of extractors effectively for identifying whether a video is registered in blackmarket services for collusive comment appraisals. Extensive experiments on our dataset show that `CollATe` is effective in detecting collusive entities with 0.905 TPR. (v) As a final contribution, we show the important implications of the collusive entity detection task. We expect this research to push further studies in online media outlets to explore the dynamics of collusive behavior.

We believe our proposed methodologies can be helpful for a variety of tasks in similar research. For the first two models, i.e., identifying videos (channels) submitted to blackmarket services for collusive likes (subscriptions), it can only be adopted to video-sharing platforms as the features used in the models are generated from video (channel) metadata present on YouTube. In case of `CollATe`, the *anomaly feature extractor* and *comment feature extractor* can be adopted by any video or non-video sharing platforms where a user posts texts on some entities of that platform. e.g., in case of Twitter to detect users who submit tweets to blackmarket services for collusive retweet appraisals, the *anomaly feature extractor* can utilize the retweet time-series data to detect sudden changes in the retweeting activity; *comment feature extractor* can utilize the text of the comments posted during collusion and compute a similarity score between the comments. However, we have not performed this experiment in the current work since the major aim of this study is to provide a detection framework for collusive YouTube entities, not developing a framework for detecting collusive entities on Twitter which we explored in our previous studies [3, 4, 5, 48, 169]. It is also to be noted that only the appraisals in online media entities are artificially manipulated by blackmarket services irrespective of the content present in the entities. However, deep learning-based visual features could be too effective to annotate the videos into tags to identify the themes present in the video. This can further help in identifying what type of events (e.g., election campaigns, event promotions etc.) are promoted using blackmarket services.

Despite encouraging results, collusive entities detection still remains a challenging problem with many open research questions. In the future, we are interested in exploring the following avenues. First, we plan to take into account the sentiment of the comments by considering the average comment sentiment during collusion for detecting the collusive activity. Second, we wish to study the interdependency of the collusive videos and collusive users that can help in identifying the core users of the blackmarket services. Third, we intend to detect collusive

entities at an inter-platform level as well. Fourth, our final goal will be to design a web-based scalable collusive entity detection system for online video sharing platforms.

Part IV

Core users in blackmarkets and released datasets

9. Spotting core collusive users in YouTube blackmarket network

Social reputation (e.g., likes, comments, shares, etc.) on YouTube is the primary tenet to popularize channels/videos. However, the organic way to improve social reputation is tedious, which often provokes content creators to seek services of online blackmarkets for rapidly inflating content reputation. Such blackmarkets act underneath a thriving collusive ecosystem comprising *core users* and *compromised accounts* (together known as *collusive users*). Core users form the backbone of blackmarkets; thus, spotting and suspending them may help in destabilizing the entire collusive network. Although a few studies focused on collusive user detection on Twitter, Facebook, and YouTube, none of them differentiate between core users and compromised accounts.
We are the first to present a rigorous analysis of core users in YouTube blackmarkets. To this end, we collect a new dataset of collusive YouTube users. We study the *core-periphery structure* of the underlying *collusive commenting network* (CCN). We examine the topology of CCN to explore the behavioral dynamics of core and compromised users. We then introduce KORSE, a novel graph-based method to automatically detect core users based *only* on the topological structure of CCN. KORSE performs a weighted k-core decomposition using our proposed metric, called *Weighted Internal Core Collusive Index* (WICCI). However, KORSE is infeasible to adopt in practice as it requires complete interactions among collusive users to construct CCN. We, therefore, propose NURSE, a deep fusion framework that *only leverages user timelines* (without considering the underlying CCN) to detect core blackmarket users. Experimental results show that NURSE is quite close to KORSE in detecting core users and outperforms nine baselines.

9.1 Introduction

In recent years, YouTube has grown as a primary video-sharing platform, where content creators create channels and upload videos. The videos are then recommended to the content consumers based on several factors, one of which is the online *social reputation* of the creators and their content. Social reputation is usually quantified by the endorsement of the viewers in terms of likes, (positive) comments, shares, etc. However, an organic way of gaining reputation is a time consuming process, and often depends on several other factors such as the quality and relevance of the video, initial viewers and their underlying connections. Unfortunately, there exist a handful of online reputation manipulation services (*aka* blackmarkets) which help content creators rapidly inflate their reputations in an artificial way [6]. Such services are built on a large thriving ecosystem of collusive network. The underlying network comprising *core users* – fake accounts or sockpuppets [219], which are fully controlled by the blackmarkets (puppet masters), and *compromised accounts* which are temporarily hired to support the core users – these two types of users are together called as *collusive users*. Core users are the spine of any collusive blackmarket; they monitor and intelligently control the entire fraudulent activities in such a way that none of their hired compromised accounts are suspended. Therefore, detecting and removing core blackmarket users from YouTube is of utmost importance to decentralize the collusive network and keep the YouTube ecosystem healthy and trustworthy. In this study, we deal with *freemium blackmarkets* [6] which invite customers to opt for the service for free, in lieu of surrendering their accounts temporarily for blackmarket activities. In doing so, customers gain virtual credit and use it to grow their content's reputation.

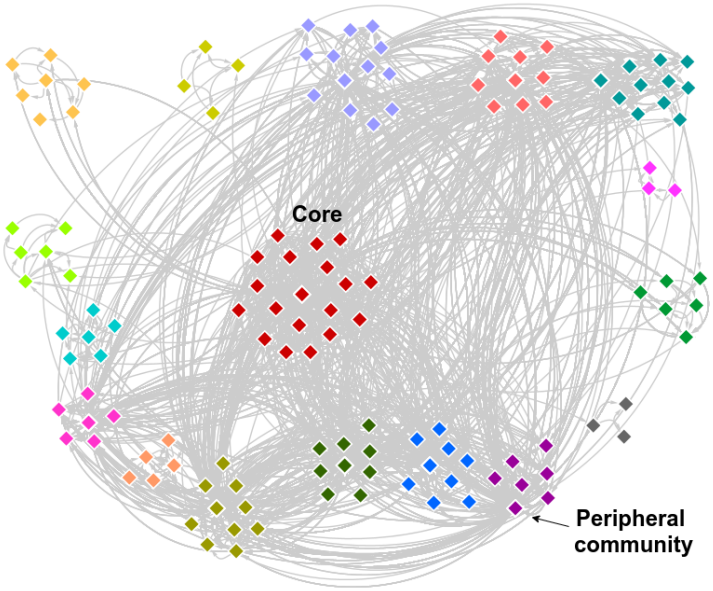


Figure 9.1: Visualization of the collusive commenting network (CCN). Unlike conventional core-periphery structure where peripheral nodes are sparsely connected internally, CCN constitutes dense peripheral communities sparsely connected with the core, indicating the growth of the network up to a certain point where it may not require core users to support compromised users for self-sustainability.

State-of-the-art and Motivation. Several efforts have been made to detect *fake activities* in different online social networks [9, 52]. However, as we suggested in [7], *collusive activities* are very different from usual fake activities. A few studies attempted to explore the dynamics of blackmarkets, mostly for Twitter [5, 9] and Facebook [189]. On YouTube, there exists our proposed method named CollATe to detect collusive users [7]. However, to our knowledge, none of these methods attempted to further divide collusive users into core and compromised accounts.

One may argue that once a collusive account (be it core or compromised) is detected, it should be banned. Then why do we need to explicitly identify core and compromised accounts, while both of them deserve punishment? We argue that the role of a core user is different from a compromised account in the collusive ecosystem; therefore, the extent of punishment may differ. Compromised users are more interested in self-promotion; they join blackmarkets temporarily; they gain appraisals for their online content both organically (genuine interest by other users) and inorganically (through blackmarket services). However, core users, being the backbone of the blackmarkets, always intend to grow and popularize their business. They are permanent members of the blackmarkets; they provoke other users to join the services; and they generally initiate the artificial inflation of the reputation of online content. Therefore, they are more harmful to pollute the online ecosystem. Due to such contrasting behavior of core and compromised users, one may consider that core users should be punished differently than compromised users. For instance, a complete ban of core users would limit the growth of the collusive blackmarkets. However, for compromised users, it may be wise to just warn them and restrict their social network activities for a limited time, instead of a complete ban. The authorities of a social media platform may design suitable policies to handle these two cases.

To our knowledge, **ours is the first attempt to identify and explore the dynamics of *core blackmarket users*. It is also the second attempt after our proposed approach CollATe [7] to explore *YouTube blackmarkets*.**

Present Work: KORSE. In this work, we investigate the dynamics of core users in YouLikeHits, one of the popular YouTube blackmarket services. We start by collecting a novel dataset from YouLikeHits and YouTube, consisting of collusive users, the videos they promote through blackmarkets, and their comments on YouTube videos. We then construct a collusive commenting network (CCN) based on the co-commenting activities among collusive users. We leverage the topological structure of CCN to detect core users using our proposed method, KORSE which is a weighted k-core decomposition method designed based on our proposed metric, *Weighted Internal Core Collusive Index* (WICCI).

Present Work: Core-periphery Structure. An exhaustive analysis on the interactions of core and peripheral nodes reveals a counter-intuitive core-periphery structure of CCN – unlike a conventional network where peripheral nodes are sparsely connected, and get disconnected upon removal of the core, CCN constitutes peripheral nodes which form several small and dense communities around the core (c.f. Fig. 9.1). We further observe that there exists a strong positive correlation between the internal interactions within peripheral communities and the interactions between the core and the peripheral communities. This gives us the evidence that in peripheral communities, compromised users who comment heavily on videos that are co-commented by core users, tend to contribute more to the collusive market. We also present a case study to highlight the major differences between core and compromised users based on their user timelines: (i) Core users, although act as heavy contributors of the blackmarket services, are not the top beneficiaries of the collusive market. (ii) Core users indulge in less self-promotion of videos. (iii) Core users are less active participants of the collusive market than compromised users; they initiate the fraudulent activities and let the compromised users finish the remaining job.

Present Work: NURSE. Although KORSE is highly accurate in detecting core users, it is practically infeasible to deploy as it requires prior information of collusive users to construct CCN. Therefore, we consider core users detected by KORSE as the ground-truth[1] and develop NURSE, a deep fusion framework that *only* considers user timeline (without the underlying CCN) and video submission information to detect core blackmarket users. Experiments on our curated dataset show that NURSE is quite close to KORSE with 0.879 F1-Score and 0.928 AUC, outperforming nine baselines.

Contributions: In short, our contributions are four-fold:

- **Novel problem:** We are the first to address the problem of *core blackmarket user* detection.
- **Unique dataset:** Our curated dataset is the first dataset, comprising *core and compromised collusive YouTube users*.
- **Novel methods:** Our proposed methods, KORSE and NURSE, are the first in detecting core blackmarket users.
- **Non-intuitive findings:** Empirical analysis of the dynamics of core and compromised users reveals several non-trivial characteristics of blackmarket services.

Reproducibility. To encourage reproducibile research, we have made the codes and the (partial) dataset[2] available at the following link: https://github.com/LCS2-IIITD/KORSE-NURSE.

9.2 Related work

We summarize related studies by dividing them in two subsections: (i) blackmarkets and collusion, and (i) network core detection.

Blackmarkets and Collusion: Recently, the activities of blackmarket services have garnered significant attention among the researchers due to the way they provide artificial appraisals to online media content. [6] provided a broad overview of the working of blackmarkets. Our previous work [5] attempted to detect collusive retweeters on Twitter. We also mentioned how collusive users are asynchronous in nature as compared to normal retweet fraudsters. [221] presented the first large-scale analysis of "lateral astroturfing attack" (an astroturfing attack where the automated tweets are posted by compromised accounts and are deleted immediately after they are created). [189] showed how collusion networks collect OAuth access tokens from colluding members and abuse them to provide fake likes or comments to their members. [222] examined five manipulation services in depth, each advertising the ability to inflate customer's standing on the Instagram social network. [51] proposed an automated

[1]Collecting the ground-truth for fake/genuine entity detection is challenging, which usually requires annotations from annotators with domain expertise [220]. However, obtaining the ground-truth data of core blackmarket users is almost impossible. We do not know any legal way to find "core" blackmarket users. Therefore, we consider KORSE as an oracle, which cannot be used in practice but can be used to create the ground-truth. One can argue that the current way of creating the ground-truth may be unconvincing. However, we perform several case studies to provide strong empirical evidence which may validate our strategy of collecting the ground-truth. We do not know any other way of ground-truth creation for this problem unless blackmarkets themselves provide the same!

[2]Full dataset will be available upon acceptance of the paper.

Table 9.1: Qualitative comparison of KORSE and NURSE with similar approaches.

	[223]	[227]	[229]	[224]	[226]	[7]	KORSE	NURSE
Detect collusive users						✓	✓	✓
Detect core blackmarket users							✓	✓
Graph-based approach	✓	✓	✓	✓	✓		✓	
Deal with weighted graph	✓			✓			✓	
Consider profile information						✓		✓
Consider content information						✓		✓

approach to detect collusive behavior in question-answering systems. Our previous work [7] is the closest to the current research, which detects collusive blackmarket users on YouTube. However, it does not focus on detecting *core blackmarket users*.

Network Core Detection: Due to the abundance of literature on network core detection, we restrict our discussion to some selected works that we deem as pertinent to our study. k-core decomposition [223] is considered to be the *de facto* to detect core nodes. It is based on the recursive removal of vertices that have degree less than k in the input network. [224] proposed an algorithm to detect core-periphery structure in networks. The goal of this algorithm is to identify densely connected core nodes and sparsely connected peripheral nodes. [225] detected core and periphery using spectral methods and geodesic paths. [226] studied the problem of collapsed k-core to identify a set of vertices whose removal can lead to the smallest k-core in the network. [227] showed empirical patterns in real-world graphs related to k-cores. We encourage the readers to go through [228] for a comprehensive survey on network core detection.

Differences with Existing Studies: Table 9.1 compares our methods (KORSE and NURSE) with a few relevant studies. In short, our methods are different from others in five aspects – (i) we are the first to address **core blackmarket user detection** problem; (ii) we are the second after our proposed method CollATe[7] to deal with **YouTube** collusive blackmarkets; (iii) We propose **both unsupervised** (KORSE) and **supervised** (NURSE) methods for core detection; (iv) our **dataset comprising core blackmarket users** is unique; and (v) we provide a **rigorous analysis** to explore the dynamic of core and compromised users.

9.3 Methodology

9.3.1 Dataset description

In this work, we consider YouLikeHits[3], a freemium blackmarket service[4]. We designed web scrapers to extract the ids of YouTube videos submitted to blackmarket services for collusive comments. We used YouTube API[5] to extract the metadata details and comment history of these videos. We extracted $26,166$ YouTube videos which were submitted to YouLikeHits for collusive comments. These videos were uploaded to $11,000$ unique YouTube channels. To our knowledge, this is the first dataset of its kind. Note that the entire data collection process was performed after taking proper Institutional Review Board (IRB) approval.

9.3.2 Preliminaries and graph construction

Here we present some important concepts used throughout the work. Table 9.2 summarises important notations.

[3] https://www.youlikehits.com/

[4] Freemium blackmarkets offer customers to enjoy their services for free with the condition that the customers will temporarily act on behalf of the blackmarkets. Upon signing up, the social media accounts of customers are compromised for a limited time for blackmarket activities, which in turn help them gain virtual credits

[5] https://developers.google.com/youtube/v3

Table 9.2: Important notations and denotations.

Notation	Denotation
$G(N, E)$	Collusive commenting network
V	Set of sets where $\{v_i\} \in \boldsymbol{V}$ indicates the set of videos created and posted by user n_i
$v_{i,j}$	j^{th} video in the video set $\boldsymbol{v_i}$
$comments(n, c)$	No. of comments posted by user n on video c
w_{ij}	Weight of the edge connecting nodes n_i and n_j
wc	weighted coreness score
$core_{th}$	Coreness threshold
G_C	Core subgraph
G_P	Induced subgraph of the peripheral nodes
G_P^L	Largest connected component in G_P
$WCS_{Core,C}$	Weighted cut set between the core and a peripheral community C

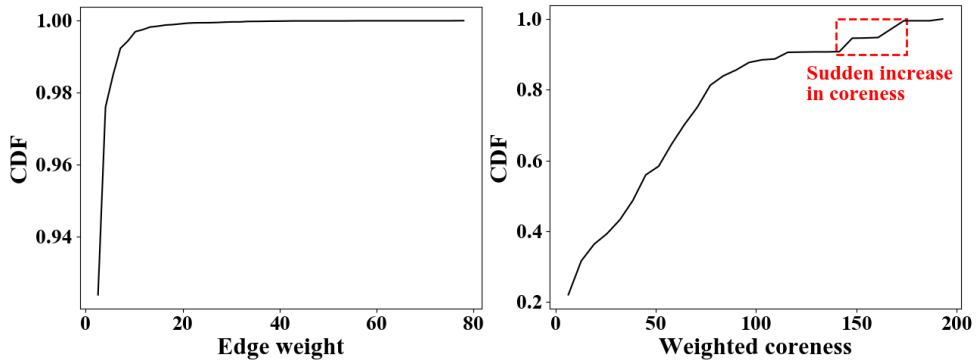


Figure 9.2: Cumulative distribution of (a) edge weights, and (b) weighted coreness scores of nodes in CCN. Contrary to the general observation that coreness score follows power law, we observe that there are relatively large number of nodes having high weighted coreness.

