Reverse Engineering the Behaviour of Twitter Bots

Thesis submitted for the degree of
Doctor of Philosophy
at the University of Leicester

by

Bello Shehu Bello
School of Informatics
University of Leicester

April 2020

Reverse Engineering the Behaviour of Twitter Bots

Bello Shehu Bello

Abstract

Online social media platforms such as Twitter offer flexible and effective means of communication on a large scale. The use of Twitter by political figures and government officials increases the wider acceptance of the medium. However, the use of automated accounts known as bots for election campaigns raises concerns that the medium is now being used to disseminate political propaganda aimed at manipulating public opinion. Recent research has shown significant success in the detection of bots. While there are approaches to distinguish automated from regular user accounts, information about their masters, targets, strategies, biases and antagonisms remains harder to obtain. This is a challenging task but can lead to a better understanding of their use in political campaigns.

In this thesis, we proposed an approach to reverse engineering the behaviour of Twitter bots to create a visual model explaining their actions. We use machine learning to infer a set of understandable rules governing the behaviour of a bot and a visual notation to make such rules accessible to a non-technical audience. We propose the notion of differential sentiment to provide means of understanding their behaviour with respect to the topics on their network in relation to both their sources of information (friends) and their target audience (followers). Respectively, this provides insights into their bias and antagonism with the target audience. We evaluated our approach using prototype bots we created and selected real Twitter bots. The results show that the approach is successful in describing the behaviour of the bots correctly and that the approach can help to understand their role and impact.

This thesis contributes to knowledge with regard to understanding the behaviour and strategies of Twitter bots. As shown by two case studies, the approach can help to monitor the use of bots to manipulate public opinion and to create transparency in public debate.

Acknowledgements

In the name of Allah, the most beneficent, the most merciful.
All praise is to Almighty Allah who give me the patience and the strength to complete this thesis. Peace blessings be upon the last prophet and the most elite, Prophet MUHAMMAD (Sallallahu alaihi wa sallama) who has commanded every Muslim to seek knowledge from cradle to grave.

I owe the greatest gratitude to my supervisor, Professor Reiko Heckel who patiently supported and advised me through all stages of this thesis. He is always patient with me even at midnight looking at my work and offering comments. This thesis would not have been completed without his valuable feedback and great mentorship. He has contributed greatly to my academic career. I have learned so many things from him both academically and personally. I wish him a wonderful and successful life ahead with his family.

I would like to thank my previous PhD co-supervisor, Dr Leandro Minku, PhD tutor Dr Fer-Jan de Vries, and PhD Director/GTA Manager Prof. Thomas Erlebach for their guidance and support during this journey. Thanks to all those PhD students who have shared time with me and made the environment friendly for me.

Thanks to the University of Leicester for the Graduate Teaching Assistantship given to me to support my PhD. My sincere appreciation goes to the Petroleum Technology Development Fund (PTDF) for the partial scholarship given to me to cover my tuition fees. I would like to thank Hajiya Rabi Waziri of PTDF for her kindness and support to me. I would like to thank Prof. Yahuza Bello and Prof. Hajiya Aisha Halliru for their support. Many thanks to Dr. Abdullahi Baffa Bichi for his support and encouragement to begin this journey.

Completing this work would have been difficult without prayers and encouragement from my family. I would like to express my heartfelt gratitude to my parents. Special to thanks my elder brother, Yaya Mustapha who is always supportive of me in pursuing my dreams. Many thanks to all my brothers and sisters. I would like to offer my profound gratitude to my beloved wife Zainab for her patience, prayers and support during this process. I also thank my son Ahmad for making me smile. Thanks to all my friends and colleagues who have supported me with their prayers.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Social media have become a popular platform where people publicly demonstrate their opinions, sharing their views on products, services, sports, politics and other interesting events. The massive amount of data generated daily on social media such as Twitter has opened a new research direction called online social network analysis. Online social network analysis is an approach for analysis of social relationships and the diffusion of information in online social networks [9]. An online social network is a social structure made up of actors called nodes connected by one or more specific relationships such as friendships, likes, common interests or beliefs [9]. Undoubtedly, in the last few years, there has been an increase in research, applications targeted at public opinions with regards to politics, natural disasters, disease epidemics and stock markets [41].

The rise of social bots have set a new challenge in social network analysis and research that deals with social media. Social bots are particularly common on Twitter; they tweet, retweet and actively participate in political discussions [64]. A review of Twitter accounts by Twitter Inc. shows that about 5% of all the Twitter user accounts are social bots [34]. Social bots are automated user accounts that interact with other users, send and reply to messages, perform various actions and decisions based on pre-defined rules [41] . Bots can perform useful tasks such as delivering news, directing users to useful links, and updating news feeds. However, bots are widely used to spread spam, mislead and manip-

ulate the content of social media [41, 64]. Bots are now widely used by political actors to attack their opponents and promote political interests. In the 2016 UK referendum on EU membership, Twitter bots were used by both parties, supporters of Brexit and of StrongerIn, to promote their interests [64]. The two most active bots are @iVoteLeave and @iVoteStay. These bots actively participated in the campaign by re-tweeting tweets containing hashtags of StrongerIn or Brexit [64].

The role of social bots in manipulating public opinion became apparent in 2010 during the US midterm elections, where they were used to spread thousands of tweets intended to damage political opponents [41, 91]. Similarly, bots were used in the 2016 US presidential elections [6, 8], and campaigns in France [40], Australia [48], Norway [69] and Venezuela [43].

The activity of bots on social media can give false impressions that a piece of information is liked or widely accepted by many people [41]. This false information can damage the reputation of a product, company or political actor. It can also alter the result of social media analysis adopted for various purposes such as rating of products and services and expert findings on scientific measurements [44].

1.1 Motivation

Twitter is a popular social media platform used by many people, news providers and organisations to share information [34]. As of 2019, Twitter had an average of 145 million daily active users [34]. This makes it a rich source of information and network to connect with people. The rise of social bots jeopardizes the content of Twitter and other social networks. Despite the efforts of Twitter Inc. to identify and block Twitter bots, research shows that such bots are being created in large numbers and subsequently live long without being shut down [44]. One important question is how to detect Twitter bots? As Twitter bots increasingly become more sophisticated and harder to identify, the detection of bots has became an interesting research question which has led to the development of

different approaches and tools for the detection of Twitter bots [20,21,30,32,49,51,65,92]. This helps to detect and block a large number of bots. Research shows that not all Twitter bots are harmful, however. Some Twitter bots are used to provide user-specific news and to recommend products and services [41]. The research highlighted the need for more detail about Twitter bots in order to identify harmful bots [41, 64]. This requires information which is not provided by the detection systems [41]. While Twitter bots are becoming great tools for political campaigns and calling for volunteers, the need for more detail about the Twitter bots has become an important concern which raises several questions [41]. How do bots take their actions? What is their target? What are their strategies?

We approach these questions by reverse-engineering the behaviour of Twitter bots. Reverse engineering is a process of analyzing an existing system to extract knowledge or design information [23].

This research aims to reverse engineer the behaviour of Twitter bots to create a model explaining their reactions to events such as incoming tweets. This will provide information which, if analysed, will help in identifying their masters, target and strategies on social media.

1.2 Problem Statement

Recent research has shown significant success in the detection of bots. While there are tools to distinguish bots from regular user accounts, information about their masters, strategies, biases and antagonisms remains harder to obtain [5, 21, 30, 77]. To uncover such details, e.g., to understand their masters, the strategies they employ and the role they play in online campaigns, this thesis intends to answer the following questions:

About the behaviour of bots: How can we observe the behaviour of a bot? How can we describe the behaviour of a bot (when and how a bot takes actions) by a set of understandable rules? How can we extract such information automatically, from the observations of a bot? What are the limitations to this approach?

About the contents of tweets produced: Social Twitter bots are created to promote specific agendas and opinions. Understanding these in detail is an interesting; but challenging task. How can we identify topics of interest to a bot and detect the opinions promoted by a bot? How can we express the bias and antagonism of a bot?

1.3 Overview of the Approach

This thesis took a reverse engineering approach to address the need for information about the behaviour of bots. Reverse engineering is a challenging task as it requires extraction, analysis and mapping different pieces of low-level information to build a correct high-level representation of the system in question [1,23,82]. In our case, the task is even more challenging as we would not have the opportunity to run the software (bot). We can only observe traces of its behaviour. The two most essential steps in reverse engineering are i) collection and analysis of traces, and ii) model construction. Model construction which is also referred to as building views is the task of creating a high-level representation of the system by analysing the information derived from the traces [17,23]. This is difficult and challenging because it requires enough understanding from reliable sources to construct the high-level representation [10, 17, 38]. To address this challenge, in Section 3.1 we create a meta-model of the Twitter API to serve as the basis for the model construction since the API is the primary source of data and actions of the bots.

The approach proposed in this thesis is divided into two parts. i) Toward reverse engineering the behaviour of a bot to understand its behaviour, and ii) toward understanding the opinion, bias and antagonism of a bot. The first part is divided into three main steps (A, B & C), as illustrated in Figure 1.1. The output of each step is the input of the next step.

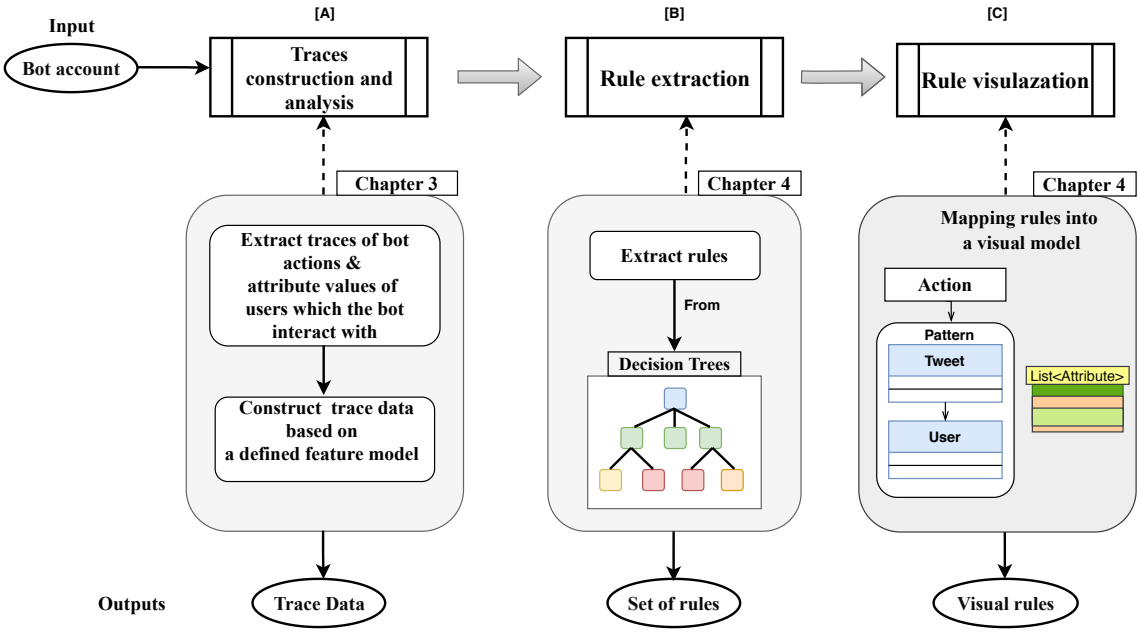


Figure 1.1: Overview of our approach toward reverse engineering the behaviour of a bot

The first step (A) focusses on extracting traces of a bot's actions via the Twitter API, with the analysis and extraction of features to construct a trace data based on Twitter and feature model presented in Chapter 3. The second step, illustrated in step (B) of Figure 1.1, focusses on the extraction of rules governing bot's behaviour. This involves building a machine learning model using a decision tree and extraction of the rules as detailed in Chapter 4 (cf. Section 4.2). The third step, shown in (C) part of Figure 1.1, focusses on providing a visual representation which simplifies and generalises the visualisation of the rules. Next, we give further details about the approach with reference to the research questions outlined in Section 1.2.

How can we observe the behaviour of a bot?

Reverse engineering begins with an analysis to derive information, mainly through observation, by running the system with different inputs, executing and testing various components of the system [10, 23, 82]. Since we do not have the opportunity to run the bot, we observed its behaviour by i) mining of data from its Twitter account to construct traces of

the bot's actions and attribute values, and ii) through searching and analysis of patterns to learn how certain attribute values trigger an action. In Section 3.3.1, we describe the set of attributes that form the fundamental constructs of our traces, and in Section 4.1, we provide details about how we observe the behaviour of a bot.

How can we describe the behaviour of a bot (when and how a bot takes actions) by a set of understandable rules?

Our aim is to provide a convenient and comprehensive method to describe the behaviour of a bot. Production rules are widely used and well-understood structures for representing knowledge; they facilitate symbolic reasoning and inference [70, 88]. In Section 4.1, the author describes how to represent the behaviour of a bot as a set of production rules. The author defines a general rule for describing the behaviour of a bot. A Twitter bot rule is composed of a pattern *P* and the invocation of an operation, i.e., IF *Pattern* THEN *Action*. The pattern of a rule can be based on attributes of tweets and/or users, but can also be more complicated, e.g., considering the follower relation. A typical example of a Twitter bot's rule is retweet any tweet (t2) which contains a hashtag, e.g., $\#x$ or $\#y$ or $\#z$. More formally, retweet : IF h in tweet (t2) where $h \in H$, $H = \{x, y, z\}$. In this case, $retweet$ is the bot's action, and tweet with hashtags $\#x$, $\#y$, $\#z$ is the pattern which forms part of the rule.

How can we extract such information automatically, from observations of a bot?

This thesis uses a machine learning approach to address this question. Machine Learning is an approach to knowledge mining, and discovering patterns and correlations in a pre-existing dataset [70]. Conventional machine learning algorithms such as Naive Bayes can classify entities and provide statistical distributions among variables, but they do not provide a conceptual description of their relationship [70]. Decision tree is a popular method by which to capture the relationship between input and output variables but has limited representational power [70]. Using the concept of graph transformation, in Section 4.2

we present a rule-based approach for representing the same information. We employ graph transformation concepts to generate rules from a decision tree for two purposes. First, general rules can cover unseen examples and combine leaf nodes of the same type; second, they simplify and enhance the readability of the model.

To tackle the issue of visualisation, in Section 4.3 we present an approach which simplifies visualisation of the rule. For each action, our visualisation produces a graph representation of a set of rules $\{r_1, r_2, \ldots, r_n\}$ formed from the set of attributes used by a bot to take actions. Figure 4.5b shows a simple example of our visual representation.

The second part of our approach is depicted in Figure 1.2. It focusses on understanding the opinion, bias and antagonism of a bot to address the following questions.

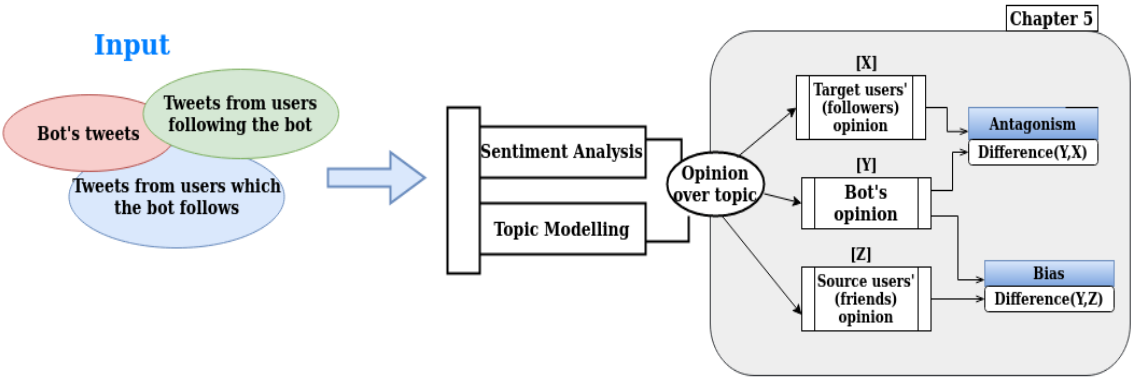


Figure 1.2: Toward understanding the opinion, bias and antagonism of a bot

How can we identify topics of interest to a bot and detect the opinions promoted by the bot?

Users on social media communicate about different topics using short texts. This leads to a collection of short texts with a mixture of different topics and opinions. Finding users' opinions on a topic involves a two-stage process, first the detection of topics followed by identification of sentiment toward each topic.

In Section 5.2, the author presents a model for identifying topics of interest to a bot and opinion of the bot on those topics. The model is based on a popular topic-modelling algorithm, Latent Dirichlet Allocation (LDA). Rather than learning on a document of

single words as is done in traditional LDA and its other extensions, we convert the tweets into a corpus of aggregated bi-grams. The use of bigrams instead of unigrams allows us to keep the correlation between the words and obtain good topics. We consider opinion to be a sentiment intensity (a float value in a range -1.0 to +1.0., e.g., -0.1, 0.0, 0.1, 0.2,.. 1.0) rather-than sentiment polarity (negative, neutral, positive) to adequately capture the differences in opinions even if they are of the same polarity.

Social bots are mainly orchestrated to promote a specific agenda in a conceivable manner, and propagate information in support of or against certain topics. In many cases, bots are not straight in their opinion on a topic. For example, if a bot is positive on a topic, it might share some weak negative opinion to attract users with an opposing opinion in an attempt to gain influence. To determine the opinion of a bot on a topic, we propose the concept of mean absolute sentiment (MAS) in Section 5.2. In Section 7.2, we show how the proposed concept can be use to identify strongly opinionated accounts in a large group of accounts with different opinions, and in Section 7.3 we use the concept to understand the level of confrontation between opponents and supporters of political candidates.

How can we express the bias and antagonism of a bot?

In this section, the thesis aims to address two important factors associated with the opinion of a bot, namely bias and antagonism. A bot can be programmed to propagate information on a topic or from specific users without considering the sentiment but in many cases, social bots are designed to be selective (bias). Antagonism can be viewed as a sign of social influence [97]. The concept of antagonism and social influence has been widely discussed in the area of social psychology, but is new in the area of online social networks (OSNs) [31,71,93,97]. Researchers of this aspect of online social networks focus mainly on identifying influencers and measuring or quantifying user influence [3, 7, 72, 76, 79, 114]. This thesis considers the bias and antagonism in a differential way, focusing on the sentiment intensity of the shared content between the users rather than users' social attributes such as the number of followers, friends, mentions and retweets, as considered

by [79] and [7]. The author argues that while users' social attributes could indicate potential influencers, those attributes do not signify influence as the change in opinion of both parties has to be considered [71]. Additionally, since other external factors may affect users' decisions, evidence of influence from social media could be unjustifiable. Hence we can only observe and measure antagonism.

If an opinion of a bot and that of users on its network is available to us, how can we express the bias and antagonism of a bot? In Section 5.3, the author introduce the notion of differential sentiment (DS) for the study of the bias and antagonism of a bot. As illustrated in Figure 1.2, the author defines the bot's *bias* as the difference in sentiment between the tweets produced by the bot and the tweets on this topic issued by the bot's friends, i.e., the users which the bot is following. The difference in sentiment between the tweets produced by the bot and those of its followers defines the bot's *antagonism*. In Section 5.4, the author introduces differential sentiment overtime (DSO) as a means to track and detect changes in the opinion of a bot. In DSO, the author considers the evolution of sentiment over time to detect changes in bias and antagonism.

What are the limitations to this approach?

One of the biggest challenges to reverse engineering approaches is an incomplete representation of the system [10, 17, 38]. This is because the information is derived from a collection of traces that solely represent those behaviours that are actually executed or observed. Thus, the model produced by our approach is incomplete, capturing only those behaviours that are actually executed by the bots.

The automation of the approach relies on machine learning, which itself depends on the availability of patterns to discover behaviour. Machine learning will generally discover a pattern if the expression of that pattern reaches a certain statistical threshold. This also affects the completeness of the approach. In Chapter 6, the author evaluates the approach and provides a detailed discussion about correctness, completeness and performance.

1.4 Contributions and Related Publications

This thesis as a whole delivers an approach to understand the behaviour of Twitter bots through reverse engineering. The approach allows the analysis of Twitter bots to understand their interests, strategies, biases and antagonisms. It allows the effective extraction and representation of rules controlling the behaviour of Twitter bots. We evaluate the performance, correctness and utilization of the approach. Specifically, the thesis consists of the following contributions:

- An approach to reverse-engineer the behaviour of a Twitter bot.
  - We present a general rule for describing the behaviour of a bot.
  - We present an automated approach to extract rules controlling the behaviour of a Twitter bot.
  - We present an approach for the representation and visualisation of the rules.

- An approach to understand opinion, bias and antagonism of a bot
  - We proposed the notion of differential sentiment for the study of bias and antagonism of a bot.
  - We introduce differential sentiment over time (DSO) to track changes in bias and antagonism of a bot. This also helps understand the response of a bot to an external event.

- Tool support and Applications:
  - As proof of concept, we provide an implementation of our proposals. i) Tool for extraction and visualisation of rules controlling the behaviour of a Twitter bot. ii) Tool for both absolute and differential sentiment analysis. These are available on GitHub [1]

[1] https://github.com/bellobichi2/botscope

– Using the proposed approach, we present a study on the behaviour of bots in post Brexit politics. We also present a study on social media campaign strategies during the 2019 Nigerian Election.

Previous publications

Parts of this thesis have been peer-reviewed and published in papers detailed below. The paper titled "Reverse Engineering the Behaviour of Twitter Bot" introduces the technical contributions of the thesis.

- Bello Shehu Bello, Reiko Heckel, and Leandro L. Minku. Reverse Engineering the Behaviour of Twitter Bots. In *The Fifth International Conference on Social Networks Analysis, Management and Security, SNAMS 2018, Valencia, Spain, October 15-18, 2018, pages 27–34, 2018.*
  URL: `https://doi.org/10.1109/SNAMS.2018.8554675`

- Bello Shehu Bello, Reiko Heckel. Analyzing the Behaviour of Twitter Bots in Post Brexit Politics. In *The Six International Conference on Social Networks Analysis, Management and Security, SNAMS 2019, Granada, Spain, October 22-25, 2019.*
  URL: `https://ieeexplore.ieee.org/document/8931874`

- Bello Shehu Bello, Reiko Heckel and Isa Inuwa-Dutse. Social Media Campaign Strategies: Analysis of the Nigerian Election. In *The Six International Conference on Social Networks Analysis, Management and Security, SNAMS 2019, Granada, Spain, October 22-25, 2019.*
  URL:`https://ieeexplore.ieee.org/document/8931869`

1.5 Thesis Structure

This thesis is organised into eight chapters, as highlighted in Figure 1.3.