[**Collusive Users and Videos**] We define *collusive users* as those who are involved in the blackmarket activities. There are two types of collusive users – core users and compromised users. We call the videos submitted to freemium blackmarkets as collusive videos.

[**Core Users**] A limited set of online accounts are fully controlled by the blackmarket authorities. These accounts can be bots (fully automated), sockpuppets (controlled by puppet masters) [219] or fake accounts. However, they are used only to benefit blackmarkets. We call these users core blackmarket users.

[**Compromised Users**] These are YouTube content creators who submit their content to the freemium blackmarkets in order to receive artificial comments within a short duration. Being freemium customers, their accounts are compromised for a limited time to perform illegal activities by commenting on videos of other blackmarket customers.

[**Collusive Commenting Network (CCN)**] A CCN is an undirected and weighted network $G(N, E)$, where each node $n \in N$ represents a collusive user, and two nodes n_i and n_j are connected by an edge $e_{ij} = \langle n_i, n_j \rangle$ if the corresponding users co-commented on the same videos. The weight w_{ij} of the edge e_{ij} is calculated as per Eq. 9.1. Let us denote a set of sets, $V = \{\{\boldsymbol{v_1}\}, \{\boldsymbol{v_2}\}, \{\boldsymbol{v_3}\}, \dots\}$, where $\{\boldsymbol{v_i}\}$ indicates the set of videos posted by collusive user n_i. $\{\boldsymbol{v_{i,j}}\}$ indicates the j^{th} video in the set $\boldsymbol{v_i}$.

[**Inter-user Comment Count**] The number of comments posted by the collusive user n on video c is denoted by $comments(n, c)$. We define *Inter-user comment count* (IUCC) for a video c and a pair of users n_i and n_j as the minimum of the number of comments by n_i and n_j on c.

$$IUCC(n_1, n_2, c) = min\big(comments(n_1, c), comments(n_2, c)\big) \tag{9.1}$$

Table 9.3: Topological properties of CCN.

Property	Value
# nodes	$1,603$
# edges	$51,424$
Avg./max/min edge weight	1.392 / 78 / 1
Avg./max/min weighted degree of nodes	89.367 / 1638 / 1
Unweighted edge density	0.040
Unweighted clustering coefficient	0.737
Network diameter	8

[**Edge weight**] We measure the *edge weight* between two nodes (collusive users) n_i and n_j as follows:

$$w_{ij} = \sum_{\substack{p=1 \\ p \neq i,j}}^{|V|} \sum_{q=1}^{|v_p|} IUCC(n_i, n_j, v_{p,q}) \tag{9.2}$$

The edge weight w_{ij} indicates the aggregated IUCC across all the videos co-commented by n_i and n_j, excluding their own videos. We exclude the videos created by n_i and n_j since the comments on these videos can be easily manipulated (added or deleted) by the owners themselves. Table 9.3 summarises the properties of CCN. Fig. 9.2(a) shows the cumulative distribution of w_{ij}.

9.3.3 Weighted k-core decomposition

Given a graph $G(N, E)$, the weighted k-core detection problem aims to find *k-core* (or core of order k), the maximal induced subgraph denoted by $G_k(N_k, E_k)$ such that $G_k \subseteq G$ and $\forall n \in N_k : deg(n) \geq k$. The following two methods are often used to solve this problem: *k-core decomposition* [224] and *core-periphery algorithm* [230]. In our case, we choose *k-core decomposition*[6]. In (weighted) k-core decomposition, to detect core users, we repeatedly delete nodes with (weighted) degree[7] less than k until no such node is left (this is also known as "shaving" method [227]). The reasons behind choosing k-core decomposition are as follows: (i) It has been empirically shown to be successful in modeling user engagement [226, 231]; (ii) Unlike k-core, core-periphery algorithm fits more closely with networks where the nodes are not closely connected to each other [232]. However, in blackmarket services, the sole purpose of collusive users to join the services is to gain credits (by providing collusive appraisals to the content of other users) which can be used by them to artificially inflate their social growth. This strengthens the connectivity among the collusive users. The reason behind expecting high interactions among users stems from the fact highlighted in our previous work [7] that different collusive users retweet the same tweets on the collusive market regardless of the topic of the tweets. We expect a similar behavior in case of YouTube comments, i.e., different collusive users tend to comment on the same videos in order to earn credits. In our dataset, a collusive video has an average of 3 comments by collusive users. This would create more relations (edges) between nodes in CCN.

9.3.4 WICCI: Expected behavior of core users

We frame the core detection problem in CCN as the weighted k-core decomposition problem in CCN. k-core decomposition assigns a *coreness value* to each vertex. In our case, the coreness value ranges from 1 to 193, with an average value of 48.7. We obtain an ordered list of vertices sorted in decreasing order of the coreness value. Typically, the node assigned with the highest coreness value is said to be the "most influential node" in the graph. The subgraph formed with such highly influential vertices is known as *degeneracy-core* or k_{max}-core. On running the weighted k-core decomposition on CCN, we obtain a *degeneracy-core* consisting of 8 users. We expect the distribution of nodes to continually decrease with increasing coreness, as observed in typical core-periphery structures. However, we observe that the fraction of nodes with a high weighted coreness is unusually high (~12.1% users with ≥ 100 coreness score as shown in Fig. 9.2(b)). This indicates the presence of *a larger set of core users*.

[6]We use the weighted version of *k-core decomposition* to incorporate the edge weights (see Eq. 9.1 for more details).

[7]The weighted degree of a node is the sum over the edge weights of the connected edges.

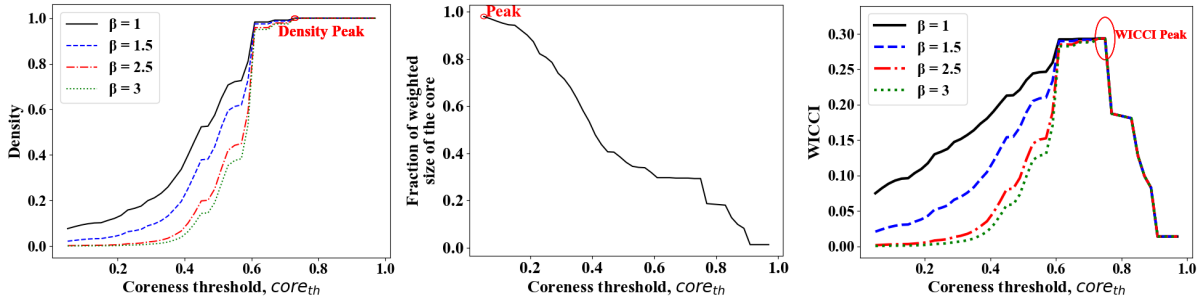


Figure 9.3: Variation of (a) density, (b) fraction of weighted size of core, and (c) WICCI with varying $core_{th}$. (a) Initially, the density dominates over fraction of weighted size of the core and hence WICCI increases rapidly in (c). (b) In the later stages, the inverse happens – fraction of weighted size of core dominates which results in WICCI declining steeply in (c). The WICCI peak of 0.294 is observed at $core_{th} = 0.73$ in (c). All nodes with a weighted coreness above $core_{th}$ are part of the core. Note that despite varying the density of core w.r.t WICCI by changing the density coefficient (β), we observe a similar WICCI peak in all cases.

Therefore, in CCN, to define the partition of core and compromised users, we propose a metric, called *Weighted Internal Core Collusive Index* (WICCI) which is motivated by [224]. WICCI is used to partition the list of decreasing weighted coreness values by a "coreness threshold". The nodes whose coreness is above the threshold are eligible to be the core nodes, while the remaining nodes are considered as compromised users. To define WICCI, we consider two important properties of core users as follows:

C1. **Density:** A core component of a network should be densely connected [224, 232]. We attempt to understand the implications of a dense core in CCN, by considering the flip-side first – *a sparse core*. A sparse core in CCN would have less number of edges connecting vertices internally. In the current scenario, it implies that different users have commented upon different sets of videos. However, the existence of such an entity would mean that there is no cohesion or strategy in the way core users operate. They may be commenting randomly on different videos. The existence of a dense core, however, would imply that different users are commenting on a same set of (collusive) videos, indicating some cohesion or strategy. Note that when we increase the coreness threshold, the subgraph of the core formed has an increasing density (and a decreasing size).

$$\texttt{WICCI} \propto density^{\beta} \tag{9.3}$$

where β is the density coefficient. We utilize β to vary the proportionality of WICCI with density.

C2. **Fraction of weighted size of core:** There is a major flaw in considering only density to define a core. Density does not take into account the edge weight i.e., the volume with which the two users have commented together on same videos. We intuitively expect that inside a core, a high fraction of the commenting activities take place. We define W_G as the weighted size (sum of the weights of edges) of CCN and W_C as the weighted size of the core subgraph G_C. Correspondingly,

$$\texttt{WICCI} \propto \frac{W_C}{W_G} \tag{9.4}$$

Combining (3) and (4), we get

$$\texttt{WICCI} = k \times \frac{W_C}{W_G} \times density^{\beta} \tag{9.5}$$

where k is the constant of proportionality. We assume it to be 1.

9.3.5 KORSE: A graph-based method for core detection

By considering the above properties of collusive entities, we design KORSE (**K**-core decomposition for c**OR**e collu**S**ive us**E**rs detection), a modified version of (weighted) k-core decomposition that is designed for detecting

Algorithm 3: KORSE algorithm

Input: CCN $G(N, E)$
Output: G_c: Subgraph containing core nodes
Initialize $wicci_{max} \leftarrow 0$
▷ *Running the weighted k-core decomposition on $G(N, E)$.*
wc = List of weighted coreness scores for nodes in $G(N, E)$.
▷ *Sort N by wc and push into stack S*
$\mathcal{S}$ = Stack of nodes in G in descending order of weighted coreness, wc.
▷ *Running set of core nodes*
$core_n \leftarrow [\,]$
▷ *Set coreness threshold as the max weighted coreness*
$core_{th} \leftarrow max(wc)$
while $core_{th} > 0$ **do**
  ▷ *Get node with maximum coreness*
  $n = \mathcal{S}.pop()$
  while $wc(n) >= core_{th}$ **do**
    ▷ *Add n to $core_n$*
    $core_n.add(n)$
    $n = \mathcal{S}.pop()$
  end
  ▷ *As $wc(n) < core_{th}$, we push n back to $\mathcal{S}$*
  $\mathcal{S}.push(n)$
  ▷ *Make induced subgraph of core using current $core_v$*
  $G_{cr} \leftarrow InducedSubgraph(G, core_n)$
  ▷ *Compute WICCI for the current core G_{cr}*
  wicci $\leftarrow WICCI(G_{cr}, G)$
  ▷ *Finding the G_{cr} with maximum WICCI*
  if *wicci* $> wicci_{max}$ **then**
    $wicci_{max} \leftarrow wicci$
    $G_c = G_{cr}$
  end
  ▷ *Iteratively decrease the coreness threshold*
  $core_{th} \leftarrow core_{th} - 1$
end

core users in blackmarket services based only on the topological structure of CCN. It takes CCN as input and detects core blackmarket users (core subgraph G_C). KORSE is implemented by decreasing the coreness threshold and consequently making larger subgraphs of the core. The subgraph with the largest WICCI is our final core.

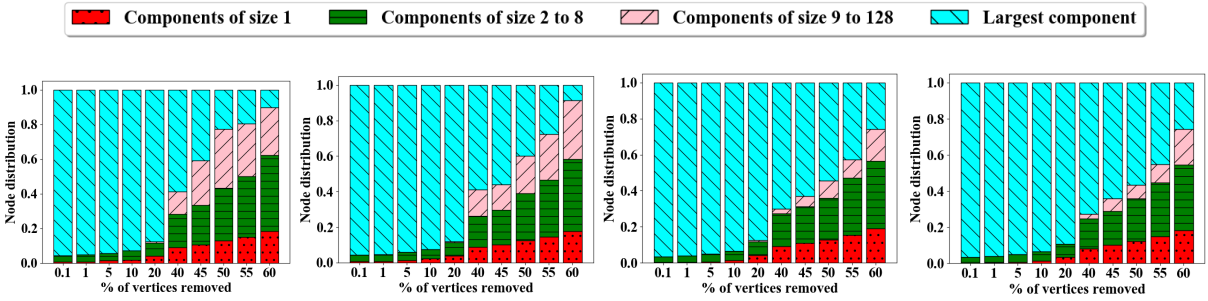
Algorithm 3 presents the pseudo-code of KORSE. Firstly, we apply weighted k-core decomposition which gives the weighted coreness score $wc(n)$ for each vertex $n \in N$. The vertices are then sorted in decreasing order of wc and pushed into a stack $\mathcal{S}$. The top of the stack is the node with the maximum weighted coreness. Next, we create a running set ($core_n$) of core nodes initially with no node. The running coreness threshold $core_{th}$ is set to the maximum value of weighted coreness wc_{max}. Next, $core_{th}$ is iteratively decreased, and the set of core nodes is updated by adding all nodes n which have $wc(n)$ greater than $core_{th}$. Next, G_{cr}, an induced subgraph is created only by the core nodes. Further, WICCI of G_{cr} is calculated. The induced subgraph with the maximum WICCI ($wicci_{max}$) is the core of the graph, and the corresponding $core_{th}$ is the coreness threshold.

On applying KORSE on CCN, we obtain an ideal coreness threshold of 0.73 on a max-normalized scale, with a peak WICCI value of 0.294 (c.f. Fig. 9.3) for different values of the density coefficient β. We explore the variation of WICCI with $core_{th}$:

C1. Initially, as $core_{th}$ increases $(0.1 - 0.5)$, users of low wc (which contribute less to the overall collusive activity of the network) are removed from the core subgraph, leading to rapid increase in density of the core subgraph and a relatively smaller decrease in the fraction of weighted size of the core. Initially, density dominates the fraction of weighted size of the core and hence WICCI increases (c.f. Fig. 9.3(a)).

C2. Towards the higher values of $core_{th}$ (> 0.8), density obtains its maximum value of 1. However, the fraction of weighted size of the core decreases rapidly due to the continued exclusion of more nodes with relatively higher wc. As $core_{th}$ increases further, the fraction of weighted size of the core dominates density towards the latter values of $core_{th}$, and hence WICCI decreases (c.f. Fig. 9.3(b)).

C3. In the mid-range values $(0.6 - 0.7)$ of $core_{th}$, the peak of WICCI is observed. The corresponding core formed by the nodes (with wc higher than $core_{th}$) leverages both the density and the fraction of weighted



(a) Weighted degree (b) Unweighted degree (c) Weighted coreness (d) Unweighted coreness

Figure 9.4: The distribution of nodes in components of sizes present in CCN after removing nodes in the decreasing order of (a) weighted degree, (b) unweighted degree, (c) weighted coreness, and (d) unweighted coreness. The network visibly disintegrates into smaller components when at least (a) 50% (b) 55% (c) 60%, and (d) 60% are removed from the network. Despite a large removal of nodes, the remaining network has a high connectivity.

size of CCN (c.f. Fig. 9.3(c)).

The core obtained on applying KORSE consists of 148 nodes and (surprisingly) *is a complete graph*. Nearly 30% of the entire collusive commenting activities of the network happens among 10% of the core nodes. The periphery consists of $1,455$ nodes and has an edge density of 0.0355. Nearly 60% of the commenting activities take place among the peripheral nodes despite 90% of the users belonging to it. The rest 10% activities are captured between the core and the peripheral nodes (cross-edges between core and periphery).

9.4 Impact of core on CCN

To closely explore the connectivity of the core in the network, we analyse the effect after removing the core from CCN. [233] reported that in a conventional social network, the removal of core breaks the graph into small disconnected components. However, in our case we notice that the graph does not break into smaller components even after removing a large fraction of core nodes (c.f. Fig. 9.4). The possible reasons for such a behavior are as follows:

C1. **Estimated core may be incorrect:** One may argue that our metric WICCI to estimate the core may be flawed. It may be possible that the core is larger than what we estimate. To verify this, we start by removing the vertices from CCN in the decreasing order of the (i) weighted degree (c.f. Fig. 9.4(a)), and (ii) weighted coreness wc (c.f. Fig. 9.4(c)). We observe that the point where the size of the largest connected component decreases and the number of small disconnected components increases drastically, should be the appropriate value of $core_{th}$. However, we notice that such a point arises only after removing 50% and 60% of nodes based on weighted degree and weighted coreness of vertices from CCN, respectively. This would suggest that at least 50% of the vertices belong to the core. However, the density of the core reduces significantly (c.f. Fig. 9.5). This violates one of the fundamental properties of a core that it should be incredibly dense. [233] observed near-complete degradation of the largest connected component after only removing 10% of the nodes based on degree. Therefore, the observed pattern is not the artifact of our proposed metric WICCI, but a result of the high connectivity even among users of low coreness.

C2. **Weighted k-core decomposition may be incorrect:** One may argue that we should consider the traditional *unweighted* k-core decomposition [233], instead of considering the weighted edges. We perform similar experiments by removing vertices in the order of the (i) unweighted degree (c.f. Fig. 9.4(b)) (as suggested in [233]) and (ii) unweighted coreness (c.f. Fig. 9.4(d)). We observe similar results in both the cases where the network breaks into many small disconnected components upon removing at least 55% of the nodes. This would again make the core incredibly sparse (c.f. Fig. 9.5). Therefore, applying weighted k-core decomposition is not a reason for the late disintegration of the graph into smaller components.

Possible explanation: connected periphery. We examine G_P, the induced subgraph of the peripheral nodes

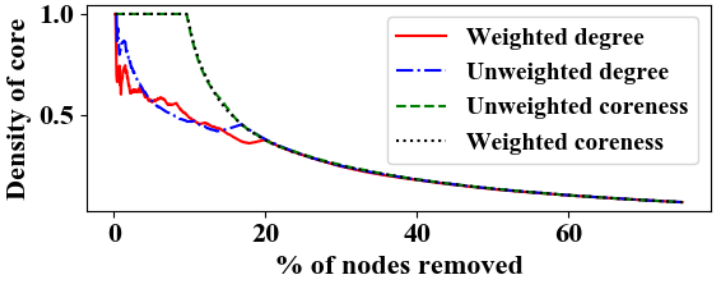


Figure 9.5: Change in the density of core with the number of nodes removed.

independently with specific focus on its largest connected component G_P^L.

- G_P^L and G_P have $1,376$ and $1,455$ nodes, respectively.
- G_P^L and G_P have edge density of 0.03674 and 0.0355, respectively.
- G_P^L has an average path length of 2.6355.
- Lastly, as stated earlier, when we progressively remove the core from CCN, the periphery largely remains intact.

This provides evidence that there is a significant connectivity among nodes in the periphery. This does not fall within the conventional structure of the periphery which is generally described as small disconnected components. Instead, we visualize the periphery in G_P^L as smaller and relatively dense communities (c.f. Fig. 9.1). One possible reason for a connected periphery may be that the graph has organically grown to a stage where despite the detection of the core users, the blackmarket service is in a self-sustainable stage and is no longer driven by the core users alone. A solution would be to detect the core users at an early stage to halt the growth of the market. To identify the network at its infancy, one would have to create multiple snapshots of the blackmarket services over a period of time, which is a computationally expensive task.

9.5 Interplay between core and peripheral communities

Here, we study the interactions between the core and periphery, and highlight critical observations. We start by dividing the videos V into three categories:

C1. **Core-core videos** are the set of videos commented exclusively by core users.

C2. **Core-periphery videos** are the set of videos commented by both core and peripheral users.

C3. **Periphery-periphery videos** are the set of videos commented exclusively by peripheral users.

Here, (1) and (2) are responsible for the formation of edges within core; (2) and (3) are responsible for the formation of edges within periphery; (2) alone is responsible for the formation of edges between core and periphery.

Next, we define the community structure in CCN. A "good" community in CCN is the one in which the users of the community have co-commented heavily on a set of videos. Due to the high connectivity observed in the periphery (mentioned in the earlier section), we speculate that the periphery consists of several small communities. To check this, we run the weighted version of the Louvain community detection method [234] for detecting peripheral communities C_P^L from G_P^L (the largest connected component in the induced subgraph in the periphery). The modularity of the community structure detected by Louvain is 0.397, and the number of large communities (with size > 40) is 9. It indicates that there exist large communities of collusive users that comment on the same set of videos. Next, we define the interaction within the peripheral community based on the amount of collusive commenting activities occurring inside the community. We categorize these interactions using (a) weighted size, and (b) average weighted degree of nodes in the peripheral community. We also quantify the interactions between core and each of the peripheral communities based on the amount of commenting activities on the core-periphery videos.

[**Internal Interaction of Peripheral Community**] We define the internal interaction of a peripheral community as a measure of the collusive commenting activities within the community.

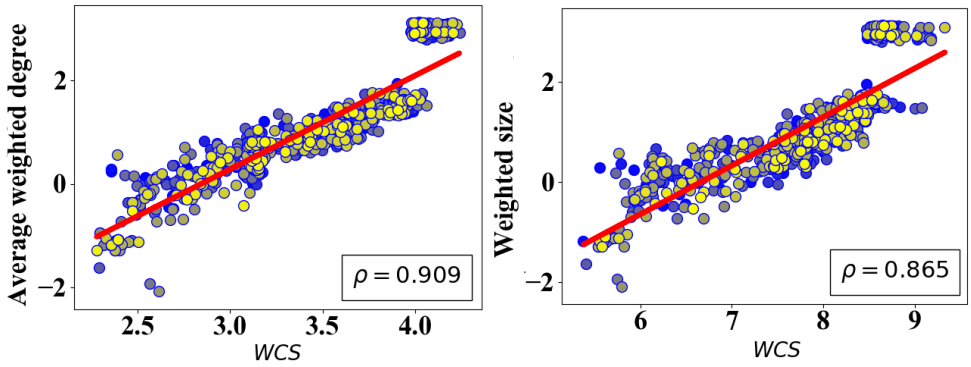


Figure 9.6: A strong positive correlation between weighted cut-set WCS and – (a) average weighted degree, and (b) weighted size of the peripheral communities. Different colors indicate communities obtained in different executions of Louvain method. The Pearson's ρ is also reported.

We further categorize the internal interaction using the following metrics:

C1. ***Average weighted degree of nodes in the community***: It captures the average collusive commenting activities taking place within the community.

C2. ***Weighted size of the community***: It is measured by the sum of weights of all the internal edges of a community, capturing the total intra-community collusive commenting activities.

[**Independent Interaction of Core and Peripheral Community**] We define the independent interactions of core and a peripheral community as a measure of the collusive commenting activities taking place between the core and the peripheral community. This indicates the participation of the peripheral users in commenting on core-periphery videos.

To capture independent interactions between core and peripheral community C, we utilize the **weighted cut-set** $WCS_{Core,C}$ as the sum of the weights of edges connecting the core and C. Since the size of the peripheral communities varies, we normalize $WCS_{Core,C}$ by only $|C|$.

$$WCS_{core,C} = \frac{\textit{Sum of weights of edges connecting core and } C}{|C|}$$

The following observations are drawn from the above (c.f. Fig 9.6):

C1. There exists a positive correlation between the average weighted degree of a peripheral community and $WCS_{core,C}$ (c.f. Fig 9.6(a)).

C2. There exists a positive correlation between the weighted size of a peripheral community and $WCS_{core,C}$ (c.f. Fig 9.6(b)).

From these observations, we conclude that there is a definite positive correlation between the internal interaction within the peripheral communities and that between the core and peripheral communities. Peripheral communities which actively participate in activities associated with the core (such as commenting on core-periphery videos), tend to contribute more to the collusive market.

9.6 NURSE: A deep fusion framework

Although the network topology based weighted k-core decomposition presented in KORSE is highly accurate to detect core blackmarket users, it may not be feasible to adopt in designing a real-world system because of the

following reasons: (i) data arrives in streaming fashion, and the generation of CCN is not possible as the entire snapshot of the blackmarkets at a certain point is impossible to collect; (ii) CCN is often incomplete and highly sparse, and (iii) k-core decomposition is comparatively slow. **However, we consider KORSE as an oracle and the core and compromised users it has detected as the ground-truth to train and evaluate the following model.** To address the above issues and towards designing a real-world system, we propose NURSE (**N**e**U**ral framework for detecting co**R**e collu**S**ive us**E**rs), a neural fusion model to detect core blackmarket users in blackmarket services *based only on the user timeline and video sharing information* (without considering the underlying CCN).

9.6.1 NURSE: Model components

NURSE comprises three components: *metadata feature extractor* **(MFE)**, *similarity feature extractor* **(SFE)**, and *textual feature extractor* **(TFE)**; the output of which are further concatenated to form the feature representation of a YouTube user. The combined representation is passed through to a *core detector* module which determines whether the user is a core or a compromised user. The architectural diagram of NURSE is shown in Fig. 9.7. Individual components of NURSE are elaborated below.

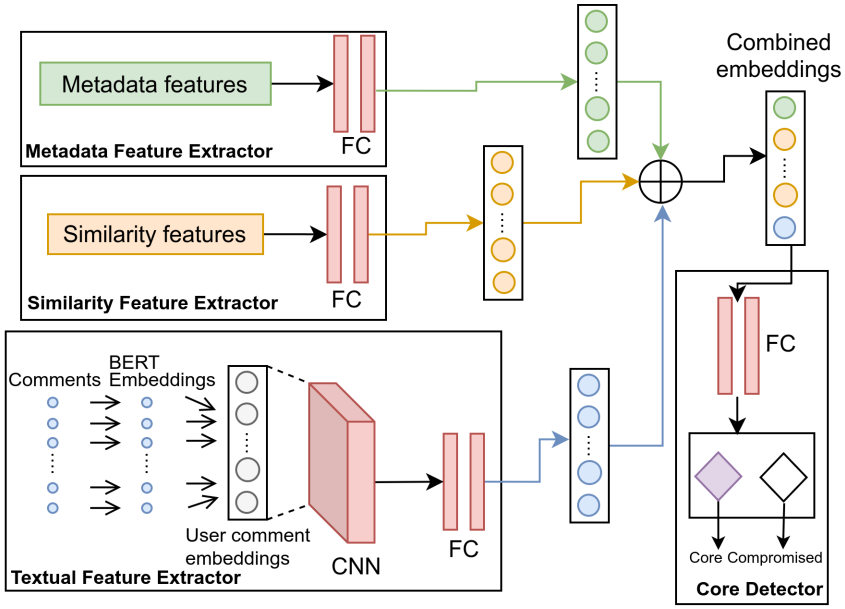


Figure 9.7: A schematic diagram of NURSE.

9.6.1.1 Metadata Feature Extractor (MFE).

We extract 26 metadata features based on the profile information, and videos uploaded by the users. These features are largely divided into four categories:

(a) **Self-comments** (MFE_{1-5}): These features are derived from the comments made by the users on their own videos. We observe that, on an average, compromised users tend to write more self-comments ($\times 1.778$) than the core users, indicating that *core users are less involved in self-promotion*. We take the maximum, minimum, total, average and variance of the comments across self-posted videos as five different features.

(b) **Number of videos uploaded** (MFE_6): It refers to the total number of videos uploaded by the user. On average, core users upload fewer videos, which is $\times 0.633$ less than that of compromised users. *A core user's efforts to benefit from the blackmarkets are lesser* (as they are created by the blackmarket services themselves) than the compromised users.

(c) **Duration of uploaded videos** (MFE_{7-11}): These features measure the duration of the videos uploaded by users. On average, a core user uploads significantly shorter videos, which is $\times 0.628$ less than that of compromised

users. The possible reason could be that *core users are less interested in their own content*; rather their primary objective is to artificially inflate the popularity of other customers' videos. We take the maximum, minimum, total, average and variance of video duration per user as five different features.

(d) **Other features:** Apart from the above features, we also consider the following features related to the rating of the videos posted by a user (in each case, we take the maximum, minimum, total, average and variance as five different features) – the number of likes (MFE_{12-16}), the number of dislikes (MFE_{17-21}) and the number of views received (MFE_{22-26}).

9.6.1.2 Similarity Feature Extractor (SFE).

Collusive users have been shown to post similar/duplicate comments regardless of the topic of the content [7]. We extract two sets of features based on the linguistic similarity of comments posted on the video and video metadata:

(a) **Comment-based features**: We capture similarity features based on the linguistic similarity of comments posted by users. For a user, let the set of her comments on her own videos and the set of comments on other videos be SC and OC, respectively. We first generate embedding of individual comments using pre-trained BERT [235]. We then measure the maximum, minimum, total, average and variance of similarities (cosine similarity) between comments in SC. Similarly, we obtain five similar features, each from the comments within OC and by comparing comments in SC and OC. This results in 15 features (SFE_{1-15}).

(b) **Video metadata based features**: In YouTube, a user can upload her own videos (SV) or act on videos posted by other users (OV). For each video, we combine the text of the video title, video description and video genre. We then generate the embedding of the combined text using BERT. Next, we extract the maximum, minimum, total, average and variance of similarities (cosine similarity) between video embeddings, each from the videos within SV and and videos across SV and OV. This results in 10 features denoted by SFE_{16-25}. We did not extract features from within OC because we observed that doing so heavily biased the model.

9.6.1.3 Textual Feature Extractor (TFE).

We capture textual features from the content of the comments posted by a user. We generate embeddings for every comment using pre-trained BERT [235]. To get a representative embedding for a user, we average out the embeddings of all the comments posted by the user. As collusive users tend to post repetitive text in their comments [5], we feed the resultant embedding into a CNN to capture this inter-dependency. In literature, CNNs have shown to perform well in capturing repetitive patterns in texts [236, 237].

9.6.1.4 Core detector.

The core detector module consists of a fully-connected layer (FC) with softmax to predict where a YouTube user is core or compromised, denoted by $G_c(., \theta_c)$, where θ_c represents the model parameters. For the prediction task, G_c generates the probability of a user u being the core user based on the combined representation $\vec{u}$.

$$P_\theta(u) = G_c(\vec{u}; \theta_c) \tag{9.6}$$

We use the cross-entropy loss (L_d) for our model:

$$L_d(\theta) = y \log \big(P_\theta(u)\big) + (1 - y) \log \big(1 - P_\theta(u)\big) \tag{9.7}$$

9.6.2 `NURSE`: Model specifications

`NURSE` executes three parallel operations - **(1) TFE:** The 1×784 textual vector is fed to a CNN (number of channels = 32, filter size = 2, no padding). Next, the resultant vector is passed to a max-pooling layer and then to a FC layer of size 64. The final output from this operation is a 1×64 vector. **(2) SFE:** The 1×25 similarity vector

is fed to a FC Layer of size 32. A dropout of 0.3 is applied on the FC layer. The final output from this operation is a 1×32 vector. **(3) MFE:** The 1×26 metadata vector is passed to a FC layer of size 16. A dropout of 0.25 is applied on the FC layer. The final output from this operation is a 1×16 vector.

The combined representation is a 1×112 vector. This is then passed to another FC layer of size 16, followed by a softmax layer of size 2 to obtain the final prediction. We utilize the ReLu activation function for all other layers.

9.7 Experiments

9.7.1 Dataset and ground-truth

Although we collected collusive users from the blackmarkets, it is unknown who among them are core blackmarket users. Thus, the ground-truth information about the core and compromised users are impossible to obtain unless blackmarkets themselves provide the data! We, therefore, consider the core and compromised users obtained from KORSE as the ground-truth since it uses the topological structure of the underlying collusive network to detect the core users. We hypothesize that KORSE is highly accurate in detecting core users. We also perform several case studies to validate our hypothesis. We intend to show how much NURSE (a non-topology based method) is close to KORSE (a pure topology-based method). We also present a case study to show whether the detected core users are really meaningful or not.

Since the number of compromised users $(1,455)$ is ~10 times higher than the number of core users (148), we generate two datasets for our analysis: **(i) Dataset (1:1)** is a balanced dataset where equal number of compromised users as that of core users are (randomly) sampled; **(ii) Complete dataset** is an imbalanced dataset where all collusive users are kept. We performe 10-fold stratified cross-validation and report the average performance.

9.7.2 Baseline methods

Since ours is the first work to detect core blackmarket users, there is no existing baseline. We therefore design our own baselines by considering individual components of NURSE in isolation and their combinations:

C1. **MFE:** This model uses only the metadata feature extractor.

C2. **SFE:** This model uses only the similarity feature extractor.

C3. **TFE:** This model uses only the textual feature extractor. Each comment is represented as a 786 dimensional vector using BERT.

We further combine these three components and design three more baselines: (4) **MFE+SFE**, (5) **MFE+TFE**, and (6) **SFE+TFE**.

These baselines also in turn serve the purpose of **feature ablation** to explain which features are important for NURSE.

Are core users the influential nodes in the network? To answer this, we consider three other approaches as baselines which aim to detect influential users:

C7. **INF:** [238] proposed a node influence indicator, called INF, based on the local neighboring information to detect influential nodes in a network.

C8. **Weighted Betweenness Centrality (WBC):** Betweenness centrality (BC) [239] is a measure of node centrality based on the shortest paths. We utilize the approach in [227] to run the weighted version of BC on CCN and detect core users.

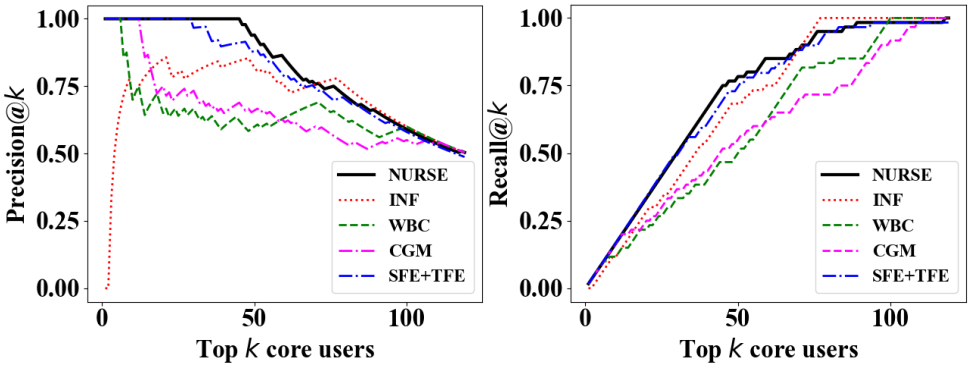


Figure 9.8: Change in the performance of competing methods with the increase of k (the number of results returned) for detecting core users from our dataset (1:1). For better visualization, among the variations of NURSE, we report the results of only the best variation (SFE+TFE).