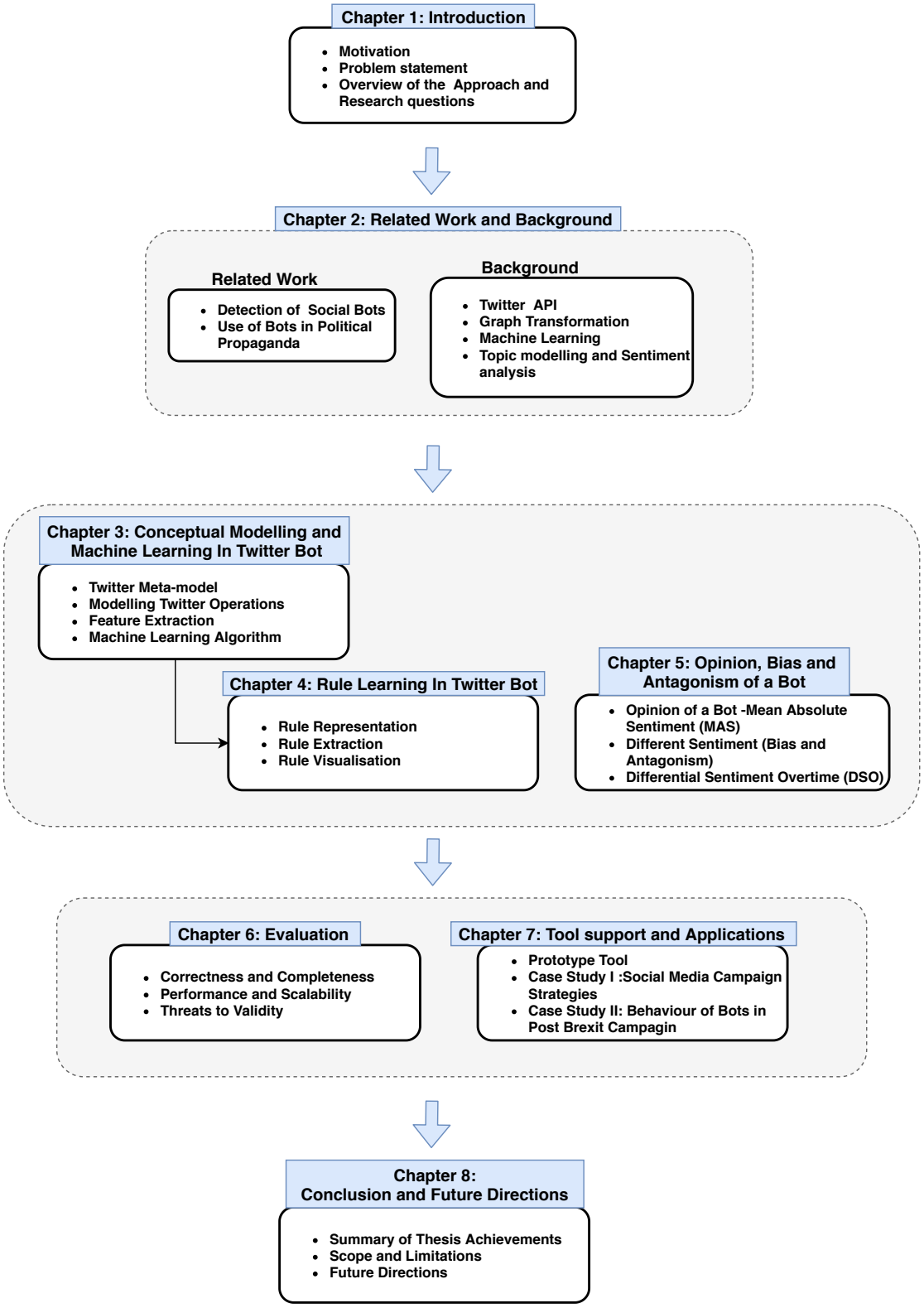


Figure 1.3: Thesis Structure

- **Chapter 2** discusses related work in the literature and how this thesis extends the current state-of-the-art. It also provides background information useful for under-

stand the subsequent chapters.

- In **Chapter 3** we create a meta-model of Twitter API and model the Twitter operations. This serves as our source for model construction and forms the foundation of our rules for describing the behaviour of Twitter bots. We describe how we extract features for learning the behaviour of bots, and preprocess and select relevant features. Finally, we describe how we select a suitable algorithm for learning and describing the behaviour of bots.

- **Chapter 4** presents an approach to learn the rules controlling the behaviour of a bot. It contains a description of how we observe the behaviour of a bot and represent it using understandable rules (cf. Section 4.1). The chapter describes how to automatically extract rules from observations in Section 4.2 and presents an approach for their visualisation in Section 4.1.

- **Chapter 5** presents concepts and approaches for understanding the opinion, bias and antagonism of a bot. We describe how to identify topics of interest to a bot and detect opinions promoted by a bot in Section 5.2. In Section 5.3, we introduce differential sentiment analysis (DS) as a means of understanding the bias and antagonism of a bot.

- **Chapter 6** contains a detailed evaluation of the proposed approach. This includes a discussion on the correctness, completeness and performance of the approach.

- **Chapter 7** presents a prototype tool created as a proof of concept. This assists us in the evaluation of the proposed approaches and in the demonstration of their applications.The chapter also includes a case study of the behaviour of bots after the Brexit referendum and a case study of the 2019 Nigerian election from a social media perspective. The two case studies are presented to demonstrate the application and usability of the proposed approaches.

Finally, **Chapter 8** provides concluding remarks and discusses possible future work.

Chapter 2

Related Work and Background

This chapter presents techniques addressing related problems and discusses how this thesis extends the current state of the art. It also provides the background information required to understand subsequent chapters.

Section 2.1.1 presents approaches for the detection of social bots and describes how this thesis extends the state of the art. In Section 2.1.2, we present related studies on the use of social bots in online campaigns and describe how the approach proposed in this thesis can be used to enhance the studies. Section 2.2 describes the nature of the tweet data used in this study and the APIs used to access the data. This thesis uses the concepts of graph transformation (GT) to describe the behaviour of bots. Section 2.3 explains the GT concepts used in the thesis. Machine learning (ML) forms the foundation of the proposed rule learning approach. Section 2.4 describes how ML is used in this thesis. Section 2.4.2 describes the performance measures used to quantitatively evaluate ML results in various sections of the thesis. Section 2.5 introduces topic modelling techniques and sentiment analysis methods useful to understand the opinion model proposed in Section 5.2.

2.1 Related Work

In this section, we discuss techniques addressing related problems and how this thesis extends the current state of the art. The thesis delivers a novel approach for reverse engineering the behaviour of social bots to extend work on the detection of social bots.

2.1.1 Detection of Social Bots

In recent years, social bots have become more sophisticated, challenging existing detection strategies. Bot detection approaches proposed in the literature can be classified into three groups: graph-based, crowd sourcing-based and feature-based [41]. The graph-based approach [18, 85, 109] focuses on the structure of the social graph to identify bots. This is based on the assumption that bots exhibit a small number of links to legitimated accounts as legitimated users refuse to interact with unknown users. This was proven to be unrealistic in experiments [36, 100] because connection and interacting with unknown users is among the main features of platforms such as Twitter and Tumblr. The crowd-sourcing approach [107] relies on the use of human intelligence to identify bots. This makes the approach practically expensive especially to platforms with many pre-existing users like Facebook and Twitter. Feature-based detection systems [20, 21, 30, 32, 51, 92] focus on the use of machine learning methods to learn behavioural signatures of human-like and bot-like behaviour. This is one of the most effective and successful approaches to date. This is closer to our work in terms of the use of accounts' features and machine learning methods. However, we extend the idea by considering additional features and analysing the detailed behaviour of the accounts. Next, we discuss recent feature-based bot detection approaches and finally describe how we extend the start of the art.

BotOrNot, now Botometer[1], is one of the feature-based detection systems made public available to check for the presence of social bots on Twitter [30]. It classifies Twitter accounts as bots by selecting and analysing features that differentiate between human and

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

automated users. The features are grouped into six main classes: retweet, user mention and hashtag co-occurrences are network features used to compute statistical information. The number of followers and posts are classified as friends' features. These provide statistics relative to the account. Tweet rate and inter-tweet time distribution are important factors to identify automated accounts. Botometer uses these as temporal features to capture the timing pattern of content generation and consumption. Other features used by Botometer are user, content and sentiment [30]. Botometer relies on pre-existing large train examples of bots and humans accounts. Thus sometimes classifies the accounts of famous people and accounts with low-levels of activity as bots.

In 2015, DARPA organised the Twitter Bot Detection Challenge to develop techniques for early detection of bots [101]. The features used by most participants to identify bots were similar to those used in Botometer. They concluded that as Twitter bots are becoming more sophisticated there is a strong need for efficient ways to categorize them. Detection of sophisticated accounts that exhibit a mixture of humans and bots (sometimes referred to as a cyborg) features is challenging, and realistically impossible for feature-based systems [113]. In [25], the authors proposed a method for classification of Twitter accounts into human, bot or cyborg by analysing their tweeting behaviour, content and account properties. Dickerson et al. [32] claim that traditional network features-based methods of detecting bots [24, 25, 29, 111] are less effective, or ineffective, if the topic of the discussion is very specific. They introduce a set of sentiment-based variables that improve accuracy for the detection of bot accounts. They found that sentiment flip-flop, positive sentiment strength, negative sentiment strength and fraction of tweets with sentiment are among the most distinguishing features. Gilani et al. [51] attempted to improve the accuracy of classifying Twitter accounts by categorizing them into four different levels of popularity based on the number of followers and by considering additional features with the later including favourites-to-tweets ratio, lists per user, favourites per tweet, retweets per tweet, user replies, source identity and content size. The majority of the methods follow the same approach, using off-the-shelf supervised machine learning algorithms trained with examples of both humans and bots behaviour based on features extracted

from tweeting behaviour, content and account profile. One major shortcoming of these approaches is the inability to detect a group of bot accounts (botnets). They also require a large amount of labelled data.

Nikan et al. [20, 22] proposed an unsupervised approach for the detection of correlated user accounts. This is based on the assumption that highly correlated user accounts are more likely to be bots because humans cannot be highly synchronous for long duration. The approach works based on accounts' activity correlation without the need for labelled data. In [21], Nikan et al. developed a system called Debot based on their activity correlation approach to detect and archive a group of bots in near real-time by collecting and analysing tweets from trending topics. As the approach relies on activity timestamps, bots that are not timely correlated in their activity can escape detection.

With every new technology being defeated by bot-masters, the authors in [44] analysed bot infiltration strategies in Twitter. Although Twitter is trying to block bots, their work shows that there are many ways in which Twitter bots can be created in large numbers and survive for a long time without being detected. They created 120 social bot accounts with different characteristics and strategies to investigate the extent to which the bots could infiltrate the Twitter network. Their results showed that even social bots with simple automated mechanisms can infiltrate Twitter successfully. They claimed that if social bots are created in large numbers, they can endanger our politics by disseminating false information, which in turn can have a significant impact on public opinion.

This thesis extends the art of bot detection by reverse engineering their behaviour to provide information which can significantly increase our understanding of their strategies and behaviour. In terms of the approach, we analyse traces of bot activity, and apart from analysing the bots' features, we analyse the features of the tweets and users they interact with. Rather than providing only statistical values for their patterns, our approach provides rules together with information which can improve our understanding of their strategies, who they target, how they take action and what topics they talk about. Apart from traces of the bots' actions, this approach does not require a large amount of pre-

existing labelled data.

2.1.2 Use of Social Bots in Political Propaganda

Social media platforms such as Twitter are now widely used by political figures and government officials to make announcements and reach out to their supporters [45]. This increases the wider acceptance of the medium. Twitter and other social media platforms play an important role in steering public participation in social policy and public activities [6, 12, 37, 80]. The use of social media in social and political campaigns has been examined by various studies, including on the online mobilisation of protest [54, 55], in the Occupy Wall Street movement [26], and in other political campaigns [104, 106]. The extent to which social media is used in this space and the acceptance of the public to actively participate in discussions on social media mean that this channel is used to spread propaganda to manipulate public opinion [63,64]. Use of social media in this manner was reported in [28,35,91,98] and found to be effective in influencing public opinion [2,7,27].

However, the rise of bots changes how politicians use social media. Political actors employ social bots to engage in political conversations [43, 64]. This offers them better opportunity to promote their agendas. The role of social bots in manipulating public opinion became apparent in 2010 during the US midterm elections, where they were used to spread thousands of tweets intended to damage political opponents [91]. Similarly, bots were used in the 2016 US presidential elections [6,8,77], the UK Brexit referendum [64], and campaigns in France [40], Australia [14, 48], Norway [37, 69] and Venezuela [43]. Political actors, organizations and other entities with adequate resources can deploy thousands of bot accounts to support or attack certain opinion [6].

In [64], the authors studied the use of political bots during the UK referendum on EU membership. They used popular campaign hashtags of both supporters of leaving the EU, those against it, and some neutral hashtags to collect tweets that yielded 313,832 distinct user accounts. Our approach, as proposed in Section 5.2 and demonstrated in

Section 7.2.3, to finding strongly opinionated accounts can indeed be used to find such accounts. One of the findings is that bots played a strategic role in the referendum's conversations. The bots mainly use a family of hashtags associated with arguments to propagate the messages. They also mentioned that @ivoteLeave and @ivotestay follow similar algorithms, but evidence of this has not been demonstrated. The required human effort in the attempt to discover and study the bots' algorithms can be reduced by using our automated rule learning approach, and the rule visualisation can be used as evidence to support the study.

The authors in [6] studied the effects of Russian trolls and bots during the 2016 U.S. election. Their first research question was about the users' political ideology, and the second question was about the role of social bots. In the result section (RQ2: The role of social bots) they mentioned that 75 bots were liberal and 2,018 are conservative. This was supported by a probability distribution of the bots scores and t-tests. The text analysis was based on statistical counts of the keywords rather than opinionated. As they indent to study the role of social bots, analysis and discussion can be enhanced and extended by using our proposed differential sentiment to study the relationship between the bots and the Russian trolls. It can also be used to study the correlation between liberal users and liberal bots, conservative users and conservative bots or in a crosswise (liberals vs. conservatives). On the study of Russian trolls, they mentioned that most of the retweets that were generated were only for three troll accounts @TEN_GOP, @Pamela_Moore13, The-FoundingSon. Our automated rule learning can be used to provide more details about the behaviour of these accounts.

In another study of the 2016 U.S. election [8], the authors investigate how the presence of social bots affected the political discussion. They collected tweets with about 2.8 million distinct users and classified them into Trump supporters and Clinton supporters based on the campaign hashtags in their tweets. They used Botometer to classify the supporters' accounts into bot and human accounts. To understand how the bots and the humans discussed the two candidates, they analysed the distribution of their sentiment in a cumulatively manner. This analysis could be enhanced by using our proposed concept

of differential sentiment over time (see Section 5.4) to get more insight into how the political discussion changes over time and by studying the correlation between the bots and the humans. Toward the end of their discussion, they presented an example of bots supporting Trump and bots supporting Clinton with their tweets as a result of their manual analysis to get more insight. Rather than presenting bots and example of tweets produced, our automated rule learning approach can be used to present strategies used by the bots. This will enhance the result and discussion of the study. Next is a discussion on concepts and background information required to understand the further chapters.

2.2 Tweet Data and Application Program Interface (API)

Twitter is the main source of data for analysis and experiments presented in this thesis. The tweet data which is encoded using JavaScript Object Notation (JSON) is collected via the Twitter REST and Streaming APIs. The REST API allows developers to get historical data, and the streaming API provides live tweet data. Twitter provides free and enterprise access to these APIs. The free access has some limitations. The REST API allows access to at most 3,200 of the latest tweets from each user account or a sample of tweets published in the last seven days in a case of search-based query. In a case of real-time tweets, the standard version of the streaming API allows developers to track at most 400 keywords, 500 users, and 25 locations. There are other limitations regarding the number of calls to these APIs. Detailed information about these is available on the Twitter developer's website[2]. The main object of a tweet data is the tweet object. This is associated with a user object which describes the author of the tweet and place object to indicate the location of the tweet if the user enables geo-tagged or location post. The structure of these objects with some of their attributes is shown below. In the example the tweet object (*tweet*) contains attribute *created_at* which indicates the time at which the tweet is posted. The attribute *text* contain the textual content published. The user object (*user*) contains attributes (e.g., id, name, location) which record details about the tweet's

[2]https://developer.twitter.com/en/docs/basics/rate-limiting.html

author.


```
{
  "tweet": {
    "created_at": "Thu Apr 06 15:24:15 +0000 2017",
    "id_str": "850006245121695744",
    "text": "Today we are sharing our vision for the-
    future of the Twitter API",
    "user": {
      "id": 2244994945,
      "name": "Twitter Dev",
      "screen_name": "TwitterDev",
      "location": "Internet",
      "url": "https:\/\/dev.twitter.com\/",
      "description": "Your official source for Twitter Platform
       news, updates & events."
    },
    "place": {
    },
    "entities": {
      "hashtags": [
      ],
      "urls": [
        {
          "url": "https:\/\/t.co\/XweGngmxlP",
                    }
        }
      ],
      "user_mentions": [
      ]
```

```
    },
     "extended_entities": {
     }
  }
}
```

The entities object stores arrays of hashtags, user mentions, URLs and cashtags. The extended entities object stores media (photo, video, or animated GIF) if the tweet contains a media file. Figure 2.1 shows a map of tweet data and details about the Twitter API can be found in Chapter 3 where we model the Twitter API and its basic operations to provide a view of the API from our research perspective.

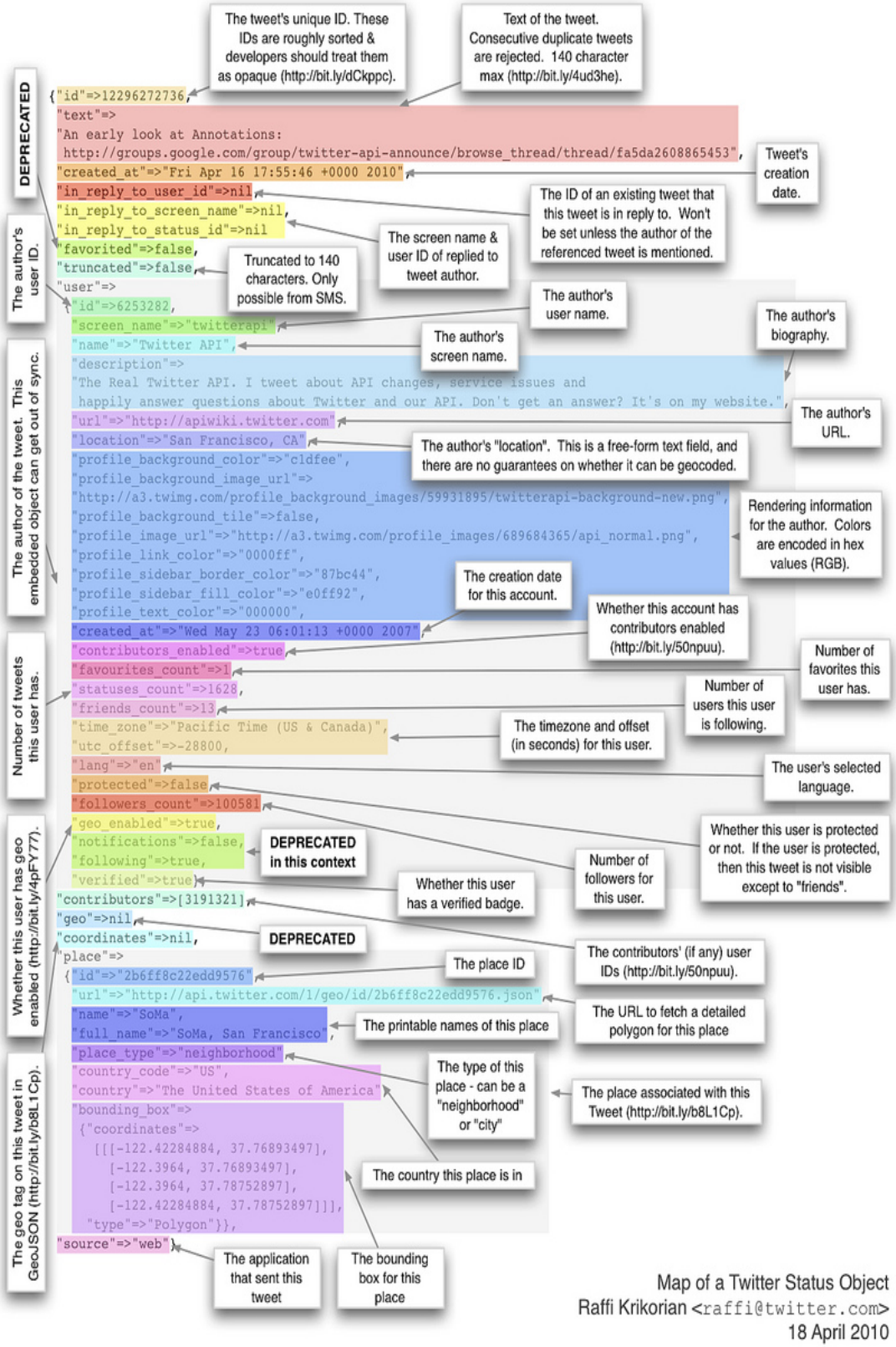


Figure 2.1: Map of a Twitter Status Object Describing the Structure of Tweet Data [74]

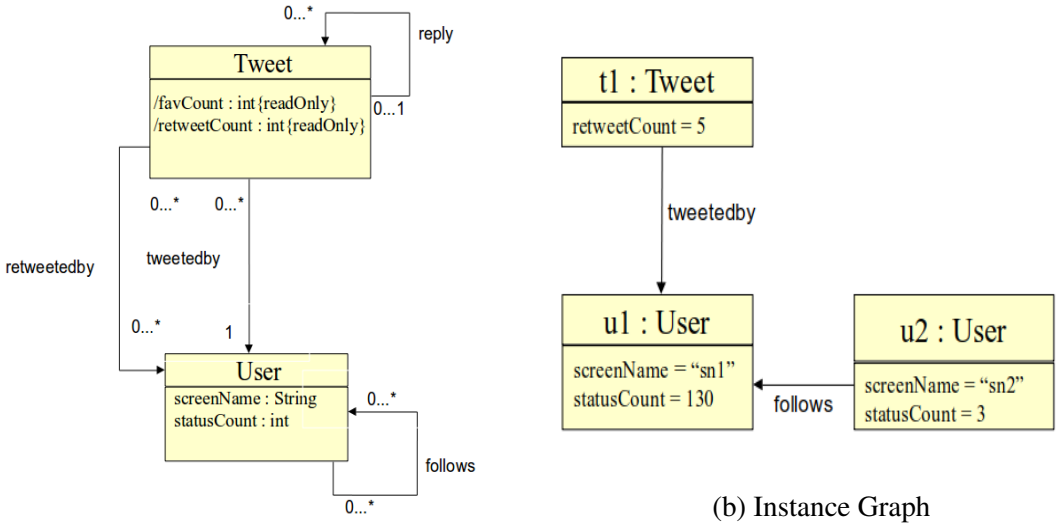
2.3 Graph Transformation

In computer science, particularly software engineering (SE), graphs and diagrams play an essential role in both representation and visualisation of complex systems, making them more readable and interpretable. In SE, we need two forms of system representation, structural and behavioural. The Unified Modeling Language (UML) provides a standard way to represent the design of a system. The graph transformation systems (GTS) are used to describe dynamic behaviour of a system. Figure 2.2 is used to explain the concepts of graph transformation used in the thesis. In GTS, graphs are used model the system states, and graph transformation rules are used to specify their evolution.

2.3.1 Type and Instance Graphs

A graph is made up of two sets called Vertices V and Edges E such that each element of the Edge set has a source and a target vertex. With the aid of UML notations of class and object diagrams, we usually depict graphs by drawing the vertices as boxes and the edges as lines between the boxes.

Type graphs are used to represent concepts and their relationship at a type level. Instance graphs capture the relations at the instance level (i.e, a in specific example). Figure 2.2a and 2.2b show an example of a *type graph (TG)* and an *instance graph (IG)*, respectively. Here, t1:Tweet represents an instance (an object) of type Tweet (a class). Both instance and type graphs may contain attributes to store values of pre-defined data types. At the type level we use a declaration of $a : T$, where a is an attribute name and T is the data type. At the instance level we have $a = v$ where v is a value assigned to the attribute a. For example, at the type level, we have declaration retweetCount: int and retweetCount=5 at the instance level. [60] explain these in detail, including conditions under which instance graphs are valid to a corresponding type graph.



(a) Type Graph

(b) Instance Graph

Figure 2.2: Example of type and instance graph from our model

2.3.2 Graph transformation rules

We use the concepts of graph transformation rules to describe the various operations available on Twitter. In this section, we briefly describe the concepts to provide a background for Chapter 3. Graph transformation rules are expressed as pre and post conditions together with an operation. More formally, $p : L \rightarrow R$, where L and R are instance graphs of the same type and attributes. The left-hand side L represents the precondition of the rule, and the right-hand side R represents the postcondition while p is the rule's name. Figure 2.3 below is an example of a GT rule.

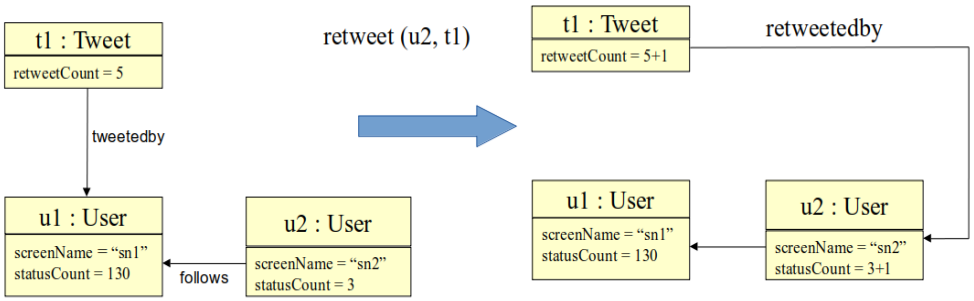


Figure 2.3: Retweet rule

In the above example, the rule is *retweet (u2, t1)*. Application of the rule will create a connection between *t1* and *u2*. The retweet count of *t1* and status count of *u2* will increase by one (retweetCount = 5+1 and statusCount= 3+1).

2.4 Machine Learning

2.4.1 Machine Learning Methods

Machine learning is a platform for generating knowledge from a set of training data. While traditional programming approach involves using input data and a program to produce output, machine learning methods use data and output to create a program (algorithm). Machine learning techniques are broadly divided into three categories: supervised, unsupervised and semi-supervised [66]. In this thesis, we use supervised machine learning method to learn patterns of rules governing the behaviour of Twitter bots. In supervised learning, the input data is presented as a set of features $\{x_1, x_2, \ldots, x_n\}$ extracted from a dataset with corresponding output labels $\{y_1, y_2, \ldots, y_n\}$, where y_i is generated by an unknown function $y = f(x)$. The goal is to find a hypothesis $h(x)$ such that $h(x)$ is an approximate function of $f(x)$. In Section 4.2, we use a machine learning method (decision tree) to learn and extract rules governing the behaviour of a bot. The following five machine learning algorithms were evaluated during the ML algorithm selection.

Instance-Based Learner (K-Nearest Neighbours): IBK is a non-parametric method for classification and regression. Here we are interested in the classification part. The application of IBK in pattern recognition brings it into our evaluation methods. It is a simple algorithm which uses instances of previous data with known output value to predict the output value of a new instance using a similarity measure (distance function). For K=1, IBK uses the distance function to find an instance closest to the new instance with the unknown output value. The output of the closest instance will be regarded as an output of the new instance. In our experiments, we use an implementation of IBK that uses the

Euclidean distance function shown in Equation 2.1.

$$D(x,y) = \sqrt{\sum_{j=1}^{n} f(x_j, y_j)} \tag{2.1}$$

$D(x,y)$ gives a distance between two instances x and y, where x_j and y_j refer to the j^{th} feature value of instance x and y respectively. For numerical attributes $f(x_j, y_j) = (x_j - y_j)^2$; for categorical values $f(x_j, y_j) = 0$ if the feature values of x_j and y_j are the same else $f(x_j, y_j) = 1$ if they are different.

Naïve Bayes: This is a simple but powerful algorithm for prediction. It provides a way of calculating the posterior probability $P(c|x)$ from a prior probability of the class $P(c)$, prior probability of the predictor $P(x)$ and likelihood $P(x|c)$ which is the probability of the predictor given class as shown in equation 2.2 [95].

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)} \tag{2.2}$$

$$P(c|X) = P(x_1|c) \times P(x_2|c) \times P(x_3|c) \times ..... \times P(x_n|c) \times P(c) \tag{2.3}$$

In our case, c is an action, e.g., retweet/notRetweet, and x is a feature with a given set of instances $\{x_1, x_2, ..., x_n\}$. For example screen-name = {uniofLeicester, uolInformatics, heForHer}. $P(c/x)$ is a probability of an action given an instance x. $P(c)$ is the prior probability of the action (c). The likehood $P(x|c)$ is the probability of an instance x given an action c. $P(x)$ is the prior probability of the instance (x).

Support Vector Machines (SVM) is a robust machine learning algorithm for classification and regression. It performs classification by finding a hyperplane that maximizes a margin between two classes. In our experiments, we have tested two implementations of SVM, LibSVM and LibLINEAR SVM. While LIBSVM implements the sequential minimal optimization (SMO) algorithm for kernelized support vector machines, LIBLINEAR is a library for large linear classification. It implements linear SVM and logistics regression

models trained using a coordinate descent algorithm.

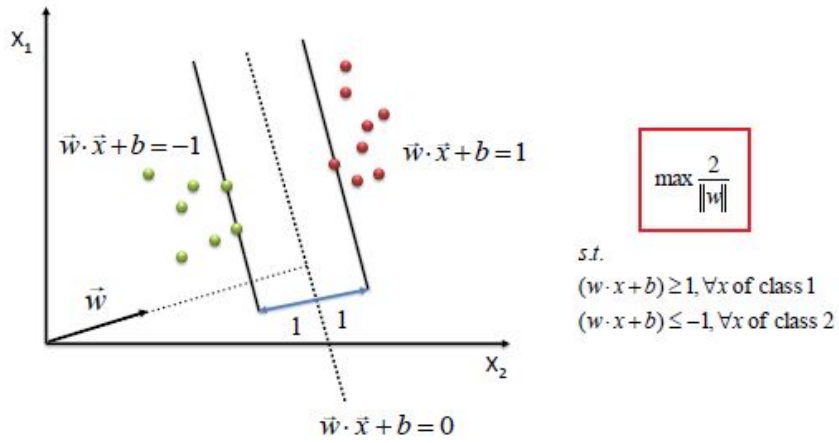


Figure 2.4: SVM trained with samples from two classes [95]

Figure 2.4 above shows an SVM trained with samples from two classes. Elements of class 1 are marked in red, and that of the second class (-1) are marked in green. The algorithm defines an optimal hyperplane by maximizing the width of the margin (w) [95]. The optimal hyperplane is the hyperplane that lies between two parallel hyperplanes which separate the two classes. The equation $\vec{w} \cdot \vec{x} - b = -1$ describes anything on or before the boundary of the hyperplane with class label -1, and $\vec{w} \cdot \vec{x} - b = 1$ describes anything on or after the boundary of the hyperplane with class label 1. The distance between these two hyperplanes is $\frac{2}{\|w\|}$. To maximize the distance between the planes, $\|w\|$ have to be minimized. The distance is computed as a distance from a point to a plane equation, i.e., $w \cdot x - b \geq 1$, if $y = 1$, and $w \cdot x - b \leq -1$, if $y = -1$. The constraint is added to ensure that each data point lie on the correct side of the margin.

Decision Trees is one of the practical methods of inductive learning. It can produce a good interpretable model. The basic idea behind the decision tree algorithm is to select the best feature(s) that split the remaining instances and make that feature a decision node. The process is repeated recursively for each child until all the instances associated with the node have the same target value, or there are no more features or instances associated with the node. In decision trees, features are represented as nodes. The leaf node indicates the output value and link is a feature value leading to another decision node or leaf node.

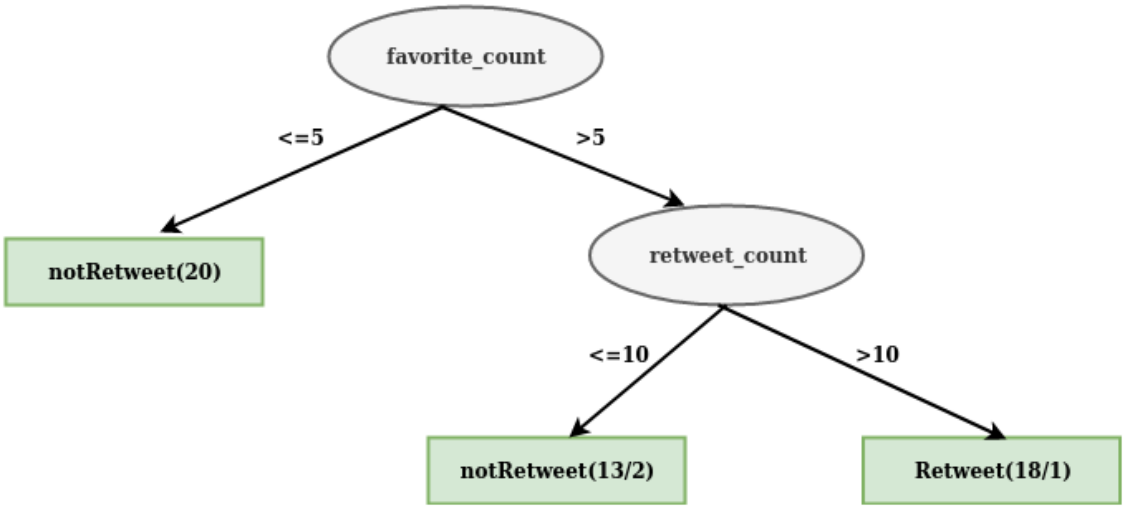


Figure 2.5: An example of a decision tree in case of Twitter

The decision tree shown in Figure 2.5 separates the two classes (retweet and notRetweet) based on favourite count and then based on retweet count. Tweets with favourite count greater-than five and retweet count greater-than ten are retweet while those with favourite counts less-than or equal to five are not retweet. Tweets with retweet count less-than or equal to ten are also not retweet . The first number shown in bracket at the leaf node indicates the total number of instances (weight) reaching the leaf node. The second number is the number (weight) of instances that are misclassified.

Random Forest: The basic idea behind a random forest is to combine many decision trees into a single model. This is achieved by creating multiple trees at the training time and outputting a class which is a model of all the classes of the individual trees. The assumption is that the prediction made based on an average of the individual trees will be less prone to over-fitting than that of the individual trees [61].

2.4.2 Performance Measures

In this thesis, we use the standard measures of machine learning and information retrieval to evaluate, quantitatively, results of classifiers used in various sections. The measures are accuracy, precision, recall and area under the receiver operating characteristic curve

(roc_auc). In many cases, roc_auc score has been used as a final measure of performance because of its insensitivity to class imbalance. It is an indicator of how successful a classifier is in identifying true positive examples and avoiding false positives. Consider the confusion matrix in Table 2.1. With regards to the classification of a bot's action as a

Table 2.1: Confusion Matrix

	Predicted: retweet	**Predicted: notRetweet**
Actual: retweet	true positive (TP)	false negative (FN)
Actual: notRetweet	false positive (FP)	true negative (TN)

retweet or notRetweet the norms in the confusion matrix can be defined as follows:

- True positive (TP) = number of retweets that are correctly classified as retweets.
- True negative (TN) = number of notRetweets that are classified as notRetweets.
- False positive (FP) = number of notRetweets that are classified as retweets.
- False negative (FN) = number of retweets that are classified as notRetweets.

Using the above confusion matrix, the performance measures can be defined as follows:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Accuracy is the ratio of correct predictions made to the total predictions made. It is presented as a percentage by multiplying the result by 100. Precision measures the exactness of the model while recall quantifies completeness of the model. These are formally defined as

$$precision = \frac{TP}{TP + FP}, recall = \frac{TP}{TP + FN}$$

The roc_auc score is the area under the curve of the true positive rate (recall) plotted against the false positive rate (FPR). The FPR is defined as $FPR = FP/(FP + TN)$.

The roc_auc score is one of the most popular measures used to evaluate the robustness of a classifier. Using retweet as an example, this is the probability that the classifier will judge a randomly chosen retweet action as a retweet rather than a randomly chosen notRetweet as retweet. This is important in our rule learning, as we want to avoid false rules while aim to recover as many correct rules as possible.

2.5 Topic Modeling and Sentiment Analysis

2.5.1 Topic Modeling

The foundation of our topic model is Latent Dirichlet Allocation (LDA), LDA is an algorithm for extracting hidden topics in text documents [11]. Despite there are many algorithms for topic modelling, research [62,67] shows that Latent Dirichlet Allocation(LDA), Bi-term Topic Model(BTM), Word2vec Gaussian Mixture Model(W2V-GMM) and Word network topic model are the prominent ones. Authors [67, 108] evaluate these algorithms on tweet texts and show that standard LDA performs very well on tweets. It is even better than some of the newly proposed models such as W2V-GMM because the model become progressively less coherent when more topic words are taken into account in the computation of the coherence score. LDA is an unsupervised machine learning technique use to identify latent topics from a collection of text documents. In standard LDA each document is treated as a vector of words and each topic is represented as a probability distribution over the number of words grouped in the topic. One major concern of using LDA on a short text is data sparsity as LDA captures the word occurrence patterns by modelling the word generation from the document level. This leads to excessive dependence on the local observation for the inference of word topic assignments z which effect learning of the topics ϕ when documents are short. To address this, we use a bi-term topic model [110]. We learn topics over aggregated bi-grams (pairs of consecutive words), i.e., instead to represent the distribution of the topics over documents, we represent the distribution over the aggregated bi-grams.

Given tweets of a bot as a collection of bi-grams, the generative process of the corpus in BTM can be described as follows :

1. For each topic z
   (a) draw a topic-specific word distribution $\phi_z \backsim Dir(\beta)$ for the whole collection.
2. Draw a topic distribution $\theta \backsim Dir(\alpha)$, where α and β are Dirichlet parameters.
3. For each bi-gram b in the bi-grams set B
   (a) draw a topic assignment $z \backsim$ Multi(θ)
   (b) draw two words: $w_i, w_j \backsim$ Multi(ϕ_z)

Following the above BTM steps, the joint probability of a bi-gram $b = (w_i, w_j)$ can be expresss as:

$$P(b) = \sum_z P(z)P(w_i|z)P(w_j|z) = \sum_z \theta_z \phi_{i|z}\phi_{j|z}$$

Thus the likelihood of the whole corpus is :

$$P(B) = \prod_{(i,j)} \sum_z \theta_z \phi_{i|z}\phi_{j|z}\theta$$

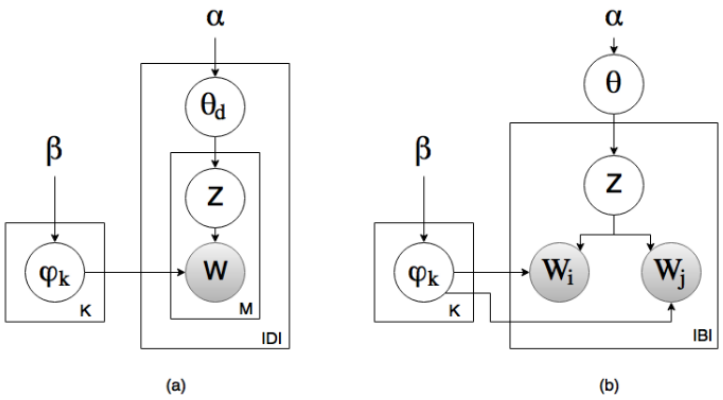


Figure 2.6: Graphical representation of learning over (a) Documents (|D|), and (b) aggregated bigrams (|B|) (Originally taken from [67])

Figure 2.6 shows the improvement of BTM over the LDA. Instead to represent the distribution of the topics over documents, we use distribution over aggregated bi-grams. This solves the LDA's data sparsity problem. The use of bi-grams instead of uni-grams allows us to keep the correlation between words and capture multiple topics in a document.

2.5.2 Sentiment Analysis

Sentiments analysis is an important area in text analysis. There are two approaches, using machine learning and using a lexicon. The machine learning approach requires training data which is manually annotated with different classes of sentiments. The training data is used to create a model for predicting the sentiment of unseen data. This approach is time-consuming and computationally expensive [53]. This leads to the use of lexicons for sentiment analysis. Several dictionaries and approaches have been introduced including Linguistic Inquiry and Word Count (LIWC) [87], Harvard General Inquirer (GI) [99], Affective Norms for English Words (ANEW) [13, 84], SentiWordNet [4], and SenticNet [15]. The major shortcoming of these approaches is inadequate attention to sentiment relevant features of social text and disregard of intensity differences among sentiment-bearing words [53].

SentiStrength [102, 103] is one of the sentiment analysis algorithms designed for social media data. It can detect slang, booster words and emoticons, but it is insensitive to contractions, conventional use of punctuation (e.g., "Good!!!") or word-shape (e.g., using ALL CAPS for words) to increase sentiment intensity. To address these shortcomings [53] proposed a rule-based sentiment model for social media text. The model uses a valence-aware dictionary for sentiment reasoning (VADER). VADER contains a validated standard list of lexical, grammatical and syntactical features of social media text combined with rules used to express and emphasize sentiment intensity in social media. The model is not only classifying a sentiment as positive, negative or neutral; it also provides a score (-1.0 to +1.0) to indicate the intensity level of the sentiment. In addition to the consideration of emoticons like "-:)", "LOL"," meh, it takes other features such as

capitalization and punctuation into account. Text like "I love it !!!", 'WONDERFUL gifts' will have higher intensity then "I love it. " and wonderful gifts. It also consider a shift in sentiment values due to the presence of words like "but". VADER has been added to the Natural Language Toolkit (NLTK). We use VADER to compute sentiment of individual tweets in this thesis.

Chapter 3

Conceptual Modelling and Machine Learning in Twitter

This chapter presents the first step in describing the behaviour of Twitter Bots. The chapter is divided into two parts: in the first, we create a meta-model of the Twitter API (c.f. Section 3.1) and then use the GT concepts described in Chapter 2 to represent the various operations available on Twitter, forming the foundation of our rule representation in Chapter 4. In the second part (Section 3.3), we describe how we extract features for learning the behaviour of bots, pre-process and select relevant features. In Section 3.3.4, we describe the role of machine learning algorithms in learning patterns and select a suitable algorithm for learning and describing the behaviour of Twitter bots. Finally, we summarise the discussion of this chapter in Section 3.4.

3.1 Twitter Meta-Model

Our goal is to describe the behaviour of Twitter bots through reverse engineering. Reverse engineering requires a reliable model to build a correct representation of the subject system [17, 23]. Twitter bots use the Twitter API to search for tweets and to take actions. Hence, we analysed the API to understand the various actions and data available

to Twitter bots and designed an appropriate meta-model of the API to serves as our primary source for the model construction. The meta-model forms the foundation of our rule representation.

Twitter Meta-Model

The meta-model is shown in Figure 3.1. The Twitter API allows programmatic access to public tweets in real-time. Tweets are the building blocks of all things on Twitter[1].

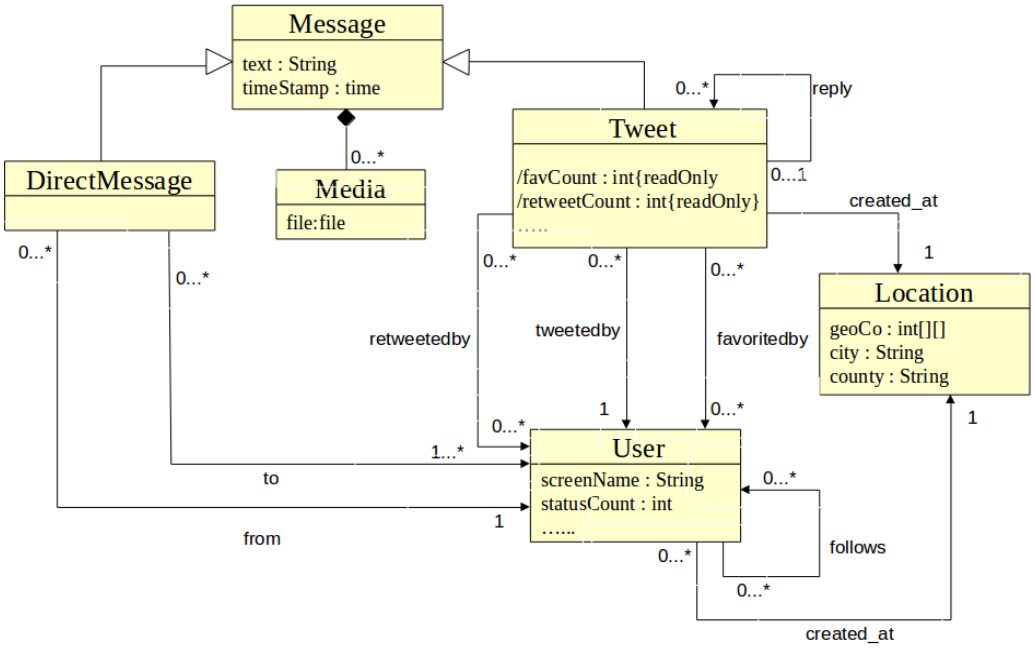


Figure 3.1: Twitter Meta-model

A Tweet is a text of not more than 280 characters representing a user's opinion or interest. It is also referred to as a status update[2]. In our model, we consider Tweet as the main class. It has several associated attributes. We classify all the attributes with numeric values, e.g., retweetCount, favouriteCount, etc., as derived attributes because their values can be computed from the number of links between the tweet and the user who made the

[1]https://dev.twitter.com/overview/api/tweets
[2]https://dev.twitter.com/overview/api/tweets

action. We consider text and timestamp to be the main attributes common to both tweet and direct message, which is why we assigned them to a class message and defined it as a superclass of Tweet and Direct_Message. URL and hashtags are text attributes. A message may contain zero or more media files. A media file can be a photo or video.