C9. **Coordination Game Model (CGM):** [240] proposed a coordination game model to find top-K nodes to maximize influence under certain spreading model.

Table 9.4: Performance (F1-Score and AUC for detecting core users) of the competing methods at $k = 148$ (break-even point). The results also explain feature ablation of NURSE.

Method	**Dataset** $(1:1)$		**Complete Dataset**	
	F1 (Core)	**AUC**	**F1 (Core)**	**AUC**
MFE	0.638	0.559	0.268	0.294
SFE	0.816	0.857	0.516	0.472
TFE	0.665	0.773	0.530	0.365
MFE+SFE	0.824	0.882	0.682	0.718
MFE + TFE	0.696	0.767	0.415	0.631
SFE+TFE	0.819	0.865	0.721	0.792
INF	0.750	0.139	0.533	0.113
WBC	0.617	0.304	0.407	0.270
CGM	0.622	0.392	0.302	0.414
NURSE	**0.879**	**0.928**	**0.833**	**0.845**

9.7.3 Performance comparison

Since all the competing methods return a score (or a probability), indicating the likelihood of a user being core, we first rank all the users based on the decreasing order of the score, and then measure the accuracy in terms of precision, recall, F1-Score and Area under the ROC curve (AUC) w.r.t. the 'core' class. Fig. 9.8 shows that NURSE dominates other baselines for almost all values of k (the top k users returned from the ranked list). Table 9.4 summarizes the performance (F1-Score and AUC) of the models at $k = 148$ (as there are 148 core users; it is also known as break even point) – NURSE turns out to be the best method, followed by MFE+SFE (for balanced dataset) and SFE+TFE (for imbalanced dataset). Similarity feature extractor (SFE) seems to be the most important component of NURSE, followed by TFE and MFE. Among influential node detection methods, both INF and CGM seem to be quite competitive.

9.8 Case studies

We further delve deeper into the characteristics of some of the core users detected by both KORSE and NURSE.

C1. **Core users are heavy contributors**: A core user, on average, comments significantly ($\times 2.665$) more than a compromised user, indicating that core users are the top contributors to the freemium collusive market.

C2. **Despite being heavy contributors, core users are not the largest beneficiaries of the collusive market**: We calculate the average number of comments received by the videos uploaded by collusive users, and rank them in decreasing order of this quantity. We find only one core user from the top 30 users. Upon further investigation, we notice that only 8 out of the top 250 users are core users. This suggests that core users, despite being heavy contributors, are not the largest beneficiaries of the collusive market.

C3. **Core users aggressively participate in the collusive market**: We observe that the average number of comments made per collusive video by core users is twice ($\times 1.997$) higher than that of compromised users. This indicates an aggressive behavior to promote the videos they comment on.

C4. **Channels controlled by core users are not popular**: We observe that the channels controlled by core users are not the popular YouTube channels. More than 85% of the channels have a subscriber count of less than $1,000$. This clearly indicates that the primary objective of the core users is not to promote their own videos/channels.

C5. **Channels controlled by core users have less uploaded videos**: We observe that the channels controlled by core users usually do not contain much YouTube videos. More than 90% of the channels have a video count of less than 100. This further corroborates the theory behind the working principle of core blackmarket users.

Despite the above suspicious characteristics exhibited by core channels, we observe that till date, ~93% of the core channels **continue to be active on YouTube**. On average, these core channels have been active on YouTube for over 4 years (1497 days). It indicates how core channels are able to evade the current in-house fake detection algorithms deployed by YouTube.

9.9 Conclusion

This work addressed the problem of detecting core users in YouTube blackmarkets. We curated a new dataset of collusive YouTube users. We then proposed KORSE, a novel graph-based method to segregate core users from compromised accounts. Empirical studies revealed interesting dynamics of core and compromised users. As KORSE is practically infeasible to design due to its dependency on the underlying collusive network, we further proposed NURSE, a deep fusion model that leverages only the user timeline and video submission information to detect core users. Extensive experiments on our dataset showed that NURSE is highly similar to KORSE in detecting core users. Experiments on our curated dataset show that NURSE is quite close to KORSE with 0.879 F1-Score and 0.928 AUC, outperforming nine baselines. Summarizing, our study contributed in four aspects – problem definition, dataset, methods and empirical observation. We also present a case study to highlight the major differences between core and compromised users based on their user timelines. We also made the code and dataset available for the purpose of reproducibility.

10. A multi-platform data repository of artificially boosted online media entities

The rise of online media has incentivized users to adopt various unethical and artificial ways of gaining social growth to boost their credibility within a short time period. In this work, we introduce ABOME, a novel multi-platform data repository consisting of artificially boosted online media entities (also known as blackmarket-driven *collusive entities*), which are prevalent but often unnoticed in online media. ABOME allows quick querying of collusive entities across platforms. These include details of collusive entities involved in blackmarket services to gain artificially boosted appraisals in the form of *likes*, *retweets*, *views*, *comments*, *follows* and *subscriptions*. ABOME contains data related to tweets and users on Twitter, YouTube videos and YouTube channels. We believe that ABOME is a unique data repository which can be used as a benchmark to identify and analyze blackmarket-driven fraudulent activities in online media. We also develop SearchBM, an API and a web portal which offers a free service to identify blackmarket entities.

10.1 Introduction

The past decade has seen a momentous rise in Online Social Networks (OSNs) such as Twitter, YouTube, and Facebook, which help people connect for personal and business interactions. These platforms now boast billions of active users, thereby making them an important component of today's human social fabric. People share and form thoughts and opinions about events, products, and other people on these platforms. This makes online media an attractive platform for people who wish to propagate their opinions to spread their agenda, such as promoting a product or political ideology. Therefore, gaining a stronger influence on online media platforms carries a high level of economic benefit.

However, in order to spread an opinion, users require a large reach across the network. This reach can either be acquired *organically* – by posting quality content over time and gaining popularity, or *inorganically* – by certain online media-driven blackmarket services that allow users to boost the reach of their content artificially. Here, we briefly describe – what are these blackmarket services, why are they used, and how do they work. A substantial number of studies have investigated the phenomena of **fake** [20, 52, 54, 140, 241], **fraudulent** [30, 63, 78, 103, 161, 242] and **spam** [23, 60, 243] activities. We encourage the readers to go through [27, 244, 245] for detailed surveys on false and fake information on the web.

There are relatively fewer studies on the detection and analysis of collusive activities that result in an artificial boosting of social growth. Our recent investigations [3, 4, 5, 7, 48, 49, 169, 201] revealed that existing fraud detection strategies are not suitable for blackmarket-driven collusive entity detection. These studies reported that collusive users are not bots or fake users; rather, they are normal users showing a mix of organic and inorganic activities with no synchronicity across their behaviors. ABOME would help researchers analyze blackmarket-driven collusive activities and build systems to detect them.

ABOME consists of multi-platform datasets for collusion in online media collected from two major blackmarket services - YouLikeHits and Like4Like. ABOME comprises datasets of two types – *historical data* and *time-series*

data. The former type of dataset is divided into two parts – the first part consists of 23,522 collusive retweets and 18,368 collusive follower requests for Twitter; the second part consists of 58,091 collusive likes, 25,106 comments, and 7,847 subscriptions requests for YouTube. The latter type of dataset consists of time-series data of 2,350 Twitter users and 4,989 tweets collected from blackmarket services.

ABOME is unique for the following five reasons:

- To the best of our knowledge, ABOME is the first public dataset of collusive entities in online media. We believe these datasets have tremendous research potential in the field of analysis and detection of collusive behavior in online media.
- ABOME comprises of two types of datasets: *historical* and *time-series* data.
- ABOME provides abundant textual and temporal information of collusive entities for Twitter and YouTube.
- ABOME has an API and a web portal, SearchBM for online media researchers and other enthusiasts to discover collusive entities using text search queries.
- ABOME protects user privacy and can be used in a wide range of research areas, such as fraudulent entity detection, diffusion modeling, social-growth prediction etc.

The entire dataset, along with a smaller sample, is available at the following link: https://zenodo.org/record/4437987
(Dataset DOI: http://doi.org/10.5281/zenodo.4437987).

10.2 Blackmarket services

Websites such as YouLikeHits (https://www.youlikehits.com/), Like4Like (https://www.like4like.org/), TraffUp (https://traffup.net/), JustRetweet (https://www.justretweet.com/) allow social media users to gain appraisals *inorganically* in different forms. They provide services for various **online social networks**, e.g., Facebook (followers, likes, shares, comments), Twitter (followers, retweets, likes), Instagram (followers, likes, comments). Other than OSNs, the blackmarket syndicates also provide service to **video subscription-sharing platforms**, e.g., YouTube (views, subscribers, likes, comments), Vimeo (plays, followers), **music-sharing platforms**, e.g., SoundCloud (plays, followers, likes, reposts, comments), ReverbNation (fans), **business and employment-oriented platforms**, e.g., LinkedIn (followers, connections, endorsements). Organ-

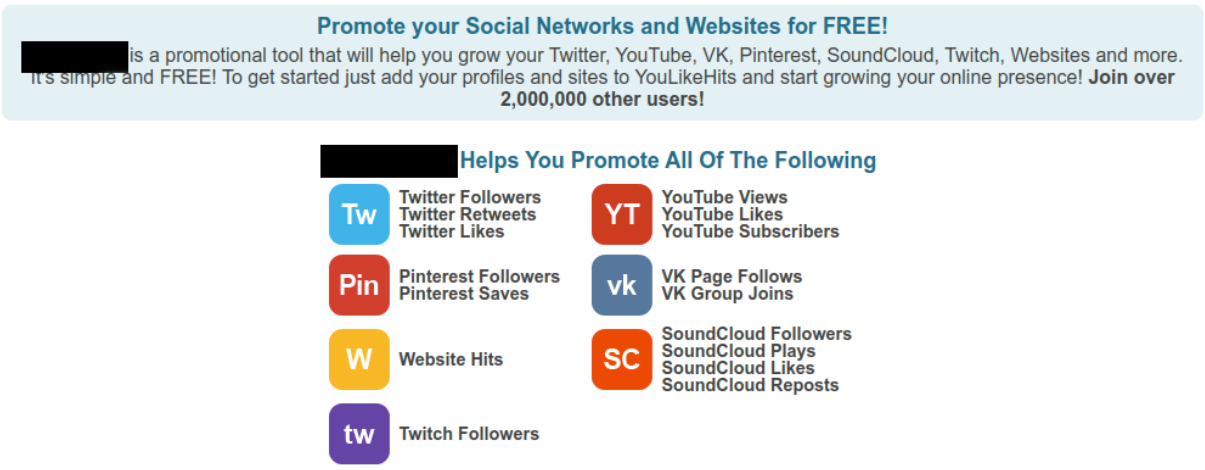


Figure 10.1: Example blackmarket service providing collusive appraisals to online media platforms such as Twitter, Pinterest, YouTube, VK, SoundCloud, Twitch (name of the blackmarket redacted).

ically gaining higher reach on online media is difficult and time-consuming, which boosts the allure of these blackmarket services. With higher reach comes stronger influence, and with stronger influence comes higher economic value. This influence can be used for product promotions or opinion propagation. Fig. 10.1 shows an example of one such blackmarket service which provides collusive appraisals.

[6] divided the blackmarket services into two types based on the model of service - *Premium* and *Freemium*. Customers in the premium services need to pay a cash lump sum to receive blackmarket services. Freemium services may not insist customers to pay. Rather, these services create a community of customers where each member gains appraisals by appraising the content of other customers registered on these blackmarket services. When a customer appraises some content through the blackmarket portal, they earn *credits*, which they can use later to gain appraisal for their own content. Such freemium services are also called *credit-based freemium services*. These services are a menace and pose a threat to the credibility of social media platforms. Unlike bots, detecting users who engage in these activities is difficult because they may display a mix of organic and inorganic behavior - whereby they appraise some content related to their interest acting as a genuine user, and also appraise some content to gain credits on a freemium service.

We made the first attempt to investigate blackmarket customers on Twitter [3]. We collected data related to users engaged in producing fake retweets from four blackmarket services and annotated each user into one of the four categories – *bots, promotional customers, normal customers, and genuine users*. Finally, we ran several classifiers on a set of 64 features to perform multi-class and binary classification to detect collusive users. We further extended this work to show how users involved in premium blackmarket services exhibit unusual properties as compared to those involved in freemium services [5]. [48] detected tweets submitted to blackmarket services using a multitask learning approach. [169] proposed a multi-view learning based approach to detecing collusive retweeters by utilizing various attribute and network views based on the user's posts and interactions on the social graph. We encourage the reader to go through [194] for a comprehensive survey on analyzing and detecting collusive activities in online media platforms.

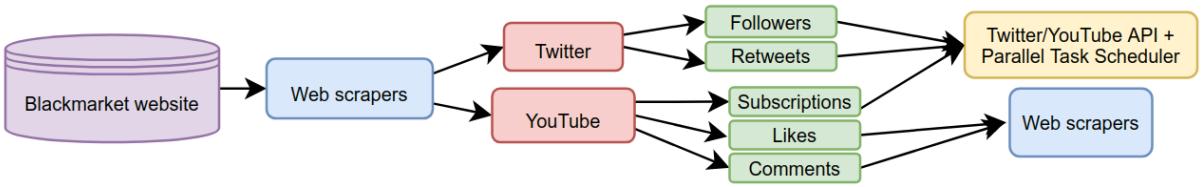


Figure 10.2: The process of collecting ABOME dataset from the blackmarket service.

10.3 Data collection

In this section, we first comment on our ethics and data privacy statement. We then describe the process of creating our datasets in detail.

10.3.1 Ethics and data privacy statement

The entire data collection process has been carried out through Twitter API[1], YouTube API[2] and web scrapers. We did not seek explicit permission from YouLikeHits and Like4Like to scrape the content because these blackmarket sites do not themselves act in line with the terms & conditions (T&C) of the services they connect with[3]. They allow users to boost their influence on these platforms artificially, thereby going against the T&C of Twitter[4] and YouTube[5]. Further, since the data we posted is anonymized, the privacy and identity of these users will not be compromised. We abode by the terms, conditions, and privacy policies of our Institute Institutional Review Board (IRB) approval.

[1] https://developer.twitter.com/en/docs
[2] https://developers.google.com/youtube/v3
[3] We don't reveal the identity of users/tweets (Twitter id, Tweet ids, YouTube channel ids, etc.))
[4] https://help.twitter.com/en/rules-and-policies/platform-manipulation
[5] https://support.google.com/youtube/answer/3399767?hl=en

10.3.2 Collecting data from blackmarket services

After careful IRB approval, we developed web scrapers for parsing two of the most popular blackmarket websites - YouLikeHits and Like4Like. The parser for YouLikeHits used Python's BeautifulSoup library to parse the HTML DOM of the website and got the details of the users and content that are posted for appraisals. The parser for Like4Like used Selenium (https://www.seleniumhq.org/) to run a headless web browser within which the website is loaded, and BeautifulSoup was then used to parse the details of users and content posted for appraisal.

10.3.3 Data anonymization

The data is anonymized by removing all Personally Identifiable Information (PII) and generating pseudo-IDs corresponding to the original IDs. A consistent mapping between the original and pseudo-IDs is used to maintain the integrity and usefulness of the data.

10.3.4 Collecting data at scale from Twitter and YouTube

We focused on collecting data from credit-based freemium services. We divide the datasets into two parts:

- **Historical data**: This consists of all the data for Twitter and YouTube from YouLikeHits gathered via sequential querying of the website's URLs (the historical data for Like4Like was not available, which is why it is missing in our collected dataset) between the period March-June, 2019. The details of the sequential querying technique is explained in last part of this section.
- **Time-series data:** This consists of time-series data (collected every 8 hours) of Twitter users and tweets collected from two blackmarket services – YouLikeHits and Like4Like between the period of March-June, 2019.

Historical data:

Since the entities we collected were a few years old in many cases, we focused on collecting the relatively static properties (such as the profile description and tweet content) and information related to those entities. We collected the following historical data from YouLikeHits:

C1. **Twitter Retweets:** Tweet ids of tweets that have been posted on YouLikeHits in order to gain retweets, and their tweet objects using the Twitter API as well as the last 100 retweets of these tweets (we were only able to collect the last 100 retweets due to Twitter API limitations).

C2. **Twitter Followers:** User ids of accounts that have been posted on YouLikeHits in order to gain followers, and their user objects using the Twitter API.

C3. **YouTube Likes and Comments:** Video ids of videos that have been posted on YouLikeHits in order to gain likes, and their corresponding metadata, which is detailed in the next section.