A User is anyone or anything controlling the Twitter account[3]. It can perform a number of actions such as tweet, retweet and follows.

A Location is a specific named place[4] with corresponding geo-coordinates indicating the location at which the tweet or the Twitter account was created. This is recorded only if the user permits it during the configuration of the account.

3.2 Modelling the Twitter Operations

A Twitter bot can tweet or retweet a tweet created by another user. It can reply or like tweets created by other users. It can also follow other users. To capture these operations correctly as part of the bots' behaviour, we use the concept of graph transformation (described in Section 2.3) to represent such operations. These operations are the building blocks of our rule for describing the behaviour of Twitter bots. The operations are tweet, follows, retweet, direct message and favorite operation.

A tweet operation is specified as $tweet\ (u : user,\ txt : string)$, where u is a user who created the tweet of type string. An example of how it works and its representation using graph transformation is shown in Figure 3.2.

[3]https://dev.twitter.com/overview/api/users
[4]https://dev.twitter.com/overview/api/places

Operation 1: $tweet$ ($u1$,"$text1$")

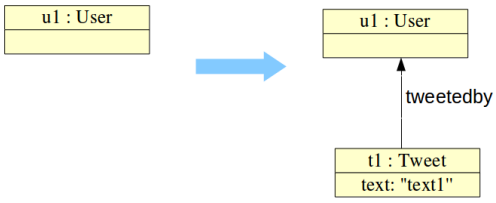


Figure 3.2: Tweet operation

The operation $tweet$ ($u1$, "$text1$") shown in Figure 3.2 will create a tweet with text "$text1$" and link it to a user $u1$. The user $u1$ is the owner of the tweet as shown in the diagram above.

A follower operation is specified as $follows$ ($u1$: $user, u2$: $user$) where a user $u1$ follows a user $u2$. An example of the operation is shown in Figure 3.3.

Operation 2: `follows (u2, u1)`

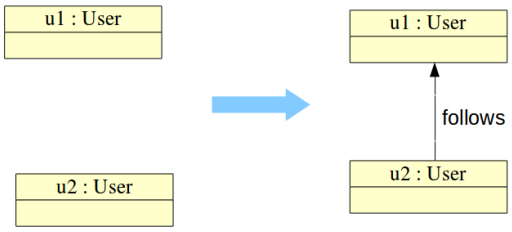


Figure 3.3: Follows operation

The operation $follows$ ($u2, u1$) shown in Figure 3.3 creates a follower relation in which a user $u2$ is a *follower* of a user $u1$. This means that the user $u2$ will be receiving all tweets made by the user $u1$. In Twitter $u2$ is regarded as a *follower* of $u1$ and $u1$ is called *friend* of $u2$. The relationship can be bidirectional in which $u1$ is a follower of $u2$ and $u2$ is a follower of $u1$. In this case, both $u1$ and $u2$ will be receiving tweets made by each other.

A reply operation is specified as $reply$ (u : $user, t$: $tweet, txt$: $string$), where a user

u replies to a tweet t with a text of type string. An example of the operation is shown in Figure 3.4.

Operation 3: $reply\ (u1, t1, "text1")$

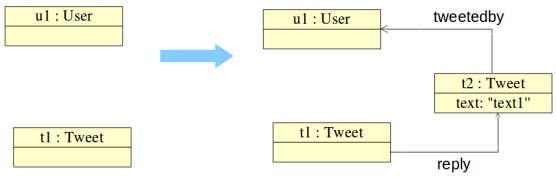


Figure 3.4: Reply Operation

The operation $reply\ (u1, t1, "text1")$ creates tweet $t2$ with a text $text1$ as a reply of the tweet $t1$.

A direct message operation is an operation which is used to send a private message from one user to another. This is specified as $dMessage\ (u2 : user, u1 : user, txt : string)$, where a user $u2$ send a direct message of type string. Direct message operation is depicted in Figure 3.5

Operation 4: $dMessage(u2, u1, "text1")$

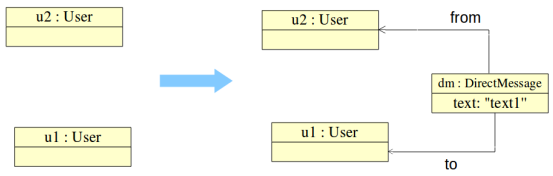


Figure 3.5: Direct message operation

A retweet is an operation by which a user can tweet an existing Tweet. This is specified as $retweet\ (u : user, t : tweet)$. An example of this operation is shown in Figure 3.6.

Operation 5: $retweet\ (u2, t1)$

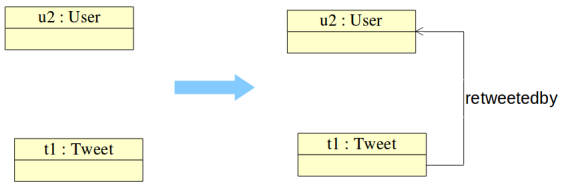


Figure 3.6: Retweet operation

A favorite operation is specified as $favorite\ (u : user, t : tweet)$. An example where user u favorites a tweet t is shown in Figure 3.7.

Operation 6: $favorite(u2, t1)$

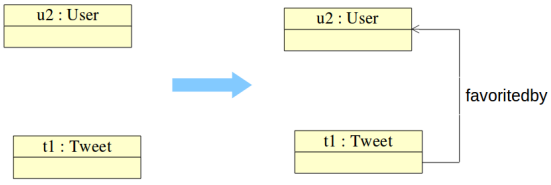


Figure 3.7: Favorite operation

The operation $favorite\ (u2, t1)$ creates a favorite relation between the user $u2$ and the Tweet $t1$.

3.3 Machine Learning In Twitter Bots

3.3.1 Feature Extraction

Twitter is a large repository of data from which various features can be extracted. Twitter bots use the Twitter API to search for tweets and take actions. Identification of such features is important in terms of understating their behaviour. Using the Twitter API, we extract an initial set of features to learn the behaviour of Twitter bots, which are detailed in the following subsections. The features include user features, tweets content features, network features, temporal and sentiment features. Many researchers have used very

similar features for the detection of bots [20, 51, 52, 101]. One key difference between the work in this thesis and research on detection of bots, researchers learn patterns from the statistical count of the features while we learn patterns from the actual values of the features and their relationship. This is because we are looking for more detailed behaviour of the bots. For example, in the hashtag feature, other researchers count the number of hashtags per tweet or group of tweets to distinguish bots from human accounts while we record the set of hashtags reacted to or used by a bot to distinguish between two campaigns promoted or opposed by a bot. While many researchers focus on features of the subject account (bot/human), we analyse both features of the subject account and those of the tweets and the users the bot interacts with. In the following sections, we discuss these features in detail.

NOTE: For each feature in the following tables, we actually have two associations, one for the bot account and one for the users which the bot interact with. Instead of the prefix *user* and *tweet* that we use to keep the report more compact, all bot's account features are prefixed with $U1$ and $U2$ for users which the bot interact with. The bot's tweet features are prefixed with $T1$ and $T2$ for tweets of the other users.

User Features

These are features which describe the Twitter user's account profile. The user account profile features play an essential role in identifying communication patterns in social media. This class of features have been used to detect bot accounts in [30], [51], and [101]. We extract both, bot's profile features and those of users it interacts with. We extract user's account features to understand the type of users which a bot interacts with or is interested in. The profile features enable us to keep track of any change in the account, such as change in account's description to change the campaign. Table 3.1 provides list and description of the user features.

Table 3.1: User features

Feature	**Description**
user_ID	Unique number which identifies the account
user_screenName	Twitter name/handle of the account
user_created_at	Date and time which the account was created
user_statusCount	Number of tweets(including retweets) produced by the user
user_Acctlocation	User's account location
user_time_zone	Time zone selected by the user
user_verified	True if Twitter verifies the account
user_default_profile	When true, indicates the user has not changed it's profile background
user_lang	Preferred user's account language
user_profile_image_url	URL link to the user's profile image
user_description	User-provided description of the account

Content Features

Analysis of tweets' content produced or reacted to by a bot will provide important details about its campaign. For each tweet produced or reacted to, we extract the following content features (cf. Table 3.2).

Table 3.2: Content features

Feature	**Description**
tweet_ID	Unique number which identifies the tweet
tweet_created_at	Date and time which the tweet was created
tweet_hashtags	List of hashtags in a tweet
tweet_userMentions	List of users mentioned in a tweet
tweet_urls	Expanded, top-level domain names of tweet's URLs
tweet_mediaType	Type of media (photo,video,GIF) in the tweet
tweet_mediaID	Unique identifier of the tweet's media file
tweet_mediaUrl	URL link of the tweet's media file
tweet_text	Full tweet's text content (See Section 3.3.2)
tweet_Source	Device/Utility used to post the tweet
tweet_Coordinates	Tweet's Geotag location (latitudes and longitudes)

Network features

These are features which signify the relationship between a tweet or a user with other users. In our Twitter meta-model 3.1, we called these derived features. Twitter bots use these features to identify relevant tweets or users. For example, Twitter bots can be programmed to follow users with a specific number of followers or friends. We extract features shown in Table 3.3 to capture such selective behaviour.

Table 3.3: Network features

Feature	Description
user_followers_count	Current number of users following the user account
user_friends_count	Current number of users followed by the user account
user_favorite_count	Total number of tweets this user has liked in the account lifetime
tweet_retweet_count	Total number of users that retweet this tweet
tweet_favorite_count	Total number of users that liked this tweet

Sentiment features

Recent research has demonstrated the importance of sentiment analysis in revealing the nature of conversation on social media [32, 53, 101]. Sentiment describes emotion and opinion polarity conveyed in a piece of text. Authors [30, 32] use statistical changes in sentiments to antithesize bots from human user accounts. The authors mainly concentrate on the sentiment of the tweets produced by the subject account to derived their sentiment features. In this thesis, we extract three initial sentiment features, sentiment of tweets produced by a bot account (T1sentiment), sentiment of tweets reacted to by the bot (T2sentiment) and sentiments of tweets produced by (followers and friends of the bot). In Section 4.3, we use the sentiment features to add more information to the rules describing the behaviour of a bot. In Section 5.2, we use sentiment to detect the opinion of a bot and study the relationship between bots and online users in Section 5.3.

Time Features

The Timestamp is a vital signature which can be used to track activities over time. Previous research [21, 30, 51, 101] have used the rate of content production and time interval to distinguish bots from human-user accounts. We extract the following basic time features to study how the behaviour of the bots changes over time and understand their evolution during online campaigns (See Section 5.4). The features are timestamp of

each tweet produced by a bot (T1created_at), original timestamp of each tweet reacted to by the bot (T2created_at), timestamp of tweets produced by followers, friends of the bot (T2created_at_f1, T2created_at_f2), bot's account creation time (U1created_at) and Users' account creation time (U2created_at).

Bot's Action

One of our goals is to learn the patterns of the bot's actions., i.e., when and how the bot takes actions. For each tweet data that we extract with the above features, we capture and record the bot's action associated with the data. The action can be a tweet, retweet, reply, favorite or follow. Including this association into the data is another unique aspect of this thesis in the study of Twitter bots.

3.3.2 Data pre-processing

Tweet text data tend to be noisy, requiring additional processing steps before being used for machine learning. Once we collect tweets, we clean the tweets by removing user handles, stop words and reserved words (e.g., RT, FAV). In the case of URLs, users mainly use shorten URLs. We expand the short URLs to full URLs and extract the top-level domain name of the URLs. This enables our rule learning model to provide details which could help to understand bots sharing news from fake websites. We vectorize the remaining tweets' text into bi-grams and select the top 20 features. The use of bi-grams instead of uni-grams allows us to keep the correlation between words. We finally integrate the vector of the text features into our set of features for the rule learning.

3.3.3 Feature Selection

Feature selection plays an essential role in machine learning. The purpose of it can be either to reduce computational cost, improve performance or quality of a model by se-

lection of the relevant features [19, 57–59]. The later is our goal. Feature selection is different from dimensional reduction; both seek to reduce the number of features. Dimension reduction methods reduce the features by creating a new combination of the existing features, whereas feature selection methods select and remove features from the original set of the features without changing them.

After pre-processing and coding the data ready for machine learning, we usually end up with a large set of features of various types and priorities. This brings the need for additional steps to make the learning at least faster and cost-effective. Feature selection has been an important research field in machine learning; plenty of methods were proposed due to the availability of different data with many variables [19]. While most of the feature selection methods can reduce computation time, improving prediction accuracy or quality of the model became a critical factor to prioritize one method over the other. In the area of Twitter bots, Gilani *et al.,* [51] used ablation test (an act of adding and removing features) to detect the most significant features that yield the best accuracy to classify a Twitter account as bot or human. Dickerson *et al.,* [32] used PCA (Principal component analysis) to reduce the dimension of features for the detection of bot accounts. On the aspect of the feature selection, there is one key difference between the related works and the work in this thesis. While the majority of the works aimed at selecting relevant features that provide higher accuracy for the detection of bots, we aimed at the selection of features that correctly described the behaviour of the bots.

Researchers [73] study feature selection with regards to the selection of relevant features or optimal features; they show that optimal features are not always relevant features. We use the following steps to select features relevant to the behaviour of a bot.

Steps for feature selection

1. Remove features with more than 50% missing values. This is to remove features that do not have enough values to describe the behaviour.

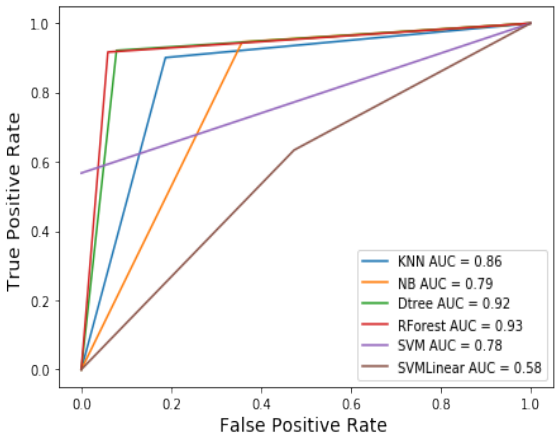
2. Remove features with low variance among the classes. Features with low variance

are features that have the same values in all the samples. At this step, features that are either one or zero in more-than 80% the samples will be removed.

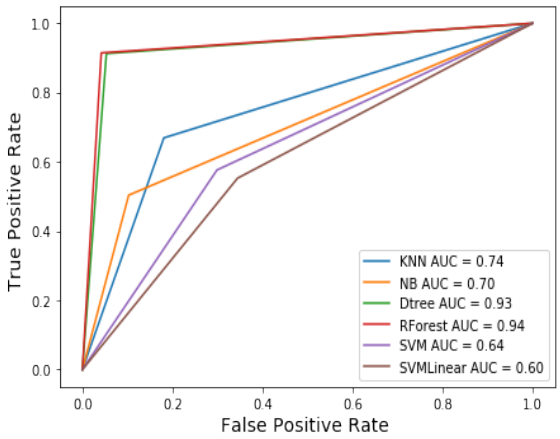
3. Use an embedded feature selection method (using regularized l_1 linear SVC) to select the relevant features. This method estimates the coefficients of each feature and select features with non-zero coefficients as the relevant features.

3.3.4 Machine Learning Algorithm

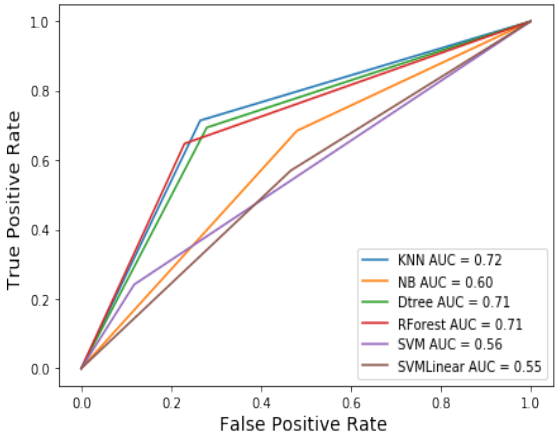
Machine learning is a diverse field, aimed at the same goals, finding and exploiting regularities in training data. Research has shown that no single machine learning algorithm is superior in all cases [75]. One important factor to consider in selection of a machine learning algorithm is the type of the problem and nature of the data. If the data fails to exhibit the statistical pattern that an algorithm exploits then the learning will fail [75]. It is hard to conclude which machine learning algorithm will be best for a problem without experiment. Hence, most evaluation in this field is experimental in nature [75]. Our interest in machine learning is to learn the behaviour of a Twitter bot and provide a conceptual explanations of the behaviour. Research from the literature [88, 89] indicated that a decision tree could be suitable for such a task. However, we experiment with five machine learning algorithms from a different group of classifiers to understand the performance of the decision tree as compared to other algorithms (cf. Section 6.1.1 detailed the data collection and the experiment). The algorithms are as follows, each preceded by the group name. Instance-based - K-nearest neighbour, Bayesian - Gaussian naive Bayes, Kernel-based - support vector machine for classification (SVC) and its linear version (LinearSVC), Tree-based - random forest and decision tree. We experiment with the algorithms on five different Twitter bots' datasets, for each dataset we use 5-fold 10-repeated stratified cross-validation to obtain the overall accuracy, precision, recall and roc_auc score. We use roc_auc score as a final measure of performance between the algorithms because of its robustness to class imbalance, and it is a good indicator of how successful a classifier is in identifying true positive examples.



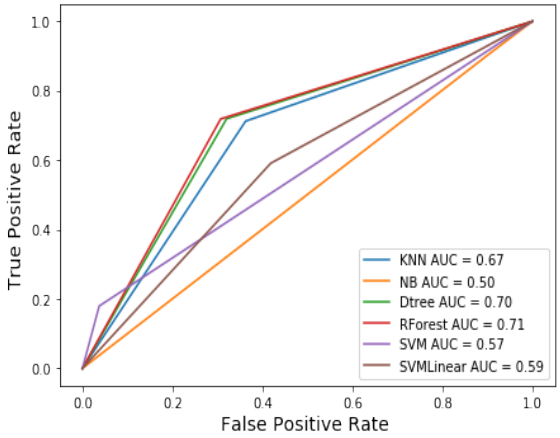
(a) @helpdeskx Dataset



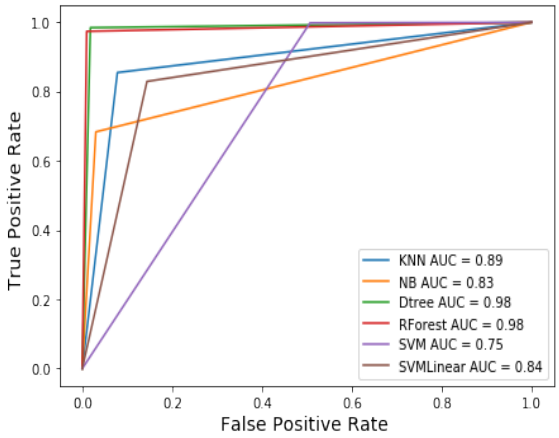
(b) @csinfo_news Dataset



(c) @Don_Vito_08 Dataset



(d) @natespuewell Dataset



(e) @BritainStays Dataset

Figure 3.8: Comparison of machine learning algorithms

Figure 3.8 shows roc_auc curve and score of each classifier on the five datasets. It shows that tree-based classifiers have achieved reasonable score in each dataset. We found that the roc_auc score of KNN and Naive Bayes ranges from 0.50 to 0.89 while that of SVC

and LinearSVC ranges from 0.56 to 0.84. Random forest and decision tree have scores which range from 0.70 to 0.98. This shows that the decision tree performs reasonably well compared with other classifiers.

3.4 Summary

In this chapter, we have presented our first step to describe the behaviour of Twitter bots. We have created a meta-model which provide a conceptual description of the actions and data available to the Twitter bots. We consider this as a knowledge base of the Twitter bots and forms the foundation of our rule representation in the next chapter.

In the second part of the chapter, we have described how we extract features for learning the behaviour of the Twitter bots. We grouped the features into user features, tweet content, network, sentiment and time features. The user features (e.g., account screen-name and location) are set of features describing the Twitter account while tweet content features (e.g., hashtags, URLs and user mentions) are features of the tweet content. As other researchers have used closely similar features, we have described the difference between this thesis and other research on this aspect. We have mentioned that researchers on Twitter bots learn the patterns from the statistical count of the features while we learn the patterns from the actual values of the features and their relationship. For example, in the hashtag feature, they count the number of hashtags per tweet or group of tweets to distinguish bots from human accounts while we record hashtags reacted to or used by a bot to distinguish between two campaigns promoted or opposed by a bot. While many researchers focus on the features of the subject account (bot or human), we extract both features of the subject account and that of the tweets and the users the bot interact with.

Finally, we have described the role of machine learning algorithms in learning patterns and stated the reason why we have selected a decision tree algorithm as a suitable algorithm for learning and describing the behaviour of Twitter bots. We have mentioned that despite the decision trees has limited representation power, it can provide conceptual

explanation of the behaviour. In the next chapter, we describe how we extract rules from a decision tree. We also propose a representation and visualisation which overcome the limitation of the decision tree in describing the behaviour of bots.