C4. **YouTube Subscriptions:** Channel ids of YouTube channels that have been posted on YouLikeHits in order to gain subscribers, and their corresponding metadata (which is detailed in the next section).

Time-series data:

Since the entities we collected here were recent and fetched periodically, we collected dynamic information (such as the follower/followee network of Twitter users) related to these entities along with the static information. To collect time-series data, we developed a parallel task scheduler that can use multiple Twitter API keys to fetch a large volume of data at high speed. The task scheduler used the Tweepy library in Python for the API requests and ran in

parallel the requests being made using the Python multiprocessing module. Multiple processes were created, and each process was assigned a given API key to work with. The code for the Parallel Task Scheduler is available at: https://github.com/uditarora/ParallelTweepy. We used the Parallel Task Scheduler to collect time-series data after every 8 hours. We collected the following data between the period of March – June 2019:

C1. **Retweets:** We parsed the tweet ids of the tweets that have been posted by blackmarket customers in order to gain retweets of their own tweets, and collected the tweet objects, retweets of these tweets, and the timelines of the authors of these tweets.

C2. **Twitter Followers:** We parsed the user ids of the user accounts which have been posted by blackmarket customers in order to gain followers on their accounts and collected their timelines, follower and followee networks.

The scheduler was designed in such a way that it was able to collect data for all Twitter entities after every k hours. In our case, we used $k = 8$ to collect data every 8 hours and ignored users who had more than 50,000 followers or followees due to the Twitter API rate limits. Figure 10.2 summarizes the steps followed to produce ABOME dataset from the blackmarket services.

10.4 Data description

We release two different types of datasets as a part of ABOME - historical data and time-series data, as explained in the previous section.

10.4.1 Historical data

We collected the metadata of each entity present in the historical data.

10.4.1.0.1 Twitter: We collected the following fields for retweets and followers on Twitter:

- `user_details`: A JSON object[6] representing a Twitter user.
- `tweet_details`: A JSON object[7] representing a tweet.
- `tweet_retweets`: A JSON list of tweet objects representing the most recent 100 retweets of a given tweet.

User Type	Total	Suspended	Verified
Retweet requests	36,029	12,507	-
Follower requests	23,152	4,784	114

Table 10.1: Summary of Twitter users for which historical information was collected from freemium blackmarket services.

The details of the fields obtained from Twitter API can be found in the Twitter API Documentation (https://developer.twitter.com/en/docs.html).

[6] https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/user-object
[7] https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/tweet-object

10.4.1.0.2 YouTube: We collected the following fields for YouTube likes and comments:

• `is_family_friendly`: Whether the video is marked as family friendly or not.

• `genre`: Genre of the video.

• `duration`: Duration of the video in ISO 8601 format (duration type). This format is generally used when the duration denotes the amount of intervening time in a time interval.

• `description`: Description of the video.

• `upload_date`: Date that the video was uploaded.

• `is_paid`: Whether the video is paid or not.

• `is_unlisted`: The privacy status of the video, i.e., whether the video is unlisted or not. Here, the flag *unlisted* indicates that the video can only be accessed by people who have a direct link to it.

• `statistics`: A JSON object containing the number of dislikes, views and likes for the video.

• `comments`: A list of comments for the video. Each element in the list is a JSON object of the text (*the comment text*) and time (*the time when the comment was posted*).

Type	Total	Suspended
Like requests	69200	11109
Comment requests	30131	5025
Subscription requests	11282	3435

Table 10.2: Summary of YouTube videos and channels for which information was collected from freemium blackmarket services.

We collected the following fields for YouTube channels:

• `channel_description`: Description of the channel.

• `hidden_subscriber_count`: Total number of hidden subscribers of the channel.

• `published_at`: Time when the channel was created. The time is specified in ISO 8601 format (YYYY-MM-DDThh:mm:ss.sZ).

• `video_count`: Total number of videos uploaded to the channel.

• `subscriber_count`: Total number of subscribers of the channel.

• `view_count`: The number of times the channel has been viewed.

• `kind`: The API resource type (e.g., *youtube#channel* for YouTube channels).

• `country`: The country the channel is associated with.

• `comment_count`: Total number of comments the channel has received.

• `etag`: The ETag of the channel which is an HTTP header used for web browser cache validation.

The historical data is stored in five directories named according to the type of data inside it. Each directory contains JSON files corresponding to the data described above.

10.4.2 Time-series data

We also collect the following time-series data for retweets and followers on Twitter:

• `user_timeline`: This is a JSON list of tweet objects in the user's timeline, which consists of the tweets posted, retweeted and quoted by the user. The file created at each time interval contains the new tweets posted by the user during each time interval.

• `user_followers`: This is a JSON file containing the user ids of all the followers of a user that were added or removed from the follower list during each time interval.

• `user_followees`: This is a JSON file consisting of the user ids of all the users followed by a user, i.e., the followees of a user, that were added or removed from the followee list during each time interval.

• `tweet_details`: This is a JSON object representing a given tweet, collected after every time interval.

• `tweet_retweets`: This is a JSON list of tweet objects representing the most recent 100 retweets of a given tweet, collected after every time interval.

The time-series data is stored in directories named according to the timestamp of the collection time. Each directory contains sub-directories corresponding to the data described above.

User Type	Total	Suspended	Verified
Retweet requests	4,989	492	-
Follower requests	2,350	297	28

Table 10.3: Summary of Twitter users for which time-series information was collected from freemium blackmarket services.

Tables 10.1 and 10.3 detail the observations of the historical data and the time-series data for Twitter. It can be seen that only a very small fraction of the user accounts and tweets are no longer available on Twitter. We also observed some blackmarket customers who are marked as 'Verified' by Twitter. Table 10.2 details the observations of the historical data for YouTube collected from the blackmarket services. It can be seen that only a very small fraction of the channels and videos have already been removed from YouTube. This further motivates the need of developing techniques to analyze and detect collusive entities on online media platforms.

10.5 SearchBM: A search engine for collusive entity discovery

To better aid the collusive entity discovery process and provide a better understanding of how and where the entities have been used, we developed `SearchBM`, an API and a web portal for our end-users to effectively query within the `ABOME` dataset. Users can directly search for a query text using the interface of `SearchBM`. The query is then sent to our backend server. The backend of the server is developed using Python-Flask (`http://flask.pocoo.org/`). Currently, the API can accept a given text at the following request paths:

• (`<text>,/collusive_twitter_retweets`): This checks for presence of the query text in the tweets submitted in blackmarket services for gaining collusive retweets.

• (`<text>,/collusive_yt_channels`): This checks for presence of the query text in the description of YouTube channels submitted in blackmarket services for gaining collusive subscriptions.

• (`<text>,/collusive_yt_likes`): This checks for presence of the query text in video description of YouTube videos submitted in blackmarket services for gaining collusive likes.

• (`<text>,/collusive_yt_comments`): This checks for presence of the query text in the video description and user comments of YouTube videos submitted in blackmarket services for gaining collusive comments.

The API returns a JSON object indicating the presence of the text in our datasets. If the entity is found in

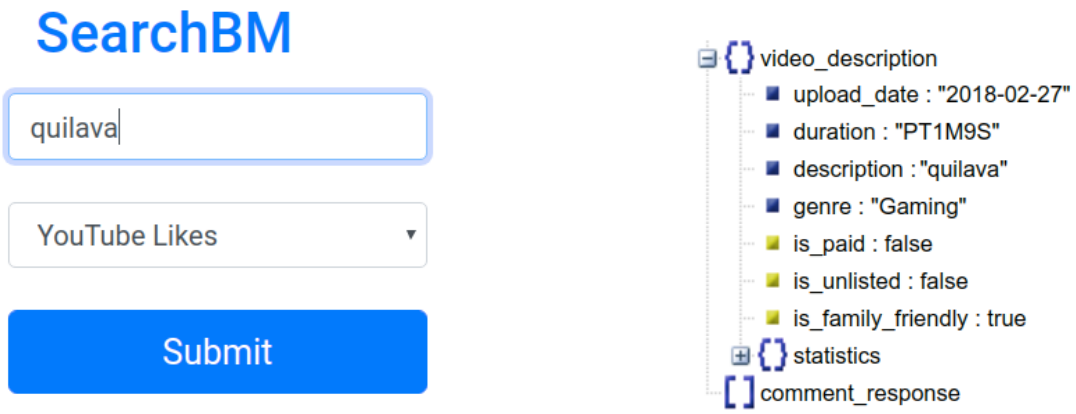


Figure 10.3: SearchBM entity query form. End-users have to enter the query text in the textbox and select one of the types from the drop-down menu.

our database, the API returns the details of the entity from our dataset. Note that to maintain anonymity, we only show specific attributes of the entity and not the identifiers (tweet/user identifier for Twitter data and video/channel identifier for YouTube). We also provide a web interface where end-users can enter the query text and select one of the services - *Twitter retweets, YouTube likes, YouTube comments and YouTube subscriptions* and get the corresponding details via the API. Fig. 10.3(a) shows the interface of SearchBM. End-users have to enter the query text in the textbox and select one of the types from the drop-down menu. On clicking the submit button, the query parameters are sent as a GET request to our API, which returns a JSON object indicating the presence of text in the collusive entity. The front-end uses a JSON viewer, as shown in Fig. 10.3(b) to display the entity details. The API and the web portal is currently live in our local web-server.

10.6 How ABOME is a FAIR-compliant dataset?

In this section, we explain how we have made the ABOME dataset compliant to the four FAIR data principles: *Findable, Accessible, Interoperable and Re-usable*.

To make the ABOME dataset *Findable* and *Accessible*, our dataset is publicly available on Zenodo which allows downloading the entire dataset with the following citation: Hridoy Sankar Dutta, Udit Arora & Tanmoy Chakraborty. (2021). ABOME: A Multi-platform Data Repository of Artificially Boosted Online Media Entities [Data set]. Zenodo. http://doi.org/10.5281/zenodo.4437987.

To make the ABOME dataset *Interoperable* and *Re-usable*, the dataset files are provided in standard JSON (JavaScript Object Notation) format that can be easily parsed using any standard JSON parser and can be exported to other data formats like CSV (Comma Separated Values), XML (Extensible Markup Language) etc. We also provide a readme file that explains each data files to optimize the re-use of the dataset.

10.7 Conclusion

Collusive entity detection is an important problem that has largely been overlooked. To the best of our knowledge, ABOME is the first dataset in the literature that consists of multi-platform data related to blackmarket-driven collusive entities collected from two credit-based freemium services - YouLikeHits and Like4Like. ABOME protects user privacy and can be used in a wide range of research areas, such as fraudulent entity detection, diffusion modeling, social-growth prediction etc. In addition, we also designed an API and a web portal, SearchBM for online media

researchers and other enthusiasts to discover collusive entities using text search queries. We believe that the datasets released in this work will provide more opportunities for researchers to advance the development of technologies in detecting collusive entities in online media, thereby creating an adequate social space. We also encourage researchers working in domain of privacy and security in OSNs to propose interesting tasks and use ABOME as a benchmark. The entire dataset, along with a smaller sample, is available at the following link: https://zenodo.org/record/4437987 (Dataset DOI: http://doi.org/10.5281/zenodo.4437987).

11. Conclusion and Future work

With this chapter, we now summarize the contributions made by this thesis and a discussion of possible directions in which future research could be extended.

As the main thrust of this thesis, we introduced the problem of collusion in online media platforms, analyzed the collusive entities and designed approaches to identify them. In Part I, we explored the fundamentals of collusion and proposed two metadata-based techniques (SCoRe and HawkesEye) to identify collusive users on Twitter. Here, we also highlighted the differences in the working of premium and freemium blackmarket services. We started by studying the problem of understanding and detecting blackmarket-based collusive retweeters. We collected a dataset of blackmarket collusive retweeters and proposed SCoRe that considers 64 novel features to identify the collusive users. We then thoroughly investigated collusive users from multiple premium and freemium Twitter blackmarket services. We provided a detailed analysis of the collusive users based on retweet-centric, profile-centric, timeline-centric and network-centric behavior. We then proposed HawkesEye, a model based on Hawkes process and LDA to identify fake retweets from their retweet timeline by taking into account textual information as well as temporal information. In Part II, we improved our previous models by leveraging the network properties of collusive users. Here, we first proposed a multiview learning approach to create six different views of a user (two attribute-level views and four network-level views), which are then combined to derive user embeddings that classify the Twitter users as collusive or not. We then proposed CoReRank, an unsupervised framework to simultaneously detect collusive users and suspicious tweets controlled by blackmarket retweeting services. In Part III, we first expanded our focus to identify collusive entities on Twitter, who are registered in blackmarket services to gain followers. We proposed DECIFE, a framework that considers the follower/followee relationships and linguistic properties of users to identify whether a user is collusive or not. Next, we shift our focus to analyzing and detecting collusive entities on YouTube. We proposed CollATe, an end-to-end neural architecture that identifies collusive YouTube videos using the combination of three feature extractors (metadata, anomaly and comment). In Part IV, we turned our attention to address the problem of detecting core users in YouTube blackmarkets. Here, we proposed two approaches: KORSE and NURSE to detect core blackmarket users. Finally, we released ABOME, a multi-platform data repository that consists of data related to collusive entities collected from two blackmarket services. We also developed two real-time systems for collusive users. In order to detect collusive users in real-time, we developed a chrome extension SCoRe for end-users. The extension classifies a Twitter user into collusive or genuine and seamlessly integrates the result into user's Twitter pages. The current version of SCoRe is written in Javascript[1] and the backend code is written in Python (using Flask Framework[2]). For a better understanding of how and where the entities have been used, we developed SearchBM, an API and a web portal for our end-users to effectively query within the ABOME dataset. Users can directly search for a query text using the interface of SearchBM. The query is then sent to our backend server. The backend of the server is developed using Python-Flask (http://flask.pocoo.org/). One of the important challenges while conducting this study is collecting the data from the blackmarket service and its associated data from the online media platform. This is due to the limitations and restrictions imposed by the API of the online media platforms and tactics used by the blackmarket services to prevent collecting the data from their platform. However, our developed customized web scrapers for blackmarket services and tools such as ParallelTweepy[3], The-YouTube-Scraper[4] enabled us to collect the large-scale dataset of users from online media platforms. During data collection, I have strictly followed the terms, conditions, and privacy policies of our Institute

[1] https://en.wikipedia.org/wiki/JavaScript

[2] http://flask.pocoo.org/

[3] https://github.com/uditarora/ParallelTweepy

[4] https://github.com/hridaydutta123/the-youtube-scraper

Institutional Review Board (IRB) approval. The privacy and identity of collusive entities is not compromised in the released dataset. The other important challenge we faced was to annotate the collusive users based on the profile information present in the online media platform. We tackled this challenge by clearly defining each type of collusive user based on terms of service of the online media platform. We also gave the annotators complete freedom to search for any information related to collusive users in the web and apply their own intuition.

We now outline interesting research topics that represent a continuation of this thesis.

Fraudulent user/entity detection: A great deal of work has been devoted to fraudulent user/entity detection in online media platforms. The task of detecting fraudulent users includes identifying fake users [20, 246, 247], spammers [60, 159], bots [103, 132, 248], collusive users (contribution of this thesis) [3, 4, 5, 48, 201], sockpuppets [10] etc. Most of the above algorithms only detect individual fraudulent users. However, in reality, it is seen that anomalous phenomena also occur in groups. The group detection task (finding a group of users that jointly exhibit fraudulent behavior) is more difficult as compared to the individual detection task due to the variation present in the inter-group dynamics. In case of collusion, the users of freemium blackmarket services perform actions (retweet/like/comment/subscribe) on collusive entities in order to gain credits. Therefore, it is almost certain that collusive users must have interacted with each other in order to gain credits. We also believe that the availability of our released dataset can help in identifying fraudulent user/entity individuals and groups. Moreover, as our data contains multilingual texts, we anticipate that it will be useful for a broad range of Natural Language Processing (NLP) tasks in the anomaly detection domain.

Connectivity patterns in collusive network: Understanding connectivity patterns in an underlying network is a well-studied problem in the literature. It include tasks such as inferring lockstep behavior [88], dense block detection [249], detecting core users [227, 250], identifying the most relevant actors in a network [251], sudden appearance/disappearance of links [252] etc. It would be interesting to create various networks among the users/entities present in the dataset to investigate various structural patterns of the network. The ABOME dataset can be represented as networks based on the actions performed by the collusive users on the content of other users of the services. It could be then use to study multiple diffusion modeling tasks such as influence maximization (selecting a seed set to maximize the influence spread) [253, 254], predicting information cascade [255, 256], measuring message propagation and social influence [257, 258] etc.

Modeling temporal evolution of collusive entities: The current thesis focuses mostly on analyzing the metadata and network properties of collusive entities collected from blackmarket services. However, little is studied about the temporal properties of these entities. As the dataset released with this thesis includes the time-series information of collusive entities, it can be used for various temporal modeling tasks such as detecting time periods containing unusual activity [63], identifying repetitive patterns in time-evolving graphs [259] etc.