Chapter 4

Rule Learning in Twitter Bots

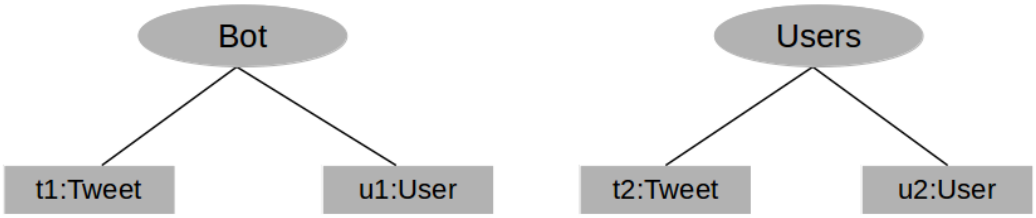
This chapter presents an approach proposed to address the first group of the research questions outline in Chapter 1 (Section 1.2). Imagine being given a software program and your task is to recover its design without seeing its code. In software engineering, this is called reverse engineering. The primary purpose of this could be for re-documentation of a legacy system, analyse a competitor's product to improve own product or to address national security issues. The last two are our objectives for reverse engineering the behaviour of bots. We provide rules describing the behaviour of bots to allow businesses and politicians to discover campaign strategies of their opponent's on social media and improve their own or to allow security agencies to discover extremists or activists activities that could be a threat to national security. In Section 4.1, we describe how to observe the behaviour of a bot and represent it in the form of understandable rules. We describe how to automatically extract rules from observations in Section 4.2, and present an approach which simplifies visualisation and representation of the rules in Section 4.1.

4.1 Rule Representation

In this section, we describe how to observe the behaviour of a bot and represent it in the form of rules. This is our first step in reverse-engineering the Twitter bots. The goal is to

provide a simple but understandable description of the behaviour.

We observe the behaviour of a bot by mining data from its Twitter account and searching for patterns. Patterns are the relationship between bot's action and a set of attribute values. For each retweet or reply action made by a bot, we extract the original tweets and the users who created them. We collect feature values of the tweets and the users in a table of features and analyse. By this, we learn how certain attribute values trigger an action. We provide an overview of the traces' attributes in Figure 4.1. Given a bot (u1:User), we collect traces of its action and behaviour. Depending on the type of action, the traces contain the following four groups of attribute values. 1) Attribute values associated with the bot (u1) to detect changes in the bot's account features. 2) Attribute values associated with tweets produced by the bots (t1:Tweet) to understand their patterns 3) Attribute values of tweets reacted to by the bot (t2:Tweet). These are tweets produced by other users and the bot reacted to in terms of retweets, reply, favorite or follows action. 4) Attribute values of authors of the tweets reacted to by the bot. Figure 4.1 shows an overview of the attributes.



Tweet features
tweet_ID
tweet_create_at
tweet_text
tweet_sentiment
tweet_hashtags
tweet_retweet_count
tweet_favorite_count
tweet_lang
tweet_Country
tweet_City
tweet_tweetCoodinates
tweet_tweetSource
tweet_usrMention
tweet_TweetUrls
tweet_mediaID
tweet_mediaType
tweet_mediaUrl

User features
user_ID
user_screenName
user_created_at
user_statusCount
user_favCount
user_followCount
user_friends_count
user_geo_enabled
user_verified
user_profile_image_url_https
user_lang
user_time_zone
user_account_location
user_description
user_default_profile
user_default_profile_image

Figure 4.1: Overview of the Trace's attributes

Following an analysis of traces from sample Twitter bots (@Don_Vito_08, @natespuewell, @BritainStays,@helpdeskx, @csinfo_news), tools for creating Twitter bots (labnol[1], botlibre[2], jetbots[3], cheapbotsdonequick[4] and botize[5]), and the Twitter API which serves as a source of input data to Twitter bots, we found that Twitter bots use features provided by the Twitter API to search for tweets or users and take action. These form the fundamental

[1]https://www.labnol.org/internet/write-twitter-bot/27902/
[2]http://twitter.botlibre.com/
[3]http://www.jetbots.com/
[4]https://cheapbotsdonequick.com/
[5]https://botize.com/

constructs of their rules. We define a general rule for describing the behaviour of a bot as follows.

Twitter Bot Rule

A Twitter bot rule is composed of a pattern *P* and the invocation of an operation, i.e, IF *pattern* THEN *action*.

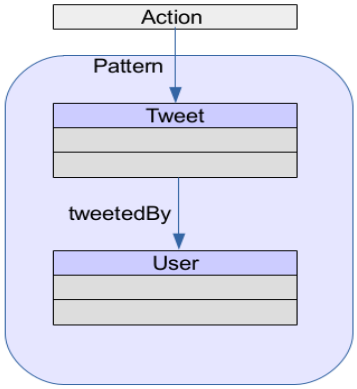


Figure 4.2: A Graphical representation of the Twitter bot rule

Definition 4.1 Let A be a set of Twitter actions, U be a set of user features, and T be a set of Tweet features. Here $A = \{a_1, a_2, a_3, ..., a_n\}$, $U = \{u_{f1}, u_{f2}, u_{f3}, ..., u_{fn}\}$ and $T = \{t_{f1}, t_{f2}, t_{f3}, ..., t_{fn}\}$

Pattern (P) is a collection of *predicates (ϕ)* over attributes of an instance (i) of the Twitter meta-model.

Predicate (ϕ) composed of a feature x , an operator Θ and a value v. Here $x \in$ {U,T,U$'$, T' , $T \cup U$, $(T \cup U)'$ } and U', T', $(T \cup U)'$ is a derivative of U, T, $(T \cup U)$ respectively. The operator (Θ) can be any of the following: $>, \geq, <, \leq, =, \neq, \in$, and the value (v) can be a string or a number.

A pattern of a rule can be based on attributes of tweets and/or users, but can also be more complicated, e.g., considering the follower relation. A typical example of a Twitter bot rule can be : retweet any tweet (t2) which contains a hashtag, $\#x$ or $\#y$ or $\#z$. More

formally, IF h in tweet (t2) THEN $retweet$. Where $h \in H$, and $H = \{x, y, z\}$. In this case, $retweet$ is the bot's action and tweet with hashtags $\#x$, $\#y$, $\#z$ is the pattern which forms part of the rule. It can be argued that x, y and z are the features, but in our rule learning, we treated these as feature values. This is to allow generalization of the bot's behaviour and identification of bots that have similar patterns which may lead to the identification of their masters. The rule can be more complicated combining tweet features such as favorite count or tweet author (u2) features such as status count or both. Below are additional examples of the Twitter bot rules.

Examples:

- *IF* $t2.hashtags =$"$Brexit$" and $t2.retweet\_count > 30$ *THEN* $retweet\ (t2)$
- *IF* $u2.screenName =$"$DonaldTrump$" and $t2.favorite\_count > 10$ *THEN* $retweet\ (t2)$
- *IF* $t2.tweet\_text =$"$StudyMscComputerScience$" *THEN* $reply$ (t2,"Hi Uni of Leicester is good `https://www2.le.ac.uk/departments/informatics/postgraduate`")

As we intend to extract Twitter bot's rules automatically from a decision tree (described in the next section). The proposed rule representation simplifies the representation of the rules generated from the decision tree. Figure 4.3 is a simple example of a decision tree.

```
T2retweet_count >9
|
|       U2screenName = LeicesterUnion: retweet (12.0)
|       U2screenName = CompSciFact: retweet (47.0)
|       U2screenName = LeicStartups: retweet (4.0)
|       U2screenName = ULSUWellbeing: retweet (3.0)
|       U2screenName = bcs: retweet (22.0)
|       U2screenName = Microsoft: retweet (14.0)
|       U2screenName = uniofleicester: retweet (44.0)
T2retweet_count <= 9
|
|       U2screenName = GaetaSusan: notRetweet (1.0)
|       U2screenName = thinkinglatina: notRetweet (1.0)
|       U2screenName = HersheSquirt: notRetweet (3.0)
|       U2screenName = Nigel_Farage: notRetweet (1.0)
|       U2screenName = FLOTUS: notRetweet (1.0)
|       U2screenName = Corrynmb: notRetweet (2.0)
```

Figure 4.3: A textual representation of a simple decision tree from a Twitter Bot

The tree can be express as a collection of production rules of the form *IF* $left-hand\ side$ *THEN* $class$ i.e.;

1. *IF* $t2.retweet\_count > 9$ and $u2.screenName = LeicesterUnion$ *THEN* $retweet$
2. *IF* $t2.retweet\_count > 9$ and $u2.screenName = CompSciFact$ *THEN* $retweet$
3. *IF* $t2.retweet\_count > 9$ and $u2.screenName = LeicStartups$ *THEN* $retweet$
4. *IF* $t2.retweet\_count > 9$ and $u2.screenName = ULSUWellbeing$ *THEN* $retweet$
5. *IF* $t2.retweet\_count > 9$ and $u2.screenName = bcs$ *THEN* $retweet$
6. *IF* $t2.retweet\_count > 9$ and $u2.screenName = Microsoft$ *THEN* $retweet$
7. *IF* $t2.retweet\_count > 9$ and $u2.screenName = LeicesterUnion$ *THEN* $retweet$

Following this, we have seven sets of rules for the $retweet$. However, if we apply our idea of rule representation using the meta-model and graph transformation, we will have more concise and general representation of the rules as follows.

- *IF* $t2.retweet\_count > 9$ and $u2.screenName \in U2screenName$ *THEN* $retweet\ (t2)$ here $U2screenName = \{CompSciFact, uniofleicester, bcs, Microsoft, LeicesterUnion, LeicStartups, ULSUWellbeing\}$.

This simplifies the representation of the rules and allows generalization to cover the unseen usernames. Suppose we get an additional username (e.g., UoLInformatics) in a new observation. The rule pattern will remain the same. We only need to update the list of the usernames.

4.2 Rule Extraction

After successful completion of the first task (observe and represent the behaviour of a bot in the form of understandable rules), we seek the answer of our next question, i.e., how

can we extract such information (how and when a bot takes actions) automatically from observations? Machine learning is a known approach for knowledge mining, discovering patterns and correlations in a pre-existing dataset. Our goal is to provide conceptual explanation of the relationship between the input variables (tweet and user features) and the output variables (bot actions). Decision tree is a popular method to capture the relationship between input and output variables, but it has limited representation power [70]. It can become very complex even for simple relationships. Using the concept of graph transformation, we propose a novel rule-based approach to represent the same information. We employ graph transformation concepts to generate rules from a decision tree for two purposes. First, general rules can cover unseen examples and combine leaf nodes of the same type. Second, they simplify and enhance the readability of the model.

Once we generate a decision tree, we traverse the tree. From the root, we follow each branch and collect set of decisions leading to each leaf node (class) as a rule instance. Then we group rule instances according to classes and remove redundant cases. For each class, we look at its set of rule instances and remove those that are already covered by other rules with a larger number of examples. Figure 4.4 provides an overview of the rule extraction process.

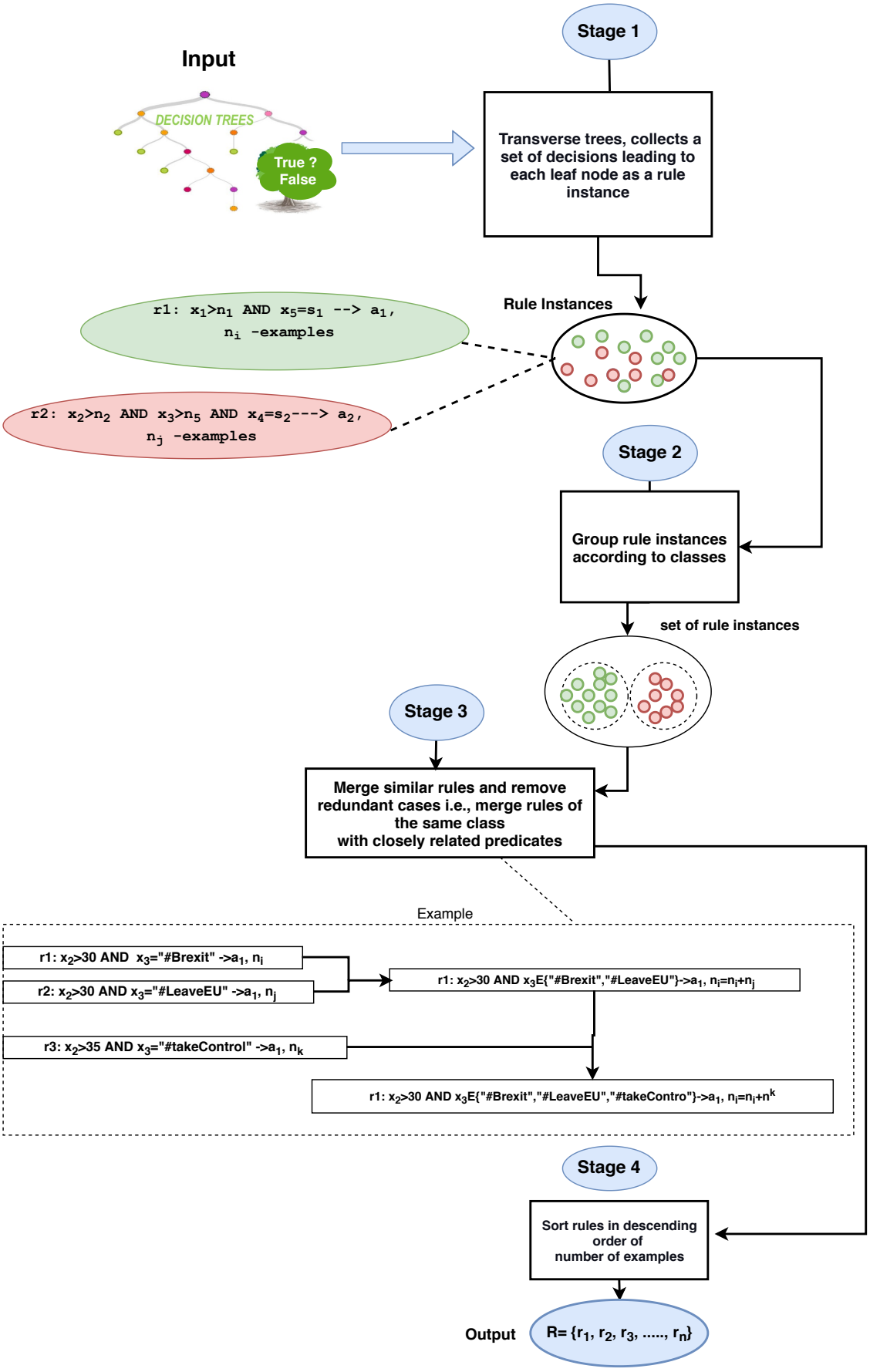


Figure 4.4: An overview of the rule extraction process

Given a decision tree, stage 1 extracts decisions leading to each leaf node as a rule instance e.g., "$r_1 := x_1 > n_1$ AND $x_5 = s_1 \implies$ a$_1$, n_1 examples". Here r_1 is a rule instance leading to action (a_1), x_1 and x_5 are features, n_1 is a numeric value, s_1 is a string value, and n_i denotes the number of examples covered by the rule (r_1). Stage 2 groups rule instances according to class (action a_i). Stage 3 focusses on creating a more compact representation of the rules by merging rules of the same class with related predicates. The merging process starts with a collection of rules with the same features. In the example shown in Figure 4.4, r_1, r_2 and r_3 are rules of the action (a_1). The rules are merged according to the similarity of the predicates. The process starts with merging of r_2 with r_1 because of the predicate ($x_2 > 30$) which is common in both r_1 and r_2. In the merging process, string values of the same feature (x_3) are collected as a set ("Brexit","LeaveEU"). Predicates with similar numerical features and the same relational operator but different in values are merge together by selection of the predicate that covers both, e.g, $x_2 > 30$ covers $x_2 > 35$. Whenever two rules are merged the number of examples will be added together (e.g., n_i = n_i + n_j). Finally, the rule extraction process produces a set of rules sorted in decreasing order of the number of examples covered by the rule ($R := \{r_1, r_2, r_3$,.....,r_n }).

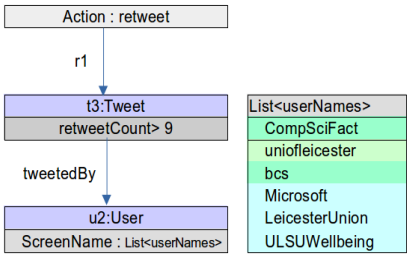
4.3 Rule Visualisation

Visualisation is an effective way to communicate both abstract and concrete concepts. When working with symbolic knowledge such as the production rules described in Section 4.1, providing a visual representation will make them more readable and interpretable. We introduce a rule visualisation which separates the high-level representation of rules from the data. For each action, our visualisation produces a graphical representation of a set of rules $\{r_1, r_2, \ldots, r_n\}$ formed from the set of attributes used by the bot. A tabular representation of values associated with nominal attributes supplies possible data values. The tabular representation is colour-coded based on sentiment. The family of green colour represents positive sentiment and that of red colour indicates negative sentiment. The val-

ues are sorted in descending order of their distribution in the data, but we intend not to include any numerical representation of their distribution because it is independent of the behaviour of the bot. Figure 4.5b shows a simple example of our visual representation. The visualisation simplifies the representation of the rule and provides additional details (sentiments).

```
T2retweet_count >9
|
|       U2screenName = LeicesterUnion: retweet (12.0)
|       U2screenName = CompSciFact: retweet (47.0)
|       U2screenName = LeicStartups: retweet (4.0)
|       U2screenName = ULSUWellbeing: retweet (3.0)
|       U2screenName = bcs: retweet (22.0)
|       U2screenName = Microsoft: retweet (14.0)
|       U2screenName = uniofleicester: retweet (44.0)
T2retweet_count <= 9
|
|       U2screenName = GaetaSusan: notRetweet (1.0)
|       U2screenName = thinkinglatina: notRetweet (1.0)
|       U2screenName = HersheSquirt: notRetweet (3.0)
|       U2screenName = Nigel_Farage: notRetweet (1.0)
|       U2screenName = FLOTUS: notRetweet (1.0)
|       U2screenName = Corrynmb: notRetweet (2.0)
```

(a) A textual representation of a simple decision tree from a Twitter Bot



(b) A representation of the decision tree in Figure 4.5a using the notion of our general rule for Twitter Bots

Figure 4.5: Rule Visualisation

Algorithm

The proposed algorithm for rule visualisation is presented in Algorithm 1. The algorithm takes rules extracted from a Twitter bot and produces a graphical representation of each rule as shown in Figure 4.5b.

Definition 4.2 Let $(Graph(T^n, U^n) \rightarrow a)$ be an instance of a meta model. Here T^n, U^n, a, represent tweet node, user node, and action respectively.

Let δ' be a list of pairs (nominal feature value, sentiment). Let $\mathcal{N}$ be a set which contains tweet and user features with numerical values, such that $\mathcal{N} \subset T \cup U$. Here T is a set of tweet features, and U is a set of user features.

Algorithm 1: Rule Visualisation

Input : $R, a, botData$. Here R is a set of rules for an action (a) extracted from traces ($botData$). $R := \{r_1, r_2, r_3, ..., r_n\}$, r_i is a collection of a predicate (ϕ) and set of feature values δ_i such that δ_i is a list of values of a nominal feature x_i.

Output: Instance of a meta-model $(Graph(T^n, U^n) \rightarrow a), \delta'$.

```
1  Begin
2  R ← {r_1, r_2, r_3, ....., r_n }
3  foreach rule(r) ∈ R do
       /*  create δ' to store a list of feature values, their
           sentiment                                              */
4      δ' ← {}
5      foreach predicate(φ_i) ∈ r do
           /*  Compute the sentiment of values associate with
               nominal features                                   */
6          if feature(x_i) ∈ φ_i AND x_i ∉ 𝒩 then
7              δ'_i ← {}
8              foreach value(v) ∈ δ_i do
9                  s ← AvgSentiment(v, botData)
10                 δ'_i.insert({v, s})
11             end
12         end
           /* Insert predicate(φ) which contains a tweet feature
              into tweet node(T^n) and predicate which contain a
              user feature into user node(U^n)                   */
13         if feature(x_i) ∈ T) then
14             T^n.insert(φ_i)
15         else
16             U^n.insert(φ_i)
17         end
18         δ'.insert(δ'_i)
19     end
20     Output Graph(T^n, U^n) → a, δ'
21 end
22 End
```

The algorithm takes a set of rules $R := \{r_1, r_2, r_3, ..., r_n\}$ for a given action (a) and produced an instance of a meta-model ($Graph(T^n, U^n) \rightarrow a$) together with a list of pairs (δ') containing values and sentiments of the nominal features in $Graph(T^n, U^n)$. An example of the visual representation of $Graph(T^n, U^n) \rightarrow a$, δ' is shown in Figure 4.5a.

4.4 Summary

In this chapter, we have presented an approach for reverse engineering the behaviour of a bot. Given a bot account, we have shown that its behaviour can be describe in the form of production rules, i.e., *IF* $pattern$ *Then* $action$. We chose to express the behaviour in the form of production rules because production rules are widely used and well-understood structures for representing knowledge, more importantly, they facilitate symbolic reasoning and inference.

To automatically observe the behaviour of a bot and extract rules, we proposed a machine learning based approach; the approach utilises decision trees which are popular in providing a conceptual explanation of the relationship between input and output variables. To tackle the issue of representation which is commonly known with decision trees, we have proposed a rule visualisation approach which uses the background knowledge (Twitter API meta-model) and concepts of graph transformation to simplify visualisation of the rules. The visualisation provides additional information about the behaviour by encoding sentiments into the rules.

In the next chapter, we have described how we study the bias and antagonism of a bot.

Chapter 5

Opinion, Bias and Antagonism of a Bot

Social Twitter bots are mainly orchestrated to promote a specific agenda and opinion. Understanding these in detail is an interesting, but challenging task. This chapter presents an approach proposed to address the second group of the research questions outline in Chapter 1 (Section 1.2). Consider a social bot on a network of users with a mixture of opinions on various topics. The behaviour of the bot in terms of content produced or reacted to has a significant role in understanding its impact on the network. We describe how to identify topics of interest to a bot and detect opinions promoted by a bot in Section 5.2 . In Section 5.3, we introduce the notion of differential sentiment (DS) to provide a means of understanding the behaviour of a bot in relation to its sources and target audience. We introduced differential sentiment over time (DSO) (in Section 5.4) to provide a way of identifying a period where a bot has a higher chance of marking an impact on the network.

5.1 Introduction to Running Example

In the 2016 presidential election of the United States bots were used by supporters of both Donald Trump and Hillary Clinton. Researchers have made this public and raised many interesting questions [6, 8, 41]. In this chapter, we use sample bots from the US

election as a running example to demonstrate our approach for the study of contents, opinion and role of bots in online campaigns. Since the rise of bots on social media, researchers have made a significant effort toward the detection of bots. This has helped to detect and block thousands of bots from social media platforms. However, this thesis argues that since not all bots are harmful, as bots become more sophisticated and harder to detect; it will be important to understand their behaviour, contents and role. This will help to distinguish harmful bots from useful ones and help to ascertain their impact in an online campaign. Given a set of bots $B = \{$*@Don_Vito_08*, *@natespuewell*$\}$, we asked the following questions.

- RQ1: How can we identify campaign topics and detect opinions promoted by the bots?
- RQ2: How can we express bias and antagonism of the bots?
- RQ3: How can we track changes in bias and antagonism of the bots?

@Don_Vito_08 is a Donald Trump supporting bot while *@natespuewell* is a bot supporting Hillary Clinton. We will be using these as a running example throughout this chapter to address the above questions. By the end of this chapter, we aim to provide details about the bots, leading to the understanding of their role in the 2016 US election.

5.2 Opinion of a Bot

Social bots connect and interact with users on social networks, engage in the discussion of different topics. However, the main goal of the bots is to promote a specific agenda in a conceivable manner. Considering that users communicate on social media using a short text, how can we identify topics of interest to a bot and detect the opinions promoted by a bot? This is important for understanding its role and will be a starting point for understanding its bias and antagonism with the users on its network.

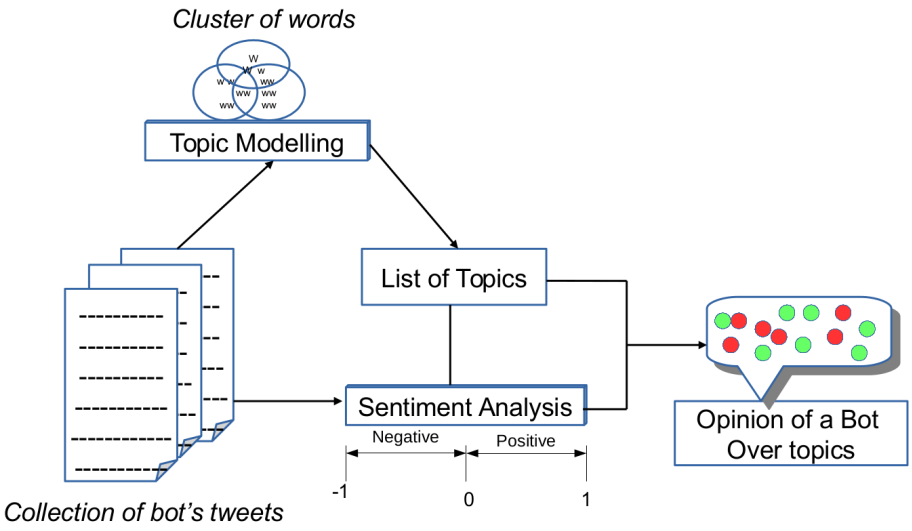


Figure 5.1: Overview of how we learn the opinion of a bot

Figure 5.1 provides an overview of our approach to detect topics of interest to a bot and learn its opinion on the topics. To detect topics of interest to a bot, we collect a set of tweets (tweets produced by the bot and tweets it reacted to in terms of retweets or replies), convert them into a collection of bi-grams after all the pre-processing steps (see Section 3.3.1) and use the topic modelling technique described in (Section 2.5.1) to compute a set of topics of interest to the bot. To learn the opinions of a bot on topics, we cluster the bot's tweets according to the topics and compute the mean absolute sentiment of all tweets belonging to each topic. We define the mean absolute sentiment as follow:

Mean Absolute Sentiment (MAS)

Let T be a set of tweets on a topic x, and $s(x,t)$ be the sentiment of tweet $t \in T$ on topic x . We define mean absolute sentiment as :

$$MAS\ (x) = \frac{\sum_{t \in T} s(t,x)}{card(\{t \in T | s(t,x) \neq 0\})} \tag{5.1}$$

The mean absolute sentiment serves as a means to understand the overall opinion of a bot on a particular topic. In a situation where a bot shares messages with an opinion opposite

to its opinion to conceivably attract users and later persuade them toward its opinion, the MAS will enable us to detect the actual opinion of the bot. In Section 5.2, we have shown how we use MAS to detect opinions of bots during the 2016 US presidential election. We have also use MAS to identify strongly opinionated accounts 7.2.

Running Example Part 1:

RQ1: How can we identify campaign topics and detect opinions promoted by the bots? To understand the campaign of *@Don_Vito_08* and *@natespuwell*, we collect 3,200 tweets from their timeline via the Twitter API, clean and pre-process the tweets as described in Section 3.3. We use a topic modelling technique (see Section 2.5.1) to compute the sets of topics in the tweets and mean absolute sentiment (MAS) to find the opinion of the bots on each topic as described in our approach above.

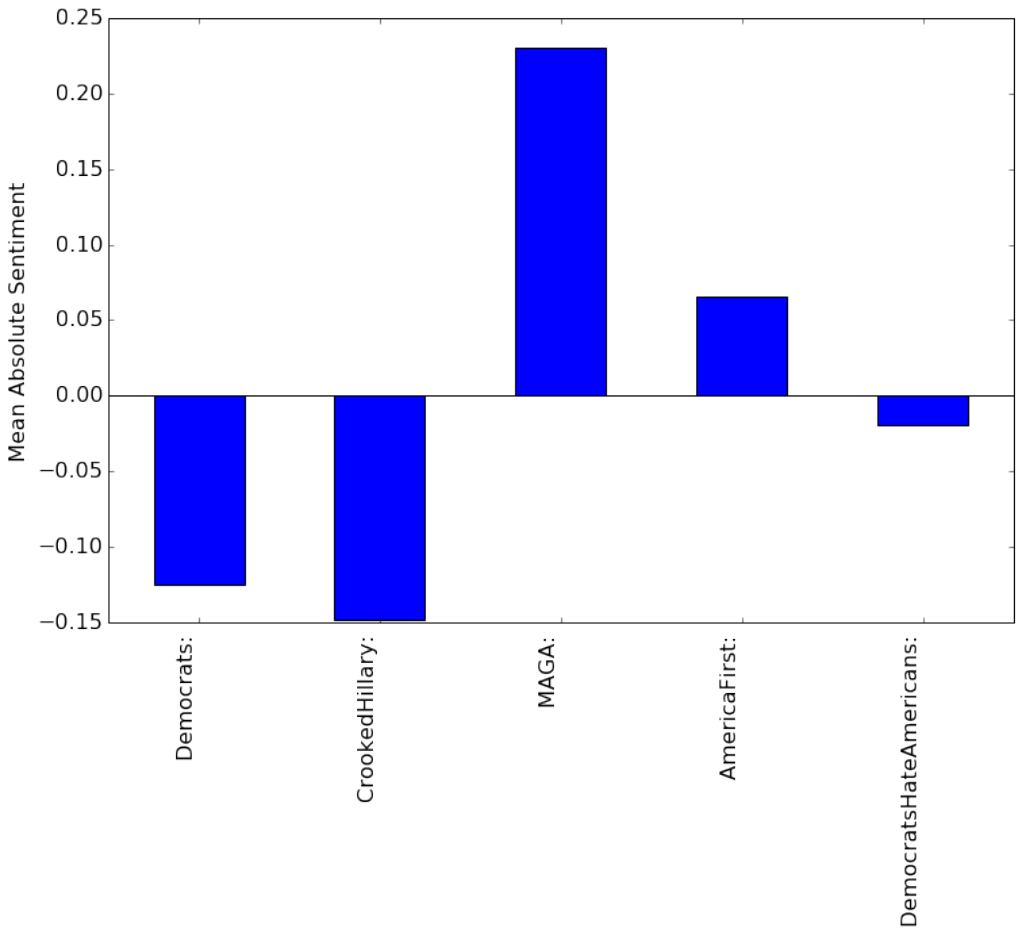


Figure 5.2: *@Don_Vito_08* campaign topics and its opinion on the topics

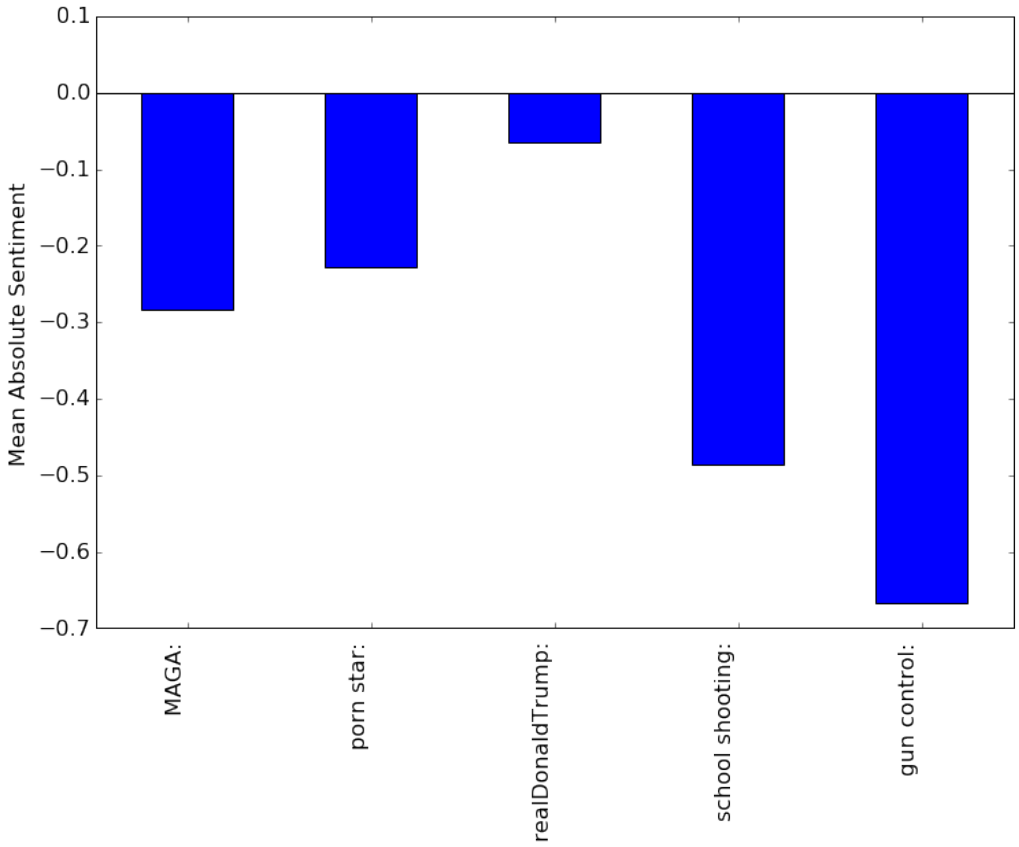


Figure 5.3: @*natespuwell* campaign topics and its opinion on the topics

Figure 5.2 and 5.3 are results obtained using the proposed approach to understand campaign topics and opinions of @*Don_Vito_08* and @*natespuwell*. From Figure 5.2, we can notice that @Don_Vito_08 campaigns for Donald Trump by promoting its campaign tags *MAGA* (Make America Great Again) and *AmericaFirst*. The bot opposes *Democrats* and *Hillary Clinton* by propagating negative tweets toward them. From Figure 5.3, we can see the campaign of @*natespuwell*. The bot opposes Donald Trump by propagating negative tweets using *MAGA, realDonaldTrump, porn star, school shooting and gun control*. The results delivered by our approach could assist researchers in social science to answer questions such as 1) Which among these bots has a better campaign approach? 2) Which among these bots is using words of psychological manipulation? (Words capable of having a psychological impact on the voters' opinion.) The results show @*Don_Vito_08* promotes Donald Trump as well as against Democrats and Hillary while @*natespuwell* concentrates more on opposing Donald Trump than promoting Hillary. This is why we did

not find topics which promotes Hillary among its top campaign topics. In terms of using words of psychological manipulation, we notice that @*Don_Vito_08* campaigns for Donald Trump by promoting that he will make America great, he will consider AmericaFirst, and its opposes Hillary Clinton by referring to her as CrookedHillary, and Democrats by saying Democrats hate Americans. Contrarily, @*natespuwell* mainly uses what transpired between Donald Trump and the American porn star Stormy Daniels, and the issues of gun control in which Donald Trump proposed to abolish gun-free zones[1](i.e., allow school teachers and civilians to hold guns) during his campaign. Next, we look at how the opinions change over time.

Mean Absolute Sentiment over time (MASO)

Opinions of a bot on a topic can change over time due to (1) a strategic plan to attract users of different opinion or (2) response to external events. In the first case, we can see the bot's opinion changes in opposite direction while in the second one, we see the bot's opinion increase or decrease in the same direction. We propose the use of mean absolute sentiment over time to track changes in opinion and understand the response of a bot to an external event. In Section 5.4, we use mean absolute sentiment over time to understand the relationship between the opinion of a bot (@Don_vito_08) and that of its followers. We also use MASO in Section 7.2 to study how the bots change their policy position over time. In Section 7.3, we use MASO to study how the opinion of the online users changes over time during the election period.

In the next section, we propose an approach to understand the bias and antagonism of a bot.

[1]https://www.nytimes.com/2016/05/20/us/politics/hillary-clinton-donald-trump-gun-control.html

5.3 Differential Sentiment

In this section, we aim to address two important factors associated with a bot's opinion, i.e., bias and antagonism. We view antagonism of a bot as an attempt of influence. The concept of antagonism and influence has been widely discussed in the area of social psychology but new in the area of online social networks (OSNs) which is a very young research area [31,71,93,97]. The most popular platforms are not older than three decades. Facebook was launched in February 2004 and Twitter in March 2006. Researchers of this aspect in an online social networks focus mainly on identifying influencers and measuring or quantifying user influence [3, 7, 72, 76, 79, 114]. We look at the bias and antagonism in a differential way focusing on sentiment intensity of the content shared between the users rather than user's social attributes such as the number of followers, friends, mentions, and retweets as considered in [79] and [7]. We argue that while the user social attributes could indicate potential influencers, those attributes do not signify influence. For example, suppose a user created a tweet, and such tweet is retweeted by one thousand users. This is not enough to define the influence of the user on the other users. The opinion of the two parties should to be considered.

Generally, social influence can be defined as "change in a person's cognition, attitude, or behaviour, which has its origin in another person or group" [93]. We tend to be more specific with the term "Social Influence" than the general definition. We regard the type of influence that happens in the online social network as informational social influence [31]. Influence is defined here as a change in the user's opinion as a result of information received by another user or group of users. Such a change in opinion can be constructive or destructive to the initial opinion of the user. However, since other external factors may affect users' decisions, evidence of influence from social media could be difficult to prove. Hence, we can only observe and measure the degree of antagonism between a bot and users on its network.

Given a Twitter bot in a network of opinionated users, how can we express bias and antagonism of the bot? Figure 5.4 is an overview of how we express the bias and antagonism

of a Twitter bot. We propose an approach called differential sentiment (DS) for the study of bias and antagonism of a bot.

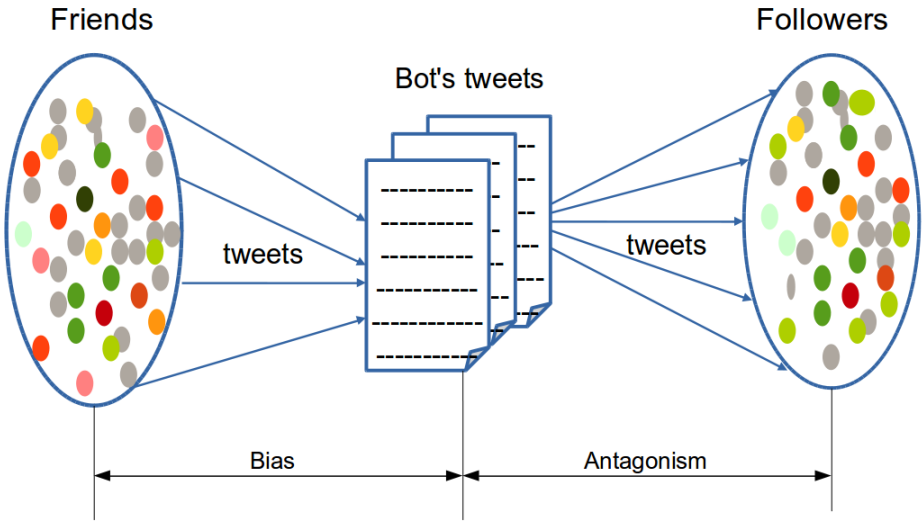


Figure 5.4: Bias and antagonism of a bot

- *Differential Sentiment:* we introduce the notion of differential sentiment (DS) to understand the behaviour of a bot in relation to the users on its network. We distinguish two forms of differential sentiment, bias and antagonism. For each topic, as depicted in Figure 5.4, we define the bot's *bias* as the difference in sentiment between the tweets produced by the bot and the tweets on this topic issued by the bot's friends, i.e., users which the bot is following. The difference in sentiment between the tweets produced by the bot and those of its followers defines the bot's *antagonism* as depicted in Figure 5.4.

- *Recall Equation 5.1* : Mean Absolute Sentiment (MAS)

- Differential Sentiment (DS)

  The differential sentiment of a bot on a topic x is the difference in opinion between a bot and users on its network (f). We formally define a differential sentiment of a bot $DS(x)$ on a topic x as follows.

$$DS(x) = MAS(x)_{bot} - MAS(x)_f \tag{5.2}$$

Here, $MAS(x)_{bot}$ represents the mean absolute sentiment of a bot on a topic (x), and $MAS(x)_f$ is the mean absolute sentiment of users (f) on a topic (x).

$$f = friends \quad for \quad Bias \quad or \quad f = followers \quad for \quad antagonism$$

In the following running example, we demonstrate how to use the differential sentiment to study bias and antagonism of bots.

Running Example Part 2 :

RQ2: How can we measure the bias and antagonism of the bots? In part 1, we have seen the campaign topics of *@Don_Vito_08* and *@natespuwell*. In this part, we show how we use the proposed concept of differential sentiment to study the relationship between the bots and users connected to their network (both their friends and followers). Specifically, we asked what their bias and antagonism is? This is important for understanding their role and impact. Figure 5.5 and 5.6 show the resulting differential sentiments of *@Don_Vito_08* and *@natespuwell*. *@Don_Vito_08* is biased opposite to its friends with regards to HillaryClinton and AmericaFirst. It promotes negative tweets about Hillary Clinton while users on its network are generally positive. This is seen as a clear sign of bias, and it can also serve as a strategy to convert users, trying to change their opinion. *@natespuwell* is biased opposite to its friends with regards to MAGA (Make America Greater Again) but shares their opinion on other topics. Looking at the results of the two bots, *@Don_Vito_08* engages in both positive and negative campaigns while *@natespuwell* is mainly negative. In particular, while *@Don_Vito_08* promotes tags from Donald Trump, it is also campaigning against Hillary Clinton. *@natespuwell* mainly opposes Donal Trump without specifically promoting content with regards to Hillary.

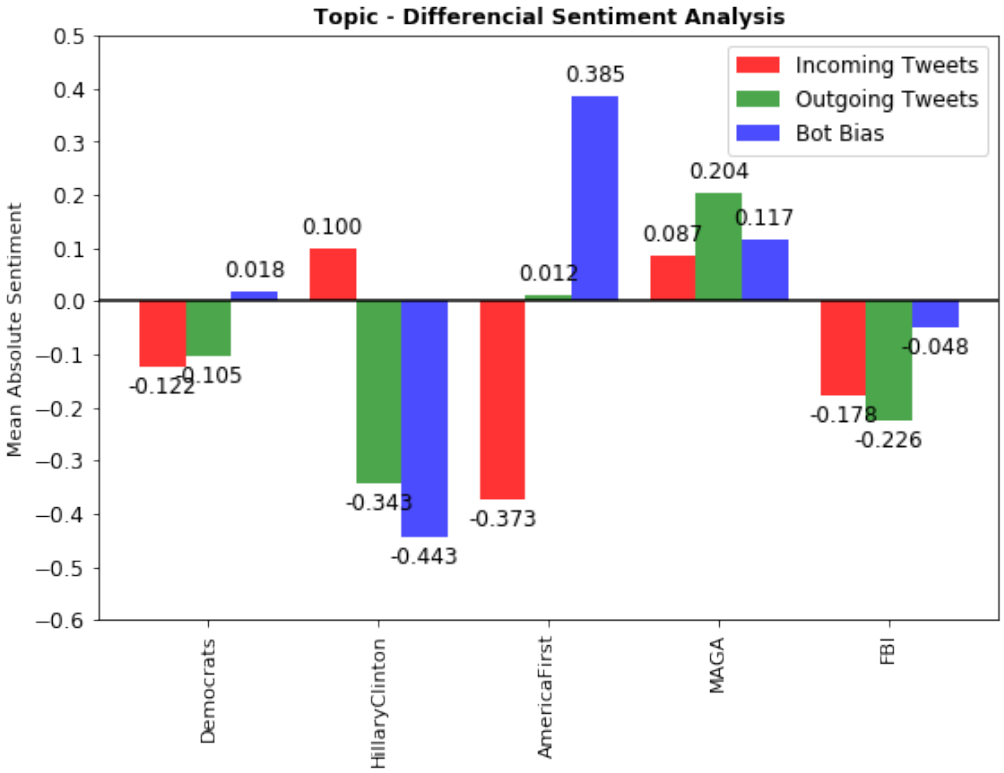


Figure 5.5: @*Don_Vito_08* opposes its friends w.r.t. to *Hillary Clinton* and *America First*

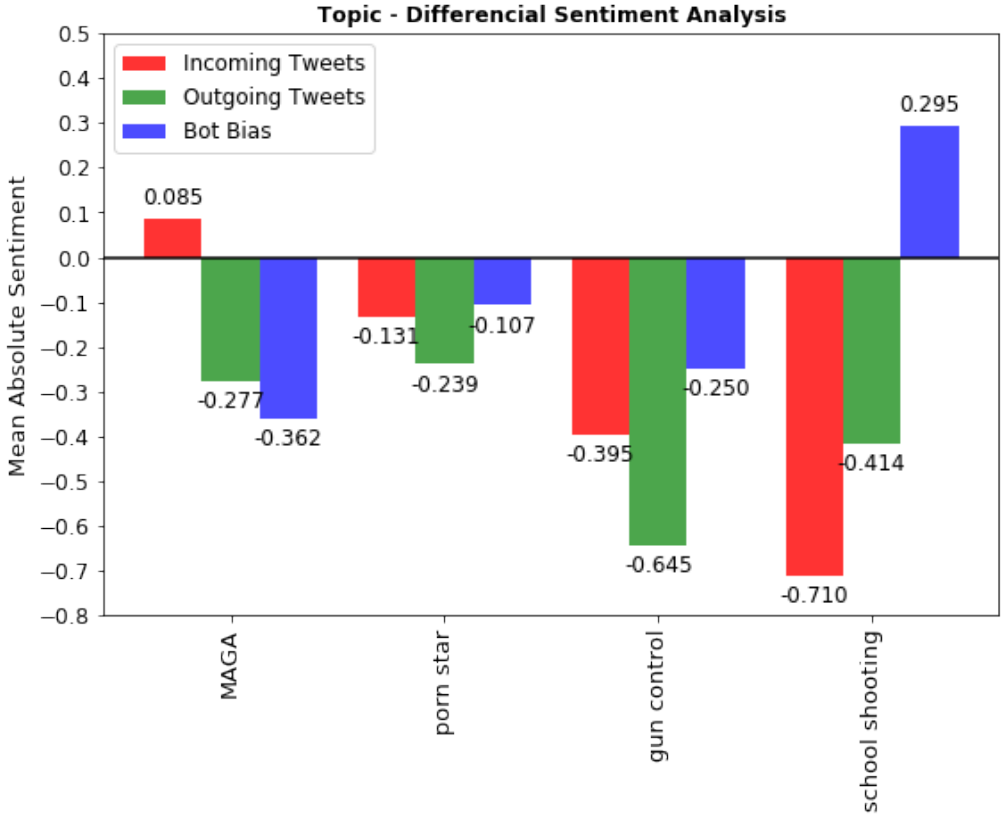


Figure 5.6: @*natespuwell* opposes its friends w.r.t. *MAGA* but agrees on other topics

In addition to the detection of bias, we use the differential sentiment to study the antagonism of the bots. As shown in Figure 5.7 for @*Don_Vito_08*, majority of the bots'(@*Don_Vito_08* and @*natespuwell*) followers have the same opinion with the bots on all topics. Even if they agree, we notice that @*Don_Vito_08* is more strongly against Hillary Clinton and supports MAGA. The followers promote the tag AmeracaFirst more than the bot.