Collusion in other online media platforms: Although this thesis makes effort in detecting and analyzing collusive entities in Twitter and YouTube, it would be interesting to see more studies on collusive entities targeted towards other online media platforms. However, we believe that data collection from platforms other than Twitter and YouTube will be challenging due to the limitations and restrictions imposed by the APIs provided by these platforms. Data collected from multiple online media platforms may consist of information from many major events [260], which can be further used for event-centric studies. Researchers can also check how these users/entities were involved in manipulating the popularity of events by artificially inflating the social growth of users/entities in online media [261]. It would also be interesting to connect multiplatform data for better detection of blackmarket users. However, we strongly believe that each platform has its own unique characteristics. This makes it a challenging task to combine the behaviors collected from different platforms. One strategy could be to propose a similarity measure to link content across multiple platforms. This would also help to track the evolution of the content across platforms.

Links to codes and datasets

I https://github.com/LCS2-IIITD/collusive-retweeters-ASONAM-2018

II https://github.com/LCS2-IIITD/HawkesEye

III https://github.com/LCS2-IIITD/CoReRank-WSDM-2019

IV https://github.com/LCS2-IIITD/CollATe

V https://github.com/uditarora/collusive-retweeters-TIST-2020

VI https://github.com/LCS2-IIITD/DECIFE

VII https://zenodo.org/record/4437987

VIII http://tiny.cc/SCoRe

References

[1] M. Giatsoglou, D. Chatzakou, N. Shah, A. Beutel, C. Faloutsos, and A. Vakali, "Nd-sync: Detecting synchronized fraud activities," in *PAKDD*, 2015, pp. 201–214.

[2] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *JMLR*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[3] H. S. Dutta, A. Chetan, B. Joshi, and T. Chakraborty, "Retweet us, we will retweet you: Spotting collusive retweeters involved in blackmarket services," in *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 2018, pp. 242–249.

[4] A. Chetan, B. Joshi, H. S. Dutta, and T. Chakraborty, "Corerank: Ranking to detect users involved in blackmarket-based collusive retweeting activities," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. ACM, 2019, pp. 330–338.

[5] H. S. Dutta and T. Chakraborty, "Blackmarket-driven collusion among retweeters–analysis, detection and characterization," *IEEE Transactions on Information Forensics and Security*, 2019.

[6] N. Shah, H. Lamba, A. Beutel, and C. Faloutsos, "The many faces of link fraud," in *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2017, pp. 1069–1074.

[7] H. S. Dutta, M. Jobanputra, H. Negi, and T. Chakraborty, "Detecting and analyzing collusive entities on youtube," *arxiv:2005.06243*, 2020.

[8] J. Kim, B. Tabibian, A. Oh, B. Schölkopf, and M. Gomez-Rodriguez, "Leveraging the crowd to detect and reduce the spread of fake news and misinformation," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, 2018, pp. 324–332.

[9] J. Castellini, V. Poggioni, and G. Sorbi, "Fake twitter followers detection by denoising autoencoder," in *Proceedings of the International Conference on Web Intelligence*. ACM, 2017, pp. 195–202.

[10] S. Kumar, J. Cheng, J. Leskovec, and V. Subrahmanian, "An army of me: Sockpuppets in online discussion communities," in *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2017, pp. 857–866.

[11] H. Li, A. Mukherjee, B. Liu, R. Kornfield, and S. Emery, "Detecting campaign promoters on twitter using markov random fields," in *2014 IEEE International Conference on Data Mining*. IEEE, 2014, pp. 290–299.

[12] G. Gee and H. Teh, "Twitter spammer profile detection," *Available online: cs229. stanford. edu/proj2010/GeeTeh-Twitter Spammer Profile Detection. pdf*, 2010.

[13] K. Lee, B. D. Eoff, and J. Caverlee, "Seven months with the devils: A long-term study of content polluters on twitter," in *Fifth international AAAI conference on weblogs and social media*, 2011.

[14] E. De Cristofaro, A. Friedman, G. Jourjon, M. A. Kaafar, and M. Z. Shafiq, "Paying for likes?: Understanding facebook like fraud using honeypots," in *Proceedings of the 2014 Conference on Internet Measurement Conference*. ACM, 2014, pp. 129–136.

[15] P. R. Badri Satya, K. Lee, D. Lee, T. Tran, and J. J. Zhang, "Uncovering fake likers in online social networks," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 2016, pp. 2365–2370.

[16] M. Ikram, L. Onwuzurike, S. Farooqi, E. D. Cristofaro, A. Friedman, G. Jourjon, M. A. Kaafar, and M. Z. Shafiq, "Measuring, characterizing, and detecting facebook like farms," *ACM Transactions on Privacy and Security (TOPS)*, vol. 20, no. 4, p. 13, 2017.

[17] I. Sen, A. Aggarwal, S. Mian, S. Singh, P. Kumaraguru, and A. Datta, "Worth its weight in likes: towards detecting fake likes on instagram," in *Proceedings of the 10th ACM Conference on Web Science*. ACM, 2018, pp. 205–209.

[18] W. Zhang and H.-M. Sun, "Instagram spam detection," in *2017 IEEE 22nd Pacific Rim International Symposium on Dependable Computing (PRDC)*. IEEE, 2017, pp. 227–228.

[19] M. Egele, G. Stringhini, C. Kruegel, and G. Vigna, "Towards detecting compromised accounts on social networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 14, no. 4, pp. 447–460, 2015.

[20] A. Gupta, H. Lamba, P. Kumaraguru, and A. Joshi, "Faking sandy: characterizing and identifying fake images on twitter during hurricane sandy," in *Proceedings of the 22nd international conference on World Wide Web*. ACM, 2013, pp. 729–736.

[21] G. Stringhini, P. Mourlanne, G. Jacob, M. Egele, C. Kruegel, and G. Vigna, "{EVILCOHORT}: Detecting communities of malicious accounts on online services," in *24th {USENIX} Security Symposium*, 2015, pp. 563–578.

[22] S. Ghosh, B. Viswanath, F. Kooti, N. K. Sharma, G. Korlam, F. Benevenuto, N. Ganguly, and K. P. Gummadi, "Understanding and combating link farming in the twitter social network," in *Proceedings of the 21st international conference on World Wide Web*. ACM, 2012, pp. 61–70.

[23] K. Thomas, C. Grier, D. Song, and V. Paxson, "Suspended accounts in retrospect: an analysis of twitter spam," in *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*. ACM, 2011, pp. 243–258.

[24] A. H. Wang, "Don't follow me: Spam detection in twitter," in *2010 international conference on security and cryptography (SECRYPT)*. IEEE, 2010, pp. 1–10.

[25] S. Nilizadeh, F. Labrèche, A. Sedighian, A. Zand, J. Fernandez, C. Kruegel, G. Stringhini, and G. Vigna, "Poised: Spotting twitter spam off the beaten paths," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017, pp. 1159–1174.

[26] M. Egele, G. Stringhini, C. Kruegel, and G. Vigna, "Compa: Detecting compromised accounts on social networks." in *NDSS*, 2013, pp. 1–17.

[27] S. Kumar and N. Shah, "False information on web and social media: A survey," *arXiv preprint arXiv:1804.08559*, 2018.

[28] M. Alsaleh, A. Alarifi, F. Al-Quayed, and A. Al-Salman, "Combating comment spam with machine learning approaches," in *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2015, pp. 295–300.

[29] A. K. Uysal, "Feature selection for comment spam filtering on youtube," *Data Science and Applications*, vol. 1, no. 1, pp. 4–8, 2018.

[30] Y. Li, O. Martinez, X. Chen, Y. Li, and J. E. Hopcroft, "In a world that counts: Clustering and detecting fake social engagement at scale," in *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2016, pp. 111–120.

[31] M. Marciel, R. Cuevas, A. Banchs, R. González, S. Traverso, M. Ahmed, and A. Azcorra, "Understanding the detection of view fraud in video content portals," in *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2016, pp. 357–368.

[32] L. Chen, Y. Zhou, and D. M. Chiu, "Analysis and detection of fake views in online video services," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 11, no. 2s, p. 44, 2015.

[33] A. Metwally, D. Agrawal, and A. El Abbadi, "Detectives: detecting coalition hit inflation attacks in advertising networks streams," in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 241–250.

[34] V. Dave, S. Guha, and Y. Zhang, "Measuring and fingerprinting click-spam in ad networks," in *Proceedings of the ACM SIGCOMM 2012 conference on Applications, technologies, architectures, and protocols for computer communication*. ACM, 2012, pp. 175–186.

[35] M. N. Hussain, S. Tokdemir, N. Agarwal, and S. Al-Khateeb, "Analyzing disinformation and crowd manipulation tactics on youtube," in *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 2018, pp. 1092–1095.

[36] M. Alassad, N. Agarwal, and M. N. Hussain, "Examining intensive groups in youtube commenter networks," in *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*. Springer, 2019, pp. 224–233.

[37] M. Faddoul, G. Chaslot, and H. Farid, "A longitudinal analysis of youtube's promotion of conspiracy videos," *arXiv preprint arXiv:2003.03318*, 2020.

[38] G. Yang, N. Z. Gong, and Y. Cai, "Fake co-visitation injection attacks to recommender systems." in *NDSS*, 2017, pp. 1–17.

[39] T. C. Alberto, J. V. Lochter, and T. A. Almeida, "Tubespam: Comment spam filtering on youtube," in *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2015, pp. 138–143.

[40] Y. Yusof and O. H. Sadoon, "Detecting video spammers in youtube social media," in *Proceedings of International Conference on Computing and Informatics*, 2017, pp. 228–234.

[41] R. Chowdury, M. N. M. Adnan, G. Mahmud, and R. M. Rahman, "A data mining based spam detection system for youtube," in *Eighth International Conference on Digital Information Management (ICDIM 2013)*. IEEE, 2013, pp. 373–378.

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

[43] S. Aiyar and N. P. Shetty, "N-gram assisted youtube spam comment detection," *Procedia computer science*, vol. 132, pp. 174–182, 2018.

[44] D. Y. Zhang, Q. Li, H. Tong, J. Badilla, Y. Zhang, and D. Wang, "Crowdsourcing-based copyright infringement detection in live video streams," in *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 2018, pp. 367–374.

[45] D. Y. Zhang, J. Badilla, H. Tong, and D. Wang, "An end-to-end scalable copyright detection system for online video sharing platforms," in *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 2018, pp. 626–629.

[46] A. Acker, "Data craft: the manipulation of social media metadata," *Data & Society Research Institute*, 2018.

[47] M. H. Keller, "The flourishing business of fake youtube views," *The New York Times*, 2018.

[48] U. Arora, W. S. Paka, and T. Chakraborty, "Multitask learning for blackmarket tweet detection," in *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 2019, pp. 127–130.

[49] S. Dhawan, S. C. R. Gangireddy, S. Kumar, and T. Chakraborty, "Spotting collusive behaviour of online fraud groups in customer reviews," in *IJCAI*, 2019, pp. 245–251.

[50] G. Wang, S. Xie, B. Liu, and S. Y. Philip, "Review graph based online store review spammer detection," in *2011 IEEE 11th International Conference on Data Mining*. IEEE, 2011, pp. 1242–1247.

[51] Z.-h. Zhu, Y. Zhi, and Y.-f. Dai, "A new approach to detect user collusion behavior in online qa system," in *International Conference on Computer Networks and Communication Technology (CNCT 2016)*. Atlantis Press, 2016, pp. 836–842.

[52] S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi, "Fame for sale: Efficient detection of fake twitter followers," *Decision Support Systems*, vol. 80, pp. 56–71, 2015.

[53] A. Aggarwal and P. Kumaraguru, "Followers or phantoms? an anatomy of purchased twitter followers," *arXiv preprint arXiv:1408.1534*, 2014.

[54] G. Stringhini, G. Wang, M. Egele, C. Kruegel, G. Vigna, H. Zheng, and B. Y. Zhao, "Follow the green: growth and dynamics in twitter follower markets," in *Proceedings of the 2013 conference on Internet measurement conference*. ACM, 2013, pp. 163–176.

[55] A. Mehrotra, M. Sarreddy, and S. Singh, "Detection of fake twitter followers using graph centrality measures," in *2016 2nd International Conference on Contemporary Computing and Informatics (IC3I)*. IEEE, 2016, pp. 499–504.

[56] M. Jiang, P. Cui, A. Beutel, C. Faloutsos, and S. Yang, "Catchsync: catching synchronized behavior in large directed graphs," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 941–950.

[57] C. A. Davis, O. Varol, E. Ferrara, A. Flammini, and F. Menczer, "Botornot: A system to evaluate social bots," in *WWW*, 2016, pp. 273–274.

[58] A. ElAzab, A. M. Idrees, M. A. Mahmoud, and H. Hefny, "Fake accounts detection in twitter based on minimum weighted feature," in *18th International Conference on Document Analysis and Recognition*, 2016, pp. 1–6.

[59] A. Aggarwal, S. Kumar, K. Bhargava, and P. Kumaraguru, "The follower count fallacy: Detecting twitter users with manipulated follower count," in *ACM SAC*, 2018.

[60] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida, "Detecting spammers on twitter," in *Collaboration, electronic messaging, anti-abuse and spam conference (CEAS)*, vol. 6, no. 2010, 2010, p. 12.

[61] S. Lee and J. Kim, "Warningbird: Detecting suspicious urls in twitter stream." in *NDSS*, vol. 12, 2012, pp. 1–13.

[62] K. Lee, J. Mahmud, J. Chen, M. Zhou, and J. Nichols, "Who will retweet this? detecting strangers from twitter to retweet information," *ACM TIST*, vol. 6, no. 3, p. 31, 2015.

[63] M. Giatsoglou, D. Chatzakou, N. Shah, C. Faloutsos, and A. Vakali, "Retweeting activity on twitter: Signs of deception," in *PAKDD*. Springer, 2015, pp. 122–134.

[64] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia, "Detecting automation of twitter accounts: Are you a human, bot, or cyborg?" *IEEE TDSC*, vol. 9, no. 6, pp. 811–824, 2012.

[65] E. Ferrara, O. Varol, C. Davis, F. Menczer, and A. Flammini, "The rise of social bots," *Comm. of the ACM*, vol. 59, no. 7, pp. 96–104, 2016.

[66] X. Hu, J. Tang, Y. Zhang, and H. Liu, "Social spammer detection in microblogging." in *IJCAI*, vol. 13, 2013, pp. 2633–2639.

[67] S. Haustein, T. D. Bowman, K. Holmberg, A. Tsou, C. R. Sugimoto, and V. Larivière, "Tweets as impact indicators: Examining the implications of automated "bot" accounts on twitter," *Journal of the Association for Information Science and Technology*, vol. 67, no. 1, pp. 232–238, 2016.

[68] G. Wang, C. Wilson, X. Zhao, Y. Zhu, M. Mohanlal, H. Zheng, and B. Y. Zhao, "Serf and turf: crowdturfing for fun and profit," in *WWW*. ACM, 2012, pp. 679–688.

[69] M. Jiang, P. Cui, A. Beutel, C. Faloutsos, and S. Yang, "Detecting suspicious following behavior in multimillion-node social networks," in *WWW*. ACM, 2014, pp. 305–306.

[70] S. Gupta, A. Khattar, A. Gogia, P. Kumaraguru, and T. Chakraborty, "Collective classification of spam campaigners on twitter: A hierarchical meta-path based approach," in *WWW*, 2018, pp. 529–538.

[71] D. M. Cook, B. Waugh, M. Abdipanah, O. Hashemi, and S. A. Rahman, "Twitter deception and influence: Issues of identity, slacktivism, and puppetry," *Journal of Information Warfare*, vol. 13, no. 1, pp. 58–IV, 2014.

[72] S. Kumar, J. Cheng, J. Leskovec, and V. Subrahmanian, "An army of me: Sockpuppets in online discussion communities," in *WWW*, 2017, pp. 857–866.

[73] S.-J. Lin, Y.-Y. Jheng, and C.-H. Yu, "Combining ranking concept and social network analysis to detect collusive groups in online auctions," *Expert Systems with Applications*, vol. 39, no. 10, pp. 9079–9086, 2012.

[74] C. De Micheli and A. Stroppa, "Twitter and the underground market," in *11th Nexa Lunch Seminar*, vol. 22, 2013.

[75] K. Thomas, D. McCoy, C. Grier, A. Kolcz, and V. Paxson, "Trafficking fraudulent accounts: The role of the underground market in twitter spam and abuse." in *USENIX Security Symposium*, 2013, pp. 195–210.

[76] Y. Liu, Y. Liu, M. Zhang, and S. Ma, "Pay me and i'll follow you: Detection of crowdturfing following activities in microblog environment." in *IJCAI*, 2016, pp. 3789–3796.

[77] M. Motoyama, D. McCoy, K. Levchenko, S. Savage, and G. M. Voelker, "An analysis of underground forums," in *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*. ACM, 2011, pp. 71–80.

[78] N. Shah, A. Beutel, B. Gallagher, and C. Faloutsos, "Spotting suspicious link behavior with fbox: An adversarial perspective," in *2014 IEEE International Conference on Data Mining*. IEEE, 2014, pp. 959–964.

[79] T. Rastogi, "A power law approach to estimating fake social network accounts," *arXiv preprint arXiv:1605.07984*, 2016.

[80] Y. Zhang and J. Lu, "Discover millions of fake followers in weibo," *Social Network Analysis and Mining*, vol. 6, no. 1, p. 16, 2016.