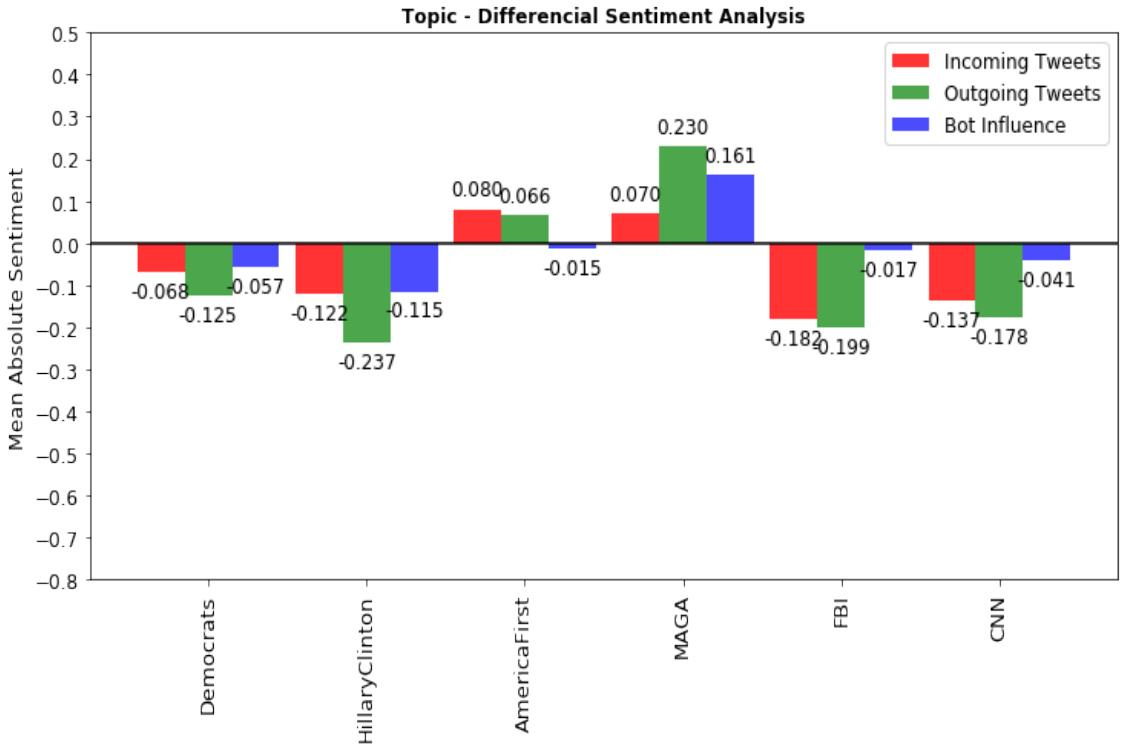


Figure 5.7: @*Don_Vito_08* shares the opinions of its followers on all topics

In Section 7.2, we also use the DS to study the behaviour of bots campaign for Scottish independence. In Section 7.3, we used the DS to study the relationship between news media and online users during the 2019 Nigerian election.

While we can use the differential sentiment to understand the relationship between the opinions of two different parties, we seek how to understand the evolution of such relationship over time. We introduce differential sentiment over time to address this.

5.4 Differential Sentiment Over time (DSO)

We introduce the notion of differential sentiment over time to study the evolution of opinion over time to detect changes in bias and antagonism. Recording the bot's sentiments for specified time periods, we slice the set of tweets based on the given intervals and compute the average sentiment of each topic. If there are not enough tweets in a given interval, we double the length of this interval. The average sentiment of the bot's followers and friends on the relevant topics are computed for the same time spans. In Section 5.4, we use DSO to detect changes in antagonism of the bots.

Running Example Part 3 :

RQ3: How can we track and detect changes in antagonism of the bots? We use differential sentiment over time (DSO) to provide more insights into how the relationship between the bots and their followers evolve. Figure 5.8 shows *@Don_Vito_08* differential sentiment over time.

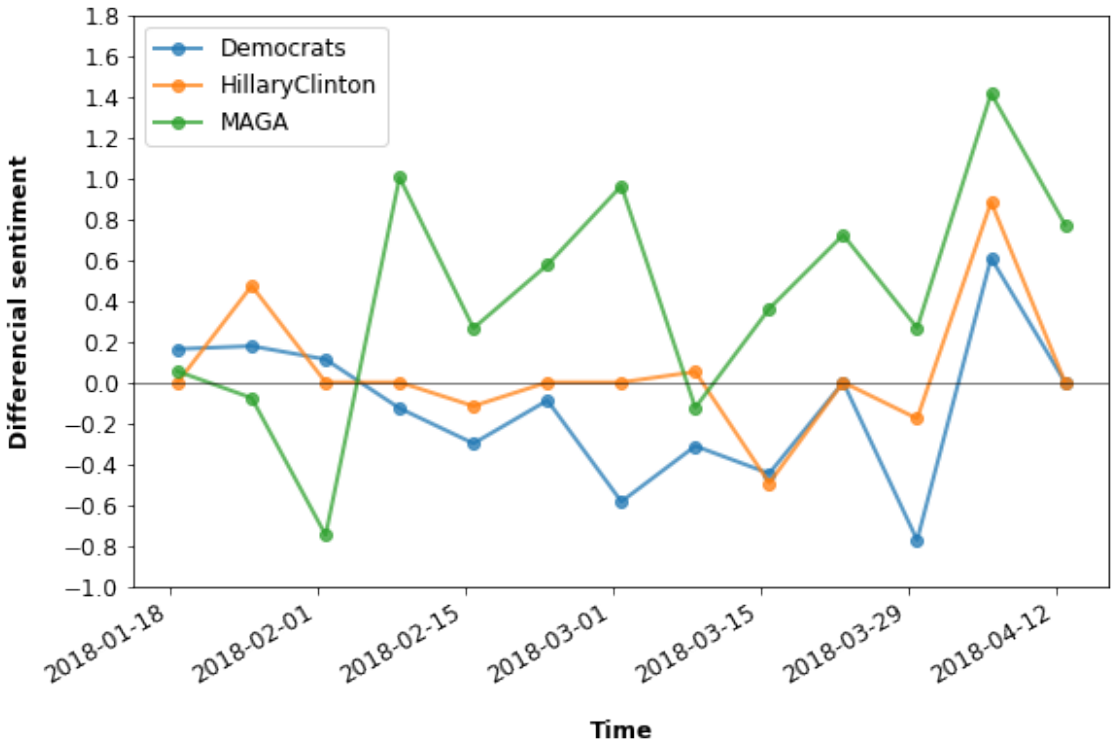


Figure 5.8: *@Don_Vito_08* differential sentiment over time

The differential sentiment over time (DSO) show us how the bot's antagonism changes over time. Given the overall polarity of a bot's differential sentiment (DS), any point that is opposite to that polarity on DS over time means the bot may have had little impact on the network with respect to that opinion at that time. For example, Figure 5.7 shows that *@Don_Vito_08* is negative toward Hillary Clinton, but if we look at its DS over time in Figure 5.8, we notice that in week 2 (25-01-2018) and in the second to the last week of the analysis (05-04-2018), the DS turns out positive. Hence *@Don_Vito_08* seems to have no impact against Hillary Clinton in these weeks. If we look at week 9 (15-03-2018), we notice that it is a period where *@Don_Vito_08* could have made a significant impact. DS over time helps to identify periods where a bot could have had an impact on a network. However, since there are many other factors, we only observe correlations, not causal links.

The absolute sentiment over time simplifies the understanding of the DSO over time. Figure 5.9 shows the absolute sentiment of @Don_Vito_08's followers over time on Hillary Clinton. We can see that in the second to last week (05-04-2018, week 12), the bot is less negative against Hillary Clinton then its followers. This is a simple explanation of why the DS over time is positive in that week to show that the bot has contribute less. Moreover, the absolute sentiment over time provides a view of how sentiment of a bot and its followers change independently. Figure 5.10 provides a view of how @Don_Vito_08 dominated the network with tweets supporting MAGA more strongly than its followers.

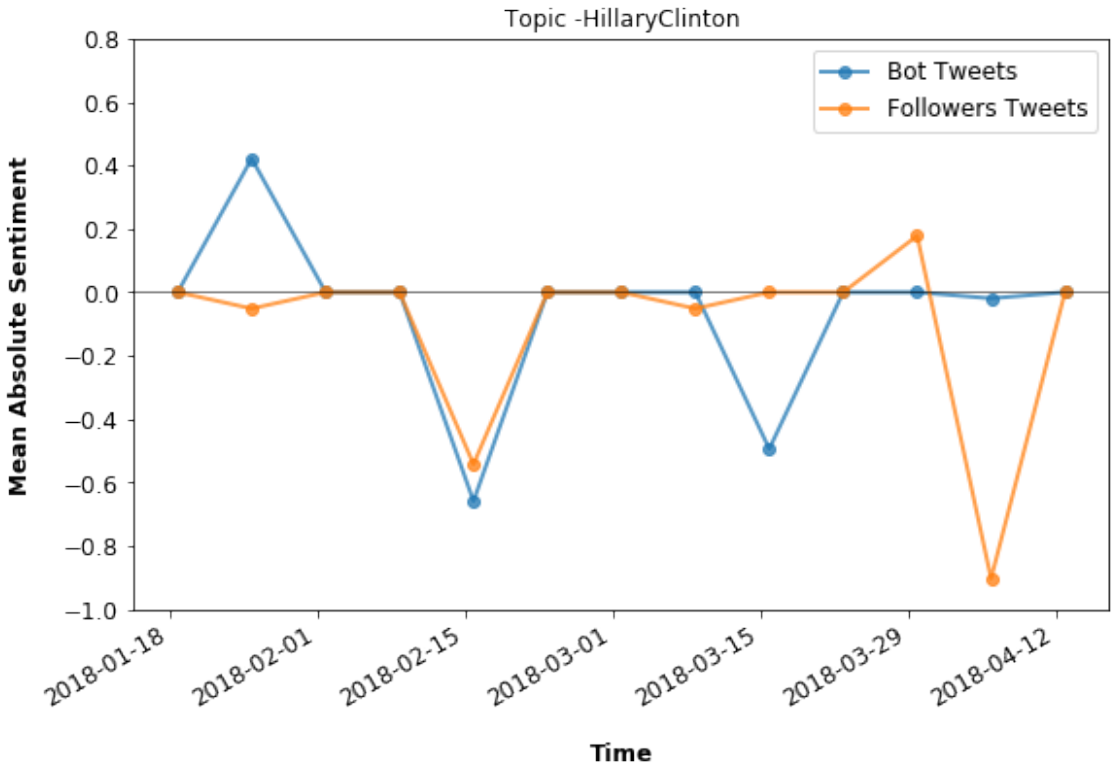


Figure 5.9: *@Don_Vito_08*'s follower's absolute sentiment over time on *Hillary Clinton*

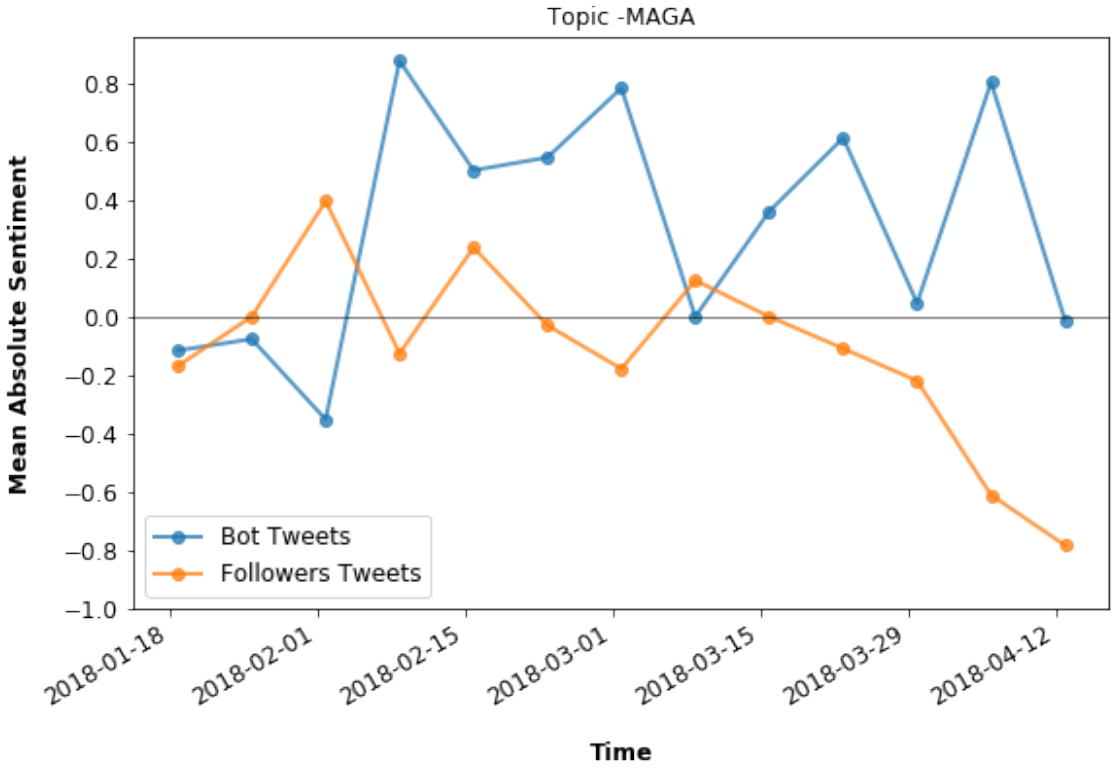


Figure 5.10: *@Don_Vito_08*'s follower's absolute sentiment over time on *MAGA*

5.5 Summary

In this chapter, we proposed an approach to understand topics, opinion and role of a bot in an online campaign. In Section 5.2, we described an approach to understand campaign topics and opinions promoted by a bot. The approach uses topic modelling techniques to find campaign topics from a collection of tweets. It groups bot's tweets according to topics and uses mean absolute sentiment (MAS) to detect the opinion of the bot on each topic. Using two bots (@*Don_Vito_08* and @*natespuwell*) from the 2016 US presidential election, we have shown how the approach provides results which could be useful to researchers to answer social questions such as which bot uses a better campaign approach? Which bot uses words of psychological manipulation? i.e, words that could have impact on the voters' opinions.

In Section 5.3, we proposed the notion of differential sentiment (DS) to understand the bias and antagonism of a bot. The differential sentiment can be used to study the relationship between a bot and other online users (its followers and friends). This is useful for understanding the position of a bot in a network and its role during an event such as an election campaign or public protest. In Section 5.4, we described how to study the evolution of the relationship between a bot and its followers over time using a proposed approach called differential sentiment over time (DSO). Using (@*Don_Vito_08* and @*natespuwell*) as a running example we use DSO to detect period where the bots could have an impact on their followers.

Concluding, the approaches presented in this chapter will contribute significantly to the understanding of the role and impact of bots in an online campaign. This is demonstrated by the running example presented in this chapter and case studies presented in Chapter 7.

Chapter 6

Evaluation

The goal of this chapter is to investigate, empirically, the effectiveness and limitations of the proposed approach. In Section 6.1, we assess the correctness and completeness of the approach using selected real Twitter bots and prototype bots. In Section 6.2, we assess the performance and scalability of the approach. More precisely, we measure the performance of the approach on different Twitter accounts (bots and humans), determine how well the approach scales with a large number of observations, and the time cost of each step as the number of observations increases.

6.1 Correctness and Completeness

One of the most significant challenges facing reverse engineering approaches is that of incomplete and/or inaccurate representation of the system [10,17,38]. This is because the information is derived from a collection of traces that represent only those behaviours that are actually executed or observed. In some cases, the derived information is inaccurate because the model excludes important behaviour or includes irrelevant information. The approach proposed in this thesis is not far away from these problems. Hence, we define correctness and completeness as follows: *Correctness* means for every behaviour/rule *R* in the implementation, there exist corresponding instances of the behaviour/rule in the

traces. *Completeness* means for each instance *I*, representing a behaviour in the collected traces, there exists a representation/rule *R* produced by the implementation which covers that instance. We use manual inspection to validate if the behaviour/rules satisfy our notion of correctness and completeness, as detailed in the experiments section of this chapter. Generally, the rules extracted and representations of the behaviour produced are correct but incomplete. This is because the automation of the approach relies on machine learning, which discovers patterns and correlations based on certain statistical thresholds of occurrences of a given behaviour. However, completeness can be improved by incrementally observing the behaviour of the bots covering new behaviours and additional examples. In the following section, we present a quantitative evaluation of completeness and correctness.

Despite the difficulty of reverse engineering a complete behaviour of bots in absolute terms (including behaviour that is not observed), our approach reveals interesting information which aids our understanding of their behaviour and their role in campaigns. Case studies in Section 7 demonstrate this further.

6.1.1 Experimental

The experiments in this section are intended to evaluate the correctness and completeness of the proposed approach for rules extraction from Twitter bots. We consider different bots, including the prototype bots we created and real Twitter bots that actively engage with real users, to validate the results of the rule extraction and sentiment analysis.

For the first experiment, we created prototype Twitter bots. We allowed each bot to run for three weeks and then crawled its data, including all the available features. In the second experiment, we selected real Twitter bots that had been successful at attracting human users. These include bots that were found to have been retweeted by @realDonaldTrump, the verified Twitter account of Donald Trump. We used Debot[1], a system for the detection of Twitter bots to confirm our selection. For easy reference, we labelled the bots as

[1] http://www.cs.unm.edu/ chavoshi/debot/check_user.php

Bot1, Bot2 and Bot3. Bot1 (@Don_Vito_08[2]) is a Donald Trump-supporting bot. It was detected as a bot by Debot on 2nd November 2016. Bot2 (@natespuewell[3]) is a Clinton-supporting bot [8]. It was detected as a bot by Debot on 15th September 2016.

Bot3 (@BritainStays[4]) is an account for the campaign against Brexit. It is not detected by Debot but rather by our application which we built to catch automated account, and further to study the behaviour of active automated accounts. Many of the accounts that we have detected were later detected and suspended by Twitter, including @UKIPNFKN which we found to be connected with @BritainStays, reacting to and propagating similar hashtags. On analysis of BritainStays, we found evidence of automation in its dataset, see also Table 6.1.

Data Collection

In each experiment, we crawled a maximum of 3,200 tweets from each bot. These included retweets and replies. We are interested in understanding the users and tweets attractive to the bots. Hence, for each retweet or reply, we extracted the original tweets and the users who created them. We collected all the available features of the tweets and the users in a table of features. We observed the table to established an initial understanding of the behaviour of the bot. Through this observation, we learned that Bot1 is retweeting from and replying to a set of users, including Donald Trump and tweets with keywords and hashtags including, AmericaFirst, and makeAmericaGreat. We also found that Bot1 is replying to many tweets using identical text. This is another feature by which one can identify bots.

It was challenging, to establish a good understanding of the behaviour of Bot2 just by looking at its dataset, so we used our approach to analyse its behaviour and then go back to the data to validate our finding. This suggests that our approach is helpful for understanding the behaviour of sophisticated bots.

[2]https://twitter.com/Don_Vito_08
[3]https://twitter.com/natespuewell
[4]https://twitter.com/BritainStays

In the dataset of Bot3 (BritainStays), we found that its tweets contained the hashtags #stopBrexit and #brexitwontwork and links to Brexit news from independent.co.uk and theguardian.com. It also retweeted many of its tweets, possibly due to poor automation. It retweets tweets containing keywords or hashtags such as Brexit, Theresa May, StopBrexit. It made a few replies, containing well-structured text, Brexit Fact #102, 103, ..., and hashtag #FinalSay.

Rule Learning

We implemented the rule-learning method based on scikit-learn [86], an open-source machine learning tool supported by Google and INRIA. We treated our learning process as a binary classification problem, wherein for each case we either learn rules for one action (as opposed to any of the others), or between two actions. We use a grid search to cover a wide range of hyperparameters to find optimal choices. These include the maximal depth of a tree, class weight, minimal number of samples per split and leaf. We used a five-fold, ten-repeats stratified cross-validation. We used the roc_auc score as our scoring parameter. This optimizes rule learning to best separate the action regarded as the positive choice from others regarded as negative.

We use a decision tree classifier to build decision trees based on the results of the feature selection process [96]. Figure 6.1 shows a representation of the decision tree generated from Bot1 prior to rule extraction. The figure indicates the limited representation power of decision trees. Looking at the visualisation in Figure 6.2, our approach simplifies and generalises the decision tree using a simple but effective rule-based representation.