[81] H. Wang and J. Lu, "Detect inflated follower numbers in osn using star sampling," in *2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2013)*. IEEE, 2013, pp. 127–133.

[82] M. Eftekhar and N. Koudas, "Some research opportunities on twitter advertising." *IEEE Data Eng. Bull.*, vol. 36, no. 3, pp. 77–82, 2013.

[83] Q. Cao, M. Sirivianos, X. Yang, and T. Pregueiro, "Aiding the detection of fake accounts in large scale social online services," in *USENIX*, 2012, pp. 15–15.

[84] A. H. Wang, "Detecting spam bots in online social networking sites: a machine learning approach," in *IFIP Annual Conference on Data and Applications Security and Privacy*. Springer, 2010, pp. 335–342.

[85] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia, "Who is tweeting on twitter: human, bot, or cyborg?" in *Proceedings of the 26th annual computer security applications conference*. ACM, 2010, pp. 21–30.

[86] S. Gianvecchio, M. Xie, Z. Wu, and H. Wang, "Measurement and classification of humans and bots in internet chat." in *USENIX security symposium*, 2008, pp. 155–170.

[87] B. Wang, A. Zubiaga, M. Liakata, and R. Procter, "Making the most of tweet-inherent features for social spam detection on twitter," *arXiv preprint arXiv:1503.07405*, 2015.

[88] A. Beutel, W. Xu, V. Guruswami, C. Palow, and C. Faloutsos, "Copycatch: stopping group attacks by spotting lockstep behavior in social networks," in *Proceedings of the 22nd international conference on World Wide Web*. ACM, 2013, pp. 119–130.

[89] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Predicting positive and negative links in online social networks," in *Proceedings of the 19th international conference on World wide web*. ACM, 2010, pp. 641–650.

[90] N. Jindal and B. Liu, "Review spam detection," in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 1189–1190.

[91] S. Ahmad, A. Pathak, and S. Jaiswal, "A survey about spam detection and analysis using users' reviews," *Malaya Journal of Matematik (MJM)*, no. 1, 2018, pp. 1–4, 2018.

[92] M. Singh, D. Bansal, and S. Sofat, "Followers or fradulents? an analysis and classification of twitter followers market merchants," *Cybernetics and Systems*, vol. 47, no. 8, pp. 674–689, 2016.

[93] S. Yang, H. Jin, B. Li, and X. Liao, "A modeling framework of content pollution in peer-to-peer video streaming systems," *Computer networks*, vol. 53, no. 15, pp. 2703–2715, 2009.

[94] N. Shah, "Flock: Combating astroturfing on livestreaming platforms," in *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2017, pp. 1083–1091.

[95] M. N. Reddy, T. Mamatha, and A. Balaram, "Analysis of e-recruitment systems and detecting e-recruitment fraud," in *International Conference on Communications and Cyber Physical Engineering 2018*. Springer, 2018, pp. 411–417.

[96] N. Shah, H. Lamba, A. Beutel, and C. Faloutsos, "OEC: open-ended classification for future-proof link-fraud detection," *CoRR*, vol. abs/1704.01420, 2017. [Online]. Available: http://arxiv.org/abs/1704.01420

[97] R. Ghosh, T. Surachawala, and K. Lerman, "Entropy-based classification of'retweeting'activity on twitter," *arXiv preprint arXiv:1106.0346*, 2011.

[98] A. G. Hawkes, "Spectra of some self-exciting and mutually exciting point processes," *Biometrika*, vol. 58, no. 1, pp. 83–90, 1971. [Online]. Available: http://www.jstor.org/stable/2334319

[99] Q. Zhao, M. A. Erdogdu, H. Y. He, A. Rajaraman, and J. Leskovec, "SEISMIC: A self-exciting point process model for predicting tweet popularity," *CoRR*, vol. abs/1506.02594, 2015. [Online]. Available: http://arxiv.org/abs/1506.02594

[100] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003. [Online]. Available: http://dl.acm.org/citation.cfm?id=944919.944937

[101] K. Lee, J. Caverlee, and S. Webb, "Uncovering social spammers: social honeypots+ machine learning," in *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2010, pp. 435–442.

[102] F. Benevenuto, T. Rodrigues, V. Almeida, J. Almeida, and M. Gonçalves, "Detecting spammers and content promoters in online video social networks," in *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2009, pp. 620–627.

[103] N. Chavoshi, H. Hamooni, and A. Mueen, "Debot: Twitter bot detection via warped correlation." in *ICDM*, 2016, pp. 817–822.

[104] A. Gupta, H. Lamba, P. Kumaraguru, and A. Joshi, "Faking sandy: characterizing and identifying fake images on twitter during hurricane sandy," in *Proceedings of the 22nd international conference on World Wide Web*. ACM, 2013, pp. 729–736.

[105] N. Vo, K. Lee, C. Cao, T. Tran, and H. Choi, "Revealing and detecting malicious retweeter groups," in *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*. ACM, 2017, pp. 363–368.

[106] S. Gupta, P. Kumaraguru, and T. Chakraborty, "Malreg: Detecting and analyzing malicious retweeter groups," in *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*, ser. CoDS-COMAD '19. New York, NY, USA: ACM, 2019, pp. 61–69. [Online]. Available: http://doi.acm.org/10.1145/3297001.3297009

[107] D. Yuan, Y. Miao, N. Z. Gong, Z. Yang, Q. Li, D. Song, Q. Wang, and X. Liang, "Detecting fake accounts in online social networks at the time of registrations," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2019, pp. 1423–1438.

[108] M. BalaAnand, N. Karthikeyan, S. Karthik, R. Varatharajan, G. Manogaran, and C. Sivaparthipan, "An enhanced graph-based semi-supervised learning algorithm to detect fake users on twitter," *The Journal of Supercomputing*, vol. 75, no. 9, pp. 6085–6105, 2019.

[109] E. Van Der Walt and J. Eloff, "Using machine learning to detect fake identities: bots vs humans," *IEEE Access*, vol. 6, pp. 6540–6549, 2018.

[110] M. M. Swe and N. N. Myo, "Fake accounts detection on twitter using blacklist," in *2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS)*. IEEE, 2018, pp. 562–566.

[111] A. Hassan Zadeh and R. Sharda, *Hawkes Point Processes for Social Media Analytics*, 2014.

[112] J. R. Zipkin, F. P. Schoenberg, K. Cornges, and A. L. Bertozzi, "Point-process models of social network interactions: Parameter estimation and missing data recovery," *European Journal of Applied Mathematics*, vol. 27, no. 3, p. 502–529, 2016.

[113] S. Gao, J. Ma, and Z. Chen, "Modeling and predicting retweeting dynamics on microblogging platforms," in *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, ser. WSDM '15. New York, NY, USA: ACM, 2015, pp. 107–116. [Online]. Available: http://doi.acm.org/10.1145/2684822.2685303

[114] R. Kobayashi and R. Lambiotte, "Tideh: Time-dependent hawkes process for predicting retweet dynamics," *CoRR*, vol. abs/1603.09449, 2016. [Online]. Available: http://arxiv.org/abs/1603.09449

[115] F. Chen, W. H. Tan *et al.*, "Marked self-exciting point process modelling of information diffusion on twitter," vol. 12, no. 4. Institute of Mathematical Statistics, 2018, pp. 2175–2196.

[116] M. Farajtabar, Y. Wang, M. Gomez-Rodriguez, S. Li, H. Zha, and L. Song, "Coevolve: A joint point process model for information diffusion and network co-evolution," in *Journal of Machine Learning Research 18 (2017) 1-49*, 2017.

[117] M.-A. Rizoiu, L. Xie, S. Sanner, M. Cebrian, H. Yu, and P. Van Hentenryck, "Expecting to be hip: Hawkes intensity processes for social media popularity," in *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2017, pp. 735–744.

[118] M. Lukasik, P. K. Srijith, D. Vu, K. Bontcheva, A. Zubiaga, and T. Cohn, "Hawkes processes for continuous time sequence classification: an application to rumour stance classification in twitter," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, 2016, pp. 393–398. [Online]. Available: http://aclweb.org/anthology/P16-2064

[119] M.-A. Rizoiu, Y. Lee, and S. Mishra, *A Tutorial on Hawkes Processes for Events in Social Media*, 12 2017, pp. 191–218.

[120] A. Steinskog, J. Therkelsen, and B. Gambäck, "Twitter topic modeling by tweet aggregation," in *Proceedings of the 21st nordic conference on computational linguistics*, 2017, pp. 77–86.

[121] J. Weng, E.-P. Lim, J. Jiang, and Q. He, "Twitterrank: Finding topic-sensitive influential twitterers," in *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, ser. WSDM '10. New York, NY, USA: ACM, 2010, pp. 261–270. [Online]. Available: http://doi.acm.org/10.1145/1718487.1718520

[122] H. Mei and J. Eisner, "The neural hawkes process: A neurally self-modulating multivariate point process," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, 2017, pp. 6754–6764.

[123] P. J. Laub, T. Taimre, and P. K. Pollett, "Hawkes processes," *arXiv preprint arXiv:1507.02822*, 2015.

[124] T. Ozaki, "Maximum likelihood estimation of hawkes' self-exciting point processes," *Annals of the Institute of Statistical Mathematics*, vol. 31, no. 1, pp. 145–155, Dec 1979. [Online]. Available: https://doi.org/10.1007/BF02480272

[125] M. Hoffman, F. R. Bach, and D. M. Blei, "Online learning for latent dirichlet allocation," in *Advances in Neural Information Processing Systems 23*, J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, Eds. Curran Associates, Inc., 2010, pp. 856–864.

[126] E. Bacry, M. Bompaire, S. Gaïffas, and S. Poulsen, "tick: a Python library for statistical learning, with a particular emphasis on time-dependent modeling," *ArXiv e-prints*, Jul. 2017.

[127] M. Hoffman, F. R. Bach, and D. M. Blei, "Online learning for latent dirichlet allocation," in *Advances in neural information processing systems*, 2010, pp. 856–864.

[128] Y. Ogata, "On lewis' simulation method for point processes," in *IEEE Transactions on Information Theory, 27(1): 23–31*, 1981.

[129] A. H. Wang, "Detecting spam bots in online social networking sites: A machine learning approach," in *Proceedings of the 24th Annual IFIP WG 11.3 Working Conference on Data and Applications Security and Privacy*, ser. DBSec'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 335–342. [Online]. Available: http://dl.acm.org/citation.cfm?id=1875947.1875979

[130] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: http://dx.doi.org/10.1162/neco.1997.9.8.1735

[131] S. Gupta, A. Khattar, A. Gogia, P. Kumaraguru, and T. Chakraborty, "Collective classification of spam campaigners on twitter: A hierarchical meta-path based approach," *arXiv preprint arXiv:1802.04168*, 2018.

[132] J. P. Dickerson, V. Kagan, and V. Subrahmanian, "Using sentiment to detect bots on twitter: Are humans more opinionated than bots?" in *Proceedings of the 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. IEEE Press, 2014, pp. 620–627.

[133] D. Wang, S. B. Navathe, L. Liu, D. Irani, A. Tamersoy, and C. Pu, "Click traffic analysis of short url spam on twitter," in *Collaborative Computing: Networking, Applications and Worksharing (Collaboratecom), 2013 9th International Conference Conference on*. IEEE, 2013, pp. 250–259.

[134] E. Lancaster, T. Chakraborty, and V. Subrahmanian, "Maltp: Parallel prediction of malicious tweets," *IEEE Transactions on Computational Social Systems*, 2018.

[135] H. Gao, Y. Chen, K. Lee, D. Palsetia, and A. N. Choudhary, "Towards online spam filtering in social networks." in *NDSS*, vol. 12, 2012, pp. 1–16.

[136] C. Grier, K. Thomas, V. Paxson, and M. Zhang, "@ spam: the underground on 140 characters or less," in *ACM CCS*. ACM, 2010, pp. 27–37.

[137] F. Li, M. Huang, Y. Yang, and X. Zhu, "Learning to identify review spam," in *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, vol. 22, no. 3, 2011, p. 2488.

[138] X. Zheng, Y. M. Lai, K.-P. Chow, L. C. Hui, and S.-M. Yiu, "Sockpuppet detection in online discussion forums," in *Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), 2011 Seventh International Conference on*. IEEE, 2011, pp. 374–377.

[139] T. Solorio, R. Hasan, and M. Mizan, "Sockpuppet detection in wikipedia: A corpus of real-world deceptive writing for linking identities," *arXiv preprint arXiv:1310.6772*, 2013.

[140] S. Gupta, P. Kumaraguru, and T. Chakraborty, "Malreg: Detecting and analyzing malicious retweeter groups," in *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*. ACM, 2019, pp. 61–69.

[141] L. Chen, A. Mislove, and C. Wilson, "An empirical analysis of algorithmic pricing on amazon marketplace," in *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2016, pp. 1339–1349.

[142] A. Hannák, C. Wagner, D. Garcia, A. Mislove, M. Strohmaier, and C. Wilson, "Bias in online freelance marketplaces: Evidence from taskrabbit and fiverr." in *CSCW*, 2017, pp. 1914–1933.

[143] S. Vidros, C. Kolias, G. Kambourakis, and L. Akoglu, "Automatic detection of online recruitment frauds: Characteristics, methods, and a public dataset," *Future Internet*, vol. 9, no. 1, p. 6, 2017.

[144] F. Abel, Q. Gao, G.-J. Houben, and K. Tao, "Analyzing user modeling on twitter for personalized news recommendations," in *International Conference on User Modeling, Adaptation, and Personalization*. Springer, 2011, pp. 1–12.

[145] Z. Xu, Y. Zhang, Y. Wu, and Q. Yang, "Modeling user posting behavior on social media," in *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2012, pp. 545–554.

[146] A. Ritter, C. Cherry, and B. Dolan, "Unsupervised modeling of twitter conversations," in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 172–180.

[147] A. Benton, R. Arora, and M. Dredze, "Learning multiview embeddings of twitter users," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, vol. 2, 2016, pp. 14–19.

[148] T. Ding, W. K. Bickel, and S. Pan, "Multi-view unsupervised user feature embedding for social media-based substance use prediction," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 2275–2284.

[149] B. Dhingra, Z. Zhou, D. Fitzpatrick, M. Muehl, and W. Cohen, "Tweet2vec: Character-based distributed representations for social media," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, vol. 2, 2016, pp. 269–274.

[150] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[151] M. van de Velden, "On generalized canonical correlation analysis," in *Proceedings of the 58th World Statistical Congress*, 2011.

[152] J. R. Kettenring, "Canonical analysis of several sets of variables," *Biometrika*, vol. 58, no. 3, pp. 433–451, 1971.

[153] P. M. Robinson, "Generalized canonical analysis for time series," *Journal of Multivariate Analysis*, vol. 3, no. 2, pp. 141–160, 1973.

[154] A. Tenenhaus and M. Tenenhaus, "Regularized generalized canonical correlation analysis," *Psychometrika*, vol. 76, no. 2, p. 257, 2011.

[155] J. D. Carroll, "Generalization of canonical correlation analysis to three or more sets of variables," in *Proceedings of the 76th annual convention of the American Psychological Association*, vol. 3, 1968, pp. 227–228.

[156] A. G. Karegowda, A. Manjunath, and M. Jayaram, "Comparative study of attribute selection using gain ratio and correlation based feature selection," *International Journal of Information Technology and Knowledge Management*, vol. 2, no. 2, pp. 271–277, 2010.

[157] C. Yang, R. Harkreader, J. Zhang, S. Shin, and G. Gu, "Analyzing spammers' social networks for fun and profit: a case study of cyber criminal ecosystem on twitter," in *Proceedings of the 21st international conference on World Wide Web*. ACM, 2012, pp. 71–80.

[158] H. Kwak, C. Lee, H. Park, and S. Moon, "What is twitter, a social network or a news media?" in *Proceedings of the 19th international conference on World wide web*. ACM, 2010, pp. 591–600.

[159] Z. Miller, B. Dickinson, W. Deitrick, W. Hu, and A. H. Wang, "Twitter spammer detection using data stream clustering," *Information Sciences*, vol. 260, pp. 64–73, 2014.

[160] F. Ahmed and M. Abulaish, "A generic statistical approach for spam detection in online social networks," *Computer Communications*, vol. 36, no. 10-11, pp. 1120–1129, 2013.

[161] B. Hooi, N. Shah, A. Beutel, S. Günnemann, L. Akoglu, M. Kumar, D. Makhija, and C. Faloutsos, "Birdnest: Bayesian inference for ratings-fraud detection," in *Proceedings of the 2016 SIAM International Conference on Data Mining*. SIAM, 2016, pp. 495–503.

[162] S. Kumar, B. Hooi, D. Makhija, M. Kumar, C. Faloutsos, and V. Subrahmanian, "Rev2: Fraudulent user prediction in rating platforms," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, ser. WSDM '18. New York, NY, USA: ACM, 2018, pp. 333–341. [Online]. Available: http://doi.acm.org/10.1145/3159652.3159729

[163] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[164] W. McKinney, "pandas: a foundational python library for data analysis and statistics," *Python for High Performance and Scientific Computing*, pp. 1–9, 2011.

[165] S. Vosoughi, P. Vijayaraghavan, and D. Roy, "Tweet2vec: Learning tweet embeddings using character-level cnn-lstm encoder-decoder," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 2016, pp. 1041–1044.

[166] D. Q. Nguyen, T. Vu, and A. T. Nguyen, "Bertweet: A pre-trained language model for english tweets," *arXiv preprint arXiv:2005.10200*, 2020.

[167] G. Stringhini, G. Wang, M. Egele, C. Kruegel, G. Vigna, H. Zheng, and B. Y. Zhao, "Follow the green: growth and dynamics in twitter follower markets," in *Proceedings of the 2013 conference on Internet measurement conference*, 2013, pp. 163–176.

[168] N. Shah, H. Lamba, A. Beutel, and C. Faloutsos, "The many faces of link fraud," in *IEEE ICDM*, 2017, pp. 1069–1074.

[169] U. Arora, H. S. Dutta, B. Joshi, A. Chetan, and T. Chakraborty, "Analyzing and detecting collusive users involved in blackmarket retweeting activities," *ACM TIST*, vol. 11, no. 3, pp. 1–24, 2020.

[170] J. Castellini, V. Poggioni, and G. Sorbi, "Fake twitter followers detection by denoising autoencoder," in *WI*, 2017, pp. 195–202.

[171] A. Aggarwal and P. Kumaraguru, "What they do in shadows: Twitter underground follower market," in *IEEE PST*, 2015, pp. 93–100.

[172] S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi, "Fame for sale: Efficient detection of fake twitter followers," *Decision Support Systems*, 2015.

[173] A. Mehrotra, M. Sarreddy, and S. Singh, "Detection of fake twitter followers using graph centrality measures," in *IEEE IC3I*, 2016, pp. 499–504.

[174] Lee, Mahmud, Chen, Zhou, and Nichols, "Who will retweet this? detecting strangers from twitter to retweet information," *ACM TIST*, 2015.

[175] N. Shah, A. Beutel, B. Gallagher, and C. Faloutsos, "Spotting suspicious link behavior with fbox: An adversarial perspective," in *IEEE ICDM*, 2014, pp. 959–964.

[176] S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi, "A fake follower story: improving fake accounts detection on twitter," *IIT-CNR, Tech. Rep. TR-03*, 2014.

[177] M. Jiang, P. Cui, A. Beutel, C. Faloutsos, and S. Yang, "Catchsync: catching synchronized behavior in large directed graphs," in *ACM SIGKDD*, 2014.

[178] H. Kwak, H. Chun, and S. Moon, "Fragile online relationship: a first look at unfollow dynamics in twitter," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2011, pp. 1091–1100.

[179] A. Aggarwal, S. Kumar, K. Bhargava, and P. Kumaraguru, "The follower count fallacy: detecting twitter users with manipulated follower count," in *SAC*. ACM, 2018, pp. 1748–1755.

[180] H. Shen and X. Liu, "Detecting spammers on twitter based on content and social interaction," in *2015 International Conference on Network and Information Systems for Computers*, Jan 2015, pp. 413–417.

[181] H. Li, A. Mukherjee, B. Liu, R. Kornfield, and S. Emery, "Detecting campaign promoters on twitter using markov random fields," ser. ICDM '14, 2014, pp. 290–299.

[182] E. Ferrara, O. Varol, C. Davis, F. Menczer, and A. Flammini, "The rise of social bots," *Commun. ACM*, pp. 96–104, 2016.

[183] Y. Shen, J. Yu, K. Dong, and K. Nan, "Automatic fake followers detection in chinese micro-blogging system," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2014, pp. 596–607.

[184] Y. Zhang and J. Lu, "Discover millions of fake followers in weibo," *Social Network Analysis and Mining*, vol. 6, no. 1, p. 16, 2016.

[185] Z. Zhang, F. Zou, L. Pan, B. Pei, and J. Li, "Detection of zombie followers in sina weibo," in *IEEE ICCC*, 2016, pp. 2476–2480.

[186] C. De Micheli and A. Stroppa, "Twitter and the underground market," in *11th Nexa Lunch Seminar*, vol. 22, 2013, pp. 1–6.

[187] J. Song, S. Lee, and J. Kim, "Crowdtarget: Target-based detection of crowdturfing in online social networks," in *ACM SIGSAC*, 2015, pp. 793–804.

[188] G. Stringhini, M. Egele, C. Kruegel, and G. Vigna, "Poultry markets: on the underground economy of twitter followers," *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 4, pp. 527–532, 2012.

[189] S. Farooqi, F. Zaffar, N. Leontiadis, and Z. Shafiq, "Measuring and mitigating oauth access token abuse by collusion networks," in *IMC*, 2017, pp. 355–368.

[190] J. Weerasinghe, B. Flanigan, A. Stein, D. McCoy, and R. Greenstadt, "The pod people: Understanding manipulation of social media popularity via reciprocity abuse," in *The WebConf*, 2020, pp. 1874–1884.

[191] M. Motoyama, D. McCoy, K. Levchenko, S. Savage, and G. M. Voelker, "An analysis of underground forums," in *ACM SIGCOMM*. ACM, 2011, pp. 71–80.

[192] K. e. a. Thomas, "Trafficking fraudulent accounts: The role of the underground market in twitter spam and abuse." in *USENIX Symposium*, 2013, pp. 195–210.

[193] M. Singh, D. Bansal, and S. Sofat, "Followers or fradulents? an analysis and classification of twitter followers market merchants," *Cybernetics and Systems*, vol. 47, no. 8, pp. 674–689, 2016.

[194] H. S. Dutta and T. Chakraborty, "Blackmarket-driven collusion on online media: a survey," *arXiv preprint arXiv:2008.13102*, 2020.

[195] Y. Sun and J. Han, "Mining heterogeneous information networks: a structural analysis approach," *Acm Sigkdd Explorations Newsletter*, vol. 14, no. 2, pp. 20–28, 2013.

[196] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft, "Deep one-class classification," in *International conference on machine learning*, 2018, pp. 4393–4402.

[197] M. Lui and T. Baldwin, "langid. py: An off-the-shelf language identification tool," in *Proceedings of the ACL 2012 system demonstrations*, 2012, pp. 25–30.

[198] M. Artetxe and H. Schwenk, "Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 597–610, 2019.

[199] M. Wang, L. Yu, D. Zheng, Q. Gan, Y. Gai, Z. Ye, M. Li, J. Zhou, Q. Huang, C. Ma *et al.*, "Deep graph library: Towards efficient and scalable deep learning on graphs." 2019.

[200] J. H. Wenpeng Yin and D. Roth, "Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach," in *EMNLP*, 2019. [Online]. Available: https://arxiv.org/abs/1909.00161

[201] H. S. Dutta, V. R. Dutta, A. Adhikary, and T. Chakraborty, "Hawkeseye: Detecting fake retweeters using hawkes process and topic modeling," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 2667–2678, 2020.

[202] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal, "Long short term memory networks for anomaly detection in time series," in *Proceedings*. Presses universitaires de Louvain, 2015, p. 89.

[203] G. Bagler, "Analysis of the airport network of india as a complex weighted network," *Physica A: Statistical Mechanics and its Applications*, vol. 387, no. 12, pp. 2972–2980, 2008.

[204] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar *et al.*, "Universal sentence encoder," *arXiv preprint arXiv:1803.11175*, 2018.

[205] D. Cer, M. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia, "SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation," in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Vancouver, Canada: Association for Computational Linguistics, Aug. 2017, pp. 1–14.

[206] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *Journal of machine learning research*, vol. 11, no. Dec, pp. 3371–3408, 2010.

[207] J. Zhang, Y. Zhang, L. Bai, and J. Han, "Lossless-constraint denoising based auto-encoders," *Signal Processing: Image Communication*, vol. 63, pp. 92 – 99, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0923596518301085

[208] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger, "From word embeddings to document distances," in *International conference on machine learning*, 2015, pp. 957–966.

[209] V. Bulakh, C. W. Dunn, and M. Gupta, "Identifying fraudulently promoted online videos," in *Proceedings of the 23rd International Conference on World Wide Web*, 2014, pp. 1111–1116.

[210] G. E. Box and D. A. Pierce, "Distribution of residual autocorrelations in autoregressive-integrated moving average time series models," *Journal of the American statistical Association*, vol. 65, no. 332, pp. 1509–1526, 1970.

[211] V. A. Sindagi and V. M. Patel, "A survey of recent advances in cnn-based single image crowd counting and density estimation," *Pattern Recognition Letters*, vol. 107, pp. 3–16, 2018.

[212] Z. Tu, W. Xie, D. Zhang, R. Poppe, R. C. Veltkamp, B. Li, and J. Yuan, "A survey of variational and cnn-based optical flow techniques," *Signal Processing: Image Communication*, vol. 72, pp. 9–24, 2019.

[213] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "Deepfm: a factorization-machine based neural network for ctr prediction," *arXiv preprint arXiv:1703.04247*, 2017.

[214] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[215] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 2008, pp. 413–422.

[216] M. Hubert, M. Debruyne, and P. J. Rousseeuw, "Minimum covariance determinant and extensions," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 10, no. 3, p. e1421, 2018.

[217] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, pp. 93–104.

[218] A. Kołcz and C. H. Teo, "Feature weighting for improved classifier robustness," in *CEAS'09: sixth conference on email and anti-spam*, 2009, pp. 1–10.

[219] Z. Bu, Z. Xia, and J. Wang, "A sock puppet detection algorithm on virtual spaces," *Knowledge-Based Systems*, vol. 37, pp. 366–377, 2013.

[220] K. Shu, S. Wang, and H. Liu, "Understanding user profiles on social media for fake news detection," in *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*. IEEE, 2018, pp. 430–435.

[221] T. Elmas, R. Overdorf, A. F. Özkalay, and K. Aberer, "Lateral astroturfing attacks on twitter trending topics," *arXiv preprint arXiv:1910.07783*, 2019.

[222] L. F. DeKoven, T. Pottinger, S. Savage, G. M. Voelker, and N. Leontiadis, "Following their footsteps: Characterizing account automation abuse and defenses," in *IMC*, 2018, pp. 43–55.

[223] V. Batagelj and M. Zaversnik, "An o (m) algorithm for cores decomposition of networks," *arXiv preprint cs/0310049*, 2003.

[224] M. P. Rombach, M. A. Porter, J. H. Fowler, and P. J. Mucha, "Core-periphery structure in networks," *SIAM Journal on Applied mathematics*, vol. 74, no. 1, pp. 167–190, 2014.

[225] M. Cucuringu, P. Rombach, S. H. Lee, and M. A. Porter, "Detection of core–periphery structure in networks using spectral methods and geodesic paths," *European Journal of Applied Mathematics*, vol. 27, no. 6, pp. 846–887, 2016.

[226] F. Zhang, Y. Zhang, L. Qin, W. Zhang, and X. Lin, "Finding critical users for social network engagement: The collapsed k-core problem," in *AAAI*, 2017.

[227] K. Shin, T. Eliassi-Rad, and C. Faloutsos, "Corescope: Graph mining using k-core analysis—patterns, anomalies and algorithms," in *ICDM*. IEEE, 2016, pp. 469–478.

[228] F. D. Malliaros, A. N. Papadopoulos, and M. Vazirgiannis, "Core decomposition in graphs: Concepts, algorithms and applications." in *EDBT*, 2016, pp. 720–721.

[229] J. Cheng, Y. Ke, S. Chu, and M. T. Özsu, "Efficient core decomposition in massive networks," in *ICDM*. IEEE, 2011, pp. 51–62.

[230] F. Della Rossa, F. Dercole, and C. Piccardi, "Profiling core-periphery network structure by random walkers," *Scientific reports*, vol. 3, no. 1, pp. 1–8, 2013.

[231] F. Zhang, Y. Zhang, L. Qin, W. Zhang, and X. Lin, "When engagement meets similarity: efficient (k, r)-core computation on social networks," *arXiv preprint arXiv:1611.03254*, 2016.

[232] S. P. Borgatti and M. G. Everett, "Models of core/periphery structures," *Social networks*, vol. 21, no. 4, pp. 375–395, 2000.

[233] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks," in *SIGCOMM*, 2007, pp. 29–42.

[234] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *JSTAT*, vol. 2008, no. 10, p. P10008, 2008.

[235] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[236] L. Lettry, M. Perdoch, K. Vanhoey, and L. Van Gool, "Repeated pattern detection using cnn activations," in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2017, pp. 47–55.

[237] K. Zhou and F. Long, "Sentiment analysis of text based on cnn and bi-directional lstm model," in *2018 24th International Conference on Automation and Computing (ICAC)*. IEEE, 2018, pp. 1–5.

[238] X. Huang, D. Chen, D. Wang, and T. Ren, "Identifying influencers in social networks," *Entropy*, vol. 22, no. 4, p. 450, 2020.

[239] U. Brandes, "A faster algorithm for betweenness centrality," *Journal of mathematical sociology*, vol. 25, no. 2, pp. 163–177, 2001.

[240] Y. Zhang and Y. Zhang, "Top-k influential nodes in social networks: A game perspective," in *SIGIR*, 2017, pp. 1029–1032.

[241] M. Alsaleh, A. Alarifi, A. M. Al-Salman, M. Alfayez, and A. Almuhaysin, "Tsd: Detecting sybil accounts in twitter," in *2014 13th International Conference on Machine Learning and Applications*. IEEE, 2014, pp. 463–469.

[242] S. Liu, B. Hooi, and C. Faloutsos, "Holoscope: Topology-and-spike aware fraud detection," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 2017, pp. 1539–1548.

[243] S. Yardi, D. Romero, G. Schoenebeck *et al.*, "Detecting spam in a twitter network," *First Monday*, vol. 15, no. 1, 2010.

[244] F. Pierri and S. Ceri, "False news on social media: A data-driven survey," *arXiv preprint arXiv:1902.07539*, 2019.

[245] F. C. D. da Silva, R. Vieira, and A. C. Garcia, "Can machines learn to detect fake news? a survey focused on social media." in *HICSS*, 2019, pp. 1–8.

[246] C.-S. Atodiresei, A. Tănăselea, and A. Iftene, "Identifying fake news and fake users on twitter," *Procedia Computer Science*, vol. 126, pp. 451–461, 2018.

[247] M. Fire, D. Kagan, A. Elyashar, and Y. Elovici, "Friend or foe? fake profile identification in online social networks," *Social Network Analysis and Mining*, vol. 4, no. 1, p. 194, 2014.

[248] N. Chavoshi, H. Hamooni, and A. Mueen, "Identifying correlated bots in twitter," in *International Conference on Social Informatics*. Springer, 2016, pp. 14–21.

[249] K. Shin, B. Hooi, and C. Faloutsos, "M-zoom: Fast dense-block detection in tensors with quality guarantees," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2016, pp. 264–280.

[250] K. Shin, T. Eliassi-Rad, and C. Faloutsos, "Patterns and anomalies in k-cores of real-world graphs with applications," *Knowledge and Information Systems*, vol. 54, no. 3, pp. 677–710, 2018.

[251] S. P. Borgatti, "Identifying sets of key players in a social network," *Computational & Mathematical Organization Theory*, vol. 12, no. 1, pp. 21–34, 2006.

[252] D. Eswaran, C. Faloutsos, S. Guha, and N. Mishra, "Spotlight: Detecting anomalies in streaming graphs," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 1378–1386.

[253] S. Jendoubi, A. Martin, L. Liétard, H. B. Hadji, and B. B. Yaghlane, "Two evidential data based models for influence maximization in twitter," *Knowledge-Based Systems*, vol. 121, pp. 58–70, 2017.

[254] Y. Mei, W. Zhao, and J. Yang, "Influence maximization on twitter: A mechanism for effective marketing campaign," in *2017 IEEE International Conference on Communications (ICC)*. IEEE, 2017, pp. 1–6.

[255] G. Rattanaritnont, M. Toyoda, and M. Kitsuregawa, "A study on characteristics of topicspecific information cascade in twitter," in *Forum on Data Engineering (DE2011)*, 2011, pp. 65–70.

[256] M. A. N. Hakim and M. L. Khodra, "Predicting information cascade on twitter using support vector regression," in *2014 International Conference on Data and Software Engineering (ICODSE)*. IEEE, 2014, pp. 1–6.

[257] S. Ye and S. F. Wu, "Measuring message propagation and social influence on twitter. com," in *International conference on social informatics*. Springer, 2010, pp. 216–231.

[258] P. E. Brown and J. Feng, "Measuring user influence on twitter using modified k-shell decomposition," in *Fifth international AAAI conference on weblogs and social media*, 2011.

[259] H. R. Zeidanloo and A. B. A. Manaf, "Botnet detection by monitoring similar communication patterns," *arXiv preprint arXiv:1004.1232*, 2010.

[260] F. Atefeh and W. Khreich, "A survey of techniques for event detection in twitter," *Computational Intelligence*, vol. 31, no. 1, pp. 132–164, 2015.

[261] Y. Zhang, X. Ruan, H. Wang, H. Wang, and S. He, "Twitter trends manipulation: a first look inside the security of twitter trending," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 1, pp. 144–156, 2016.