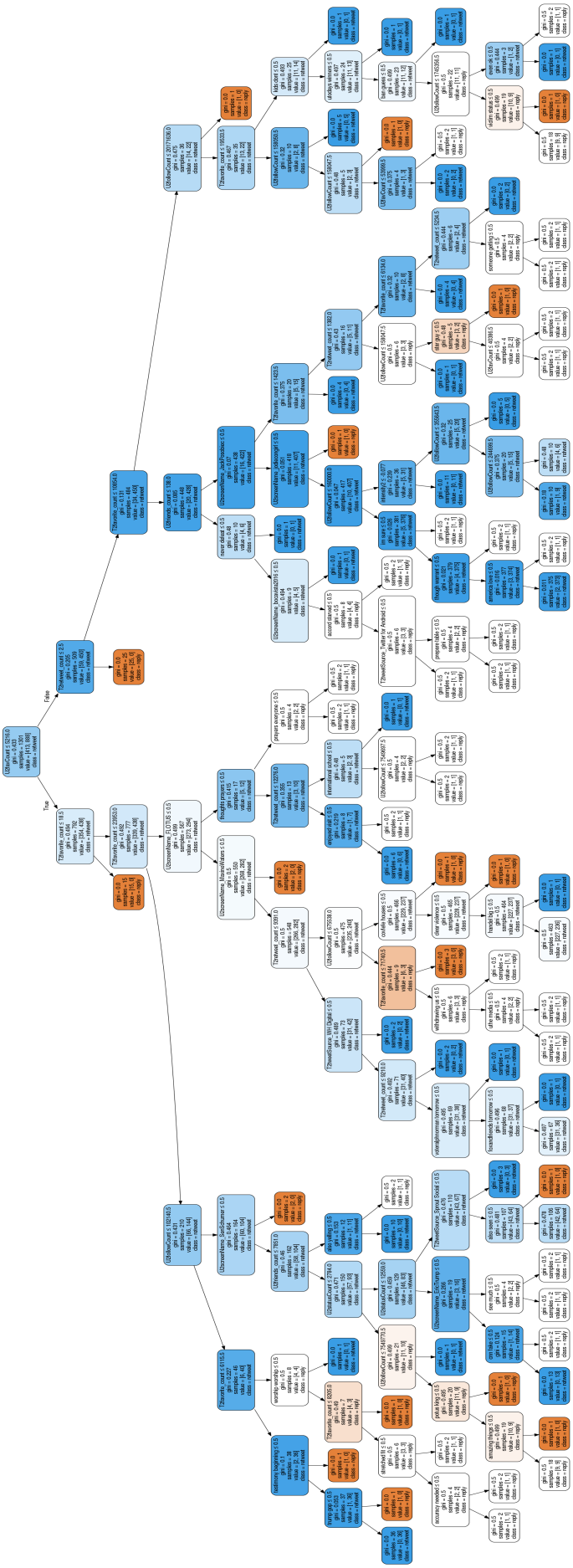


Figure 6.1: Decision tree generated from Bot1. Blue nodes represent decisions leading to retweet, orange nodes indicate replies.

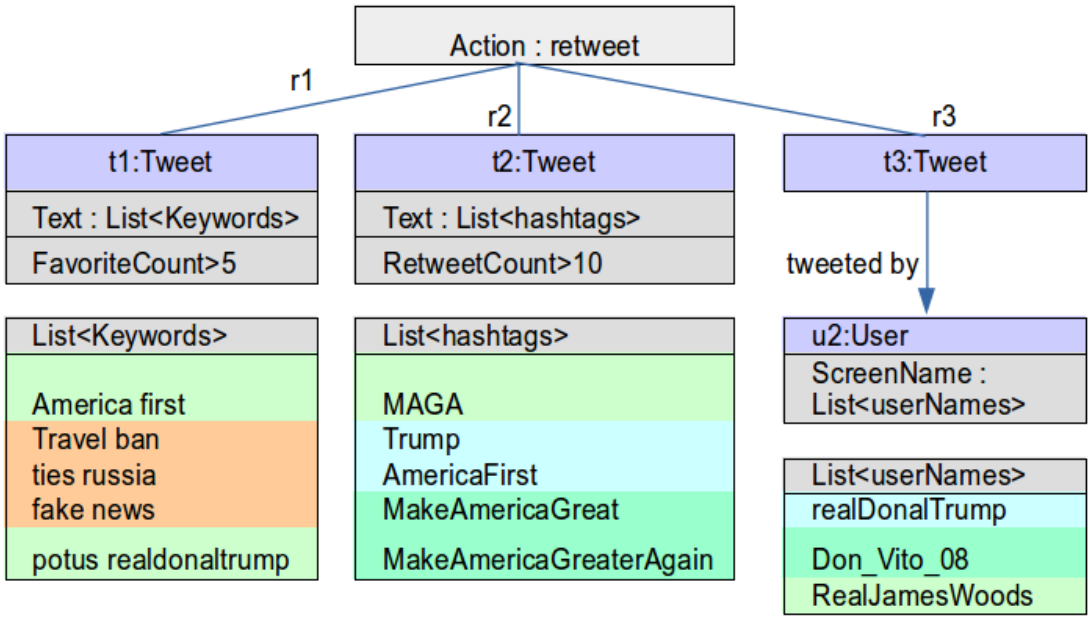


Figure 6.2: Retweet rules for Bot1 generated from the decision tree in Fig. 6.1.

Figure 6.2 visualises the retweeting behaviour of the bot. It covers unseen examples by separating structural features from attribute values. The representation includes lists of users, keywords or hashtags that are part of the rules, but were not captured in the sample data.

Rule 1 says that the bot retweets tweets containing a set of keywords with a favorite count of at least five. The keywords are listed in an attribute table which is colour-coded based on sentiment. Keywords such as Travel ban, Ties Russian are coloured orange because the tweets are negative while AmericaFirst is green because tweets associated with it have a positive sentiment.

Rule 2 states that the bot retweets tweets containing the particular hashtags that have been retweeted at least ten times.

Rule 3 specifies that the bot retweets tweets created by a set of users, including realDonaldTrump and the bot itself (Don_Vito_08). This is another feature identifying bots, which retweet many of their own tweets due to automation.

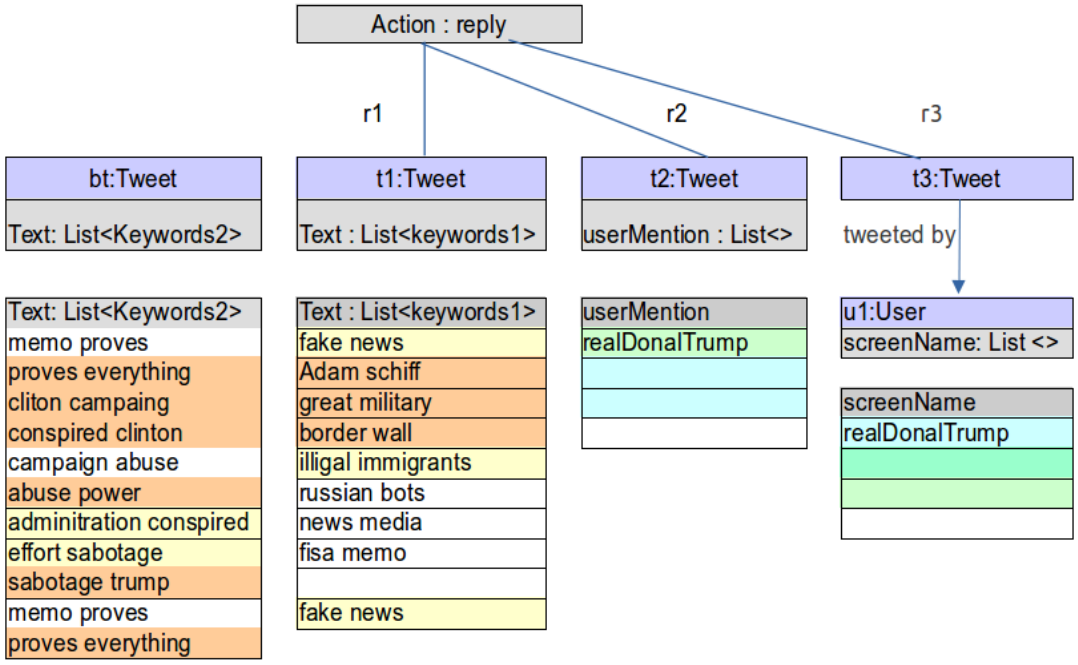


Figure 6.3: Reply rules for Bot1.

Fig. 6.3 represents the reply behaviour of Bot1 using our approach. There are three patterns which trigger the bot's reply. First, tweets that contain specific keywords; second, tweets with user mentions for realDonaldTrump; and third, tweets from certain users. The table below each rule provides details about the attribute values associated with the rule. The separation of rules from attribute values simplifies and generalises the presentation.

6.1.2 Qualitative

To assess the quality of the rules produced by the proposed approach, we assess the extent to which the results produced match our observations of the actual behaviour of the real bots or the rules controlling our prototype bots. The prototype bots provide the first ground truth, as we are aware of the actual behaviour of these bots. We found that our approach has successfully recovered the rules. For the real bots, the behaviour observed via the dataset forms the ground truth. We found that our approach recovers details that match the observed behaviour. It also captures additional information that may be difficult to find through human observation. It can be seen that the retweet rule for Bot1 shown in

Figure 6.2 matches our initial observation that the bot retweets tweets that contain specific keywords or hashtags related to Donald Trump. In addition, it captures the relationship between the keyword and the tweet's favorite count, hashtags and the tweet's retweet count. Although we did not notice this during the observation, the bot uses this as a strategy to select the most relevant tweets.

6.1.3 Quantitative

To quantitatively evaluate the approach, we use standard machine learning techniques. We gather separate independent test data to check how well the model from which we generate the rules will perform on unseen data (new observations). For Bot1 and Bot2, the training data consists of 2,767 and 1,792 tweets, while the test data comprises 2,908 and 927 tweets, respectively. In the case of our prototype bots, we explore different cases, such as limited data and class imbalance. Our intention is that the approach should capture patterns of behaviour within smaller representative examples. We varied the size of training versus test data, using positive and negative examples to check the quality of the decision trees generated. The two prototype bots have training date sizes of 604 and 342 tweets, and test data sizes of 339 and 388. The prototype bots are called Helpdeskx and CSInfo_news. The results are presented in Table 6.1.

Table 6.1: The performance results of the model on unseen data

Data set	***Accuracy***	***Precision***	***Recall***	***F-Measure***	***AUC_Score***
Helpdeskx	0.95	0.95	0.95	0.95	0.95
CSInfo_news	0.86	0.87	0.86	0.85	0.82
Don_Vito_08	0.80	0.83	0.78	0.80	0.75
natespuewell	0.90	0.91	0.90	0.88	0.70
BritainStays	0.98	0.98	0.98	0.98	0.98

The roc_auc score is one of the most popular measures used to evaluate robustness of a classifier. A roc_auc score of 0.5 is considered to be weak, while 1.0 is excellent. In predictive studies of psychology, law and human behaviour,a score of 0.70 or higher

is considered strong [94]. In that sense, the results reported in Table 6.1 show that the models from which we generated the rules are good enough, the lowest roc_auc score of natespuewell being 0.70. We noticed that some of the topics in the training data are not in the test data, but the overall behaviour does not change. This is the advantage of our rule representation. The rules will remain the same; only the tabular values will change. We achieved roc_auc scores of 0.95 and 0.98 in the Helpdesk and BritainStays datasets because of the strong pattern of the test data in the training data. This shows the strong predictive power of the models.

It is important to note that bots behaviour is mainly event-specific. It may change over time due to shifts from one event to another. Therefore, unlike other machine learning problems, having more training data will not always lead to a good (or improved) predictive model. We use incremental learning to detect changes in behaviour to allow our approach to report different rules for different behaviours.

6.2 Performance and Scalability

As social bots are increasingly becoming more sophisticated, we explore the performance of the approach across a range of different Twitter accounts. A Twitter account can be controlled by simple automation, such as retweeting from a set of users or tweets with a set of hashtags. It can be complex, taking actions according to complex rules and analysing various features. In the case of organizational or campaign accounts, the account can be managed by single or multiple users. We conducted several experiments with different groups of Twitter accounts to explore the performance of the approach. We asked the following research questions:

- What is the performance (completeness and correctness) with regard to different Twitter accounts? e.g., Bots, humans, simple and complex accounts.
- How well does the approach scales with a large number of observations (traces)?

Specifically, , we asked what the effect of an increase in the number of tweets on the accuracy score and the number of rules produced?

- What is the time cost of each step; data pre-processing, building the tree model, rule extraction and rule visualisation as the number of tweets increases?

6.2.1 Experimental Setup

To explore the performance of the approach across a different range of Twitter accounts, we experiment with a different set of accounts. We select 500 bots and 500 humans accounts from a publicly available labelled dataset [51]. We called this Dateset1, which is from Ginali *et al.* This dataset was grouped into different classes (based on the number of followers), including celebrity level accounts. Celebrity level accounts are popular users (humans) such as BarackObama, KimKardashian, Cristiano and globally renowned agents (bots) such as CNN, BBCWorld, and Reuters. This allows us to explore the performance of the approach on the different classes. We select a new set of 500 bots and 500 human accounts from the Brexit campaign dataset (see Section 7.2 for details). We called this Dataset2. This dataset was classified into bot and human accounts using botometer [5] [30], a publicly available service to check the likelihood of an account being a bot. The purpose of using this dataset is to check if there is any correlation between the performance (accuracy score) of our approach and the score provided by the service. Research [6, 8] shows that a botometer score of 50% is a valid basis on which to label an account as a bot. We classify accounts with a score higher than or equal 50% as bots and accounts with a score below 30% as humans. We consider a gap of 20% between the score of a bot and a human account to reduce the chances of having bots in the set of human accounts. To explore the performance of the approach on ordinary user accounts, we collect an additional 200 accounts (100 humans), which we call Dataset3. We manually screened these accounts. The human user accounts are users who interact with Twitter handles of organisations such as universities and have correspondence in terms

[5] https://botometer.iuni.iu.edu/!/

Table 6.2: Summary of datasets utilised in the study.

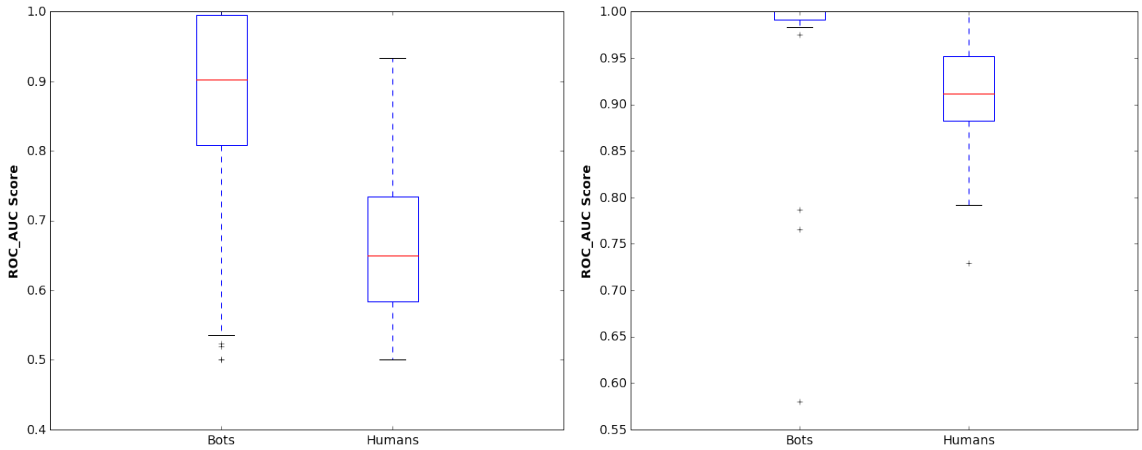
Category	***Accounts***	
	Bot	***Human***
Dataset1	500	500
Dataset2	500	500
Dataset3	100	100

of replies. The prototype tool collects a maximum of 600 tweets, traces of activity from each account, implements the rule learning approach and records the accuracy, roc_auc score, recall, precision, time to process the data, time to build a decision tree model, time for rule extraction and time for rule visualisation.

6.2.2 Results and Discussion

Performance

Figure 6.4 shows the performance of our rule learning approach on ordinary user accounts and celebrity level accounts using a box and whisker plot. The box plot provides a visual summary of results, where the lower line of the box represents the first quartile, the red horizontal line that goes through the box represents the median, and the upper end of the box represents the third quartile. The whisker goes to the minimum from the first quartile and maximum at the third quartile.



(a) Ordinary user accounts (b) Celebrity level accounts

Figure 6.4: Performance of our approach for rule learning on bots and humans accounts

The results in Figure 6.4 show that the rule learning score of an ordinary human user is usually around 0.6 to 0.75, while that of a bot starts from 0.8. Looking at the celebrity level accounts, the score of the human account is around 0.85, and that of the bot mainly starts from 0.98. This shows that celebrity human accounts behave more like bots. Our analysis of results from the different accounts show that the performance (achieving higher or lower scores) is not well correlated with the number of followers; rather, it depends on the behaviour of the account. Rule learning achieves a higher score that is similar to that of bots for some human accounts. We examined some of the human accounts with higher scores and found that the majority of these behave in an orderly pattern, selective in terms of the users they interact with and whose contents they retweet. Their tweets mainly focus on specific topics. We also found that many of these are working-class people. Toward the end of these experiments, we begin to perceive that the rule learning could be used for Twitter user profiling, categorising users as chatty or professional and identifying their topics of interest. However, we cannot conclude this with any real certainty as this would require additional experiments. Figure 6.5, 6.6 and 6.7 show a detailed quantitative evaluation of completeness and correctness of the approach in terms of recall and precision. The results show that the approach performs reasonably for bot accounts across the three different datasets. It achieved a score (both recall and precision) greater than 0.78 on bot accounts. Looking at the scores from the perspective of human

accounts, the performance is somewhat variable depending on how regular the associated behaviour is.

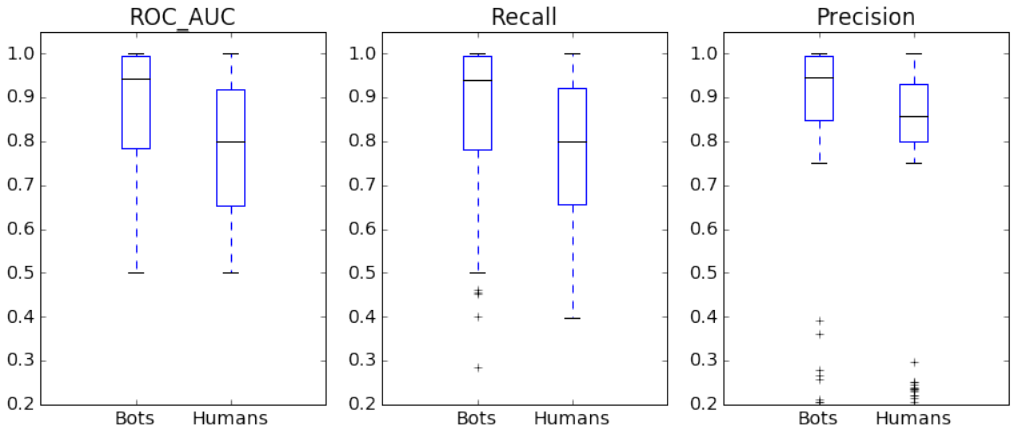


Figure 6.5: Dataset1 from Gilani *el al.* [50]

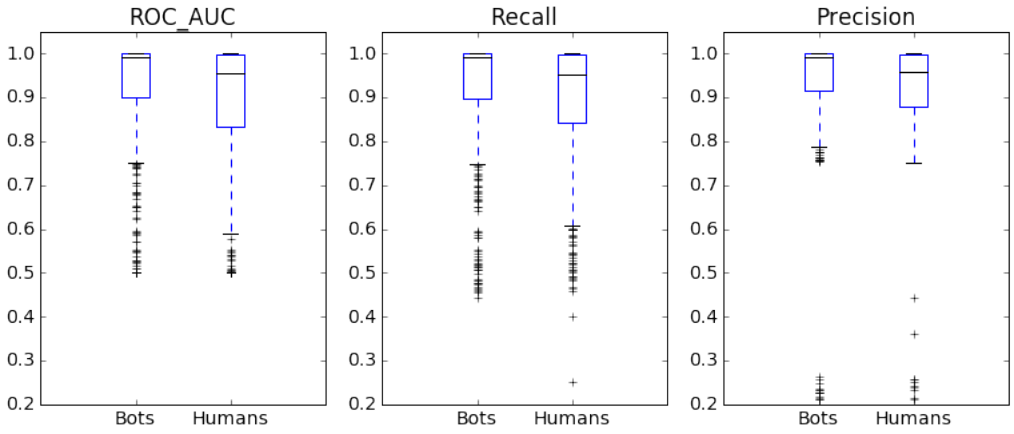


Figure 6.6: Dataset2 from Brexit study (cf. Section 7.2)

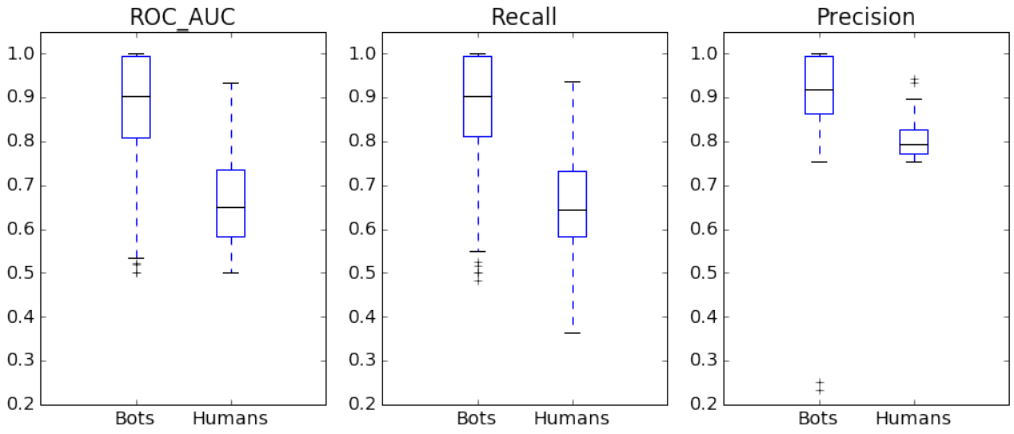


Figure 6.7: Dataset3, self manually checked data

One interesting point to note from Figures 6.5, 6.6, and 6.7 is the difference between the line of bots and humans in each of the three datasets. There is a clear gap between the line of bots and humans in Dataset1 (Figure 6.5) and Dataset2 (Figure 6.6), but the gap in Dataset3 (Figure 6.7) is less apparent. This could be because both human and bot accounts in Dataset3 engage in similar campaigns, while Dataset1 and Dataset2 contain accounts which have various interests. This revealed that some humans behave like bots during a given campaign period, but experimentation with a large number of accounts would be required to establish this claim.

Scalability

To understand how well the approach scales with a large number of observations, we collect a maximum of 3,200 tweets from the accounts and observe the changes in accuracy and time cost of each step. We start from 200 tweets and repeatedly increase the size by 200.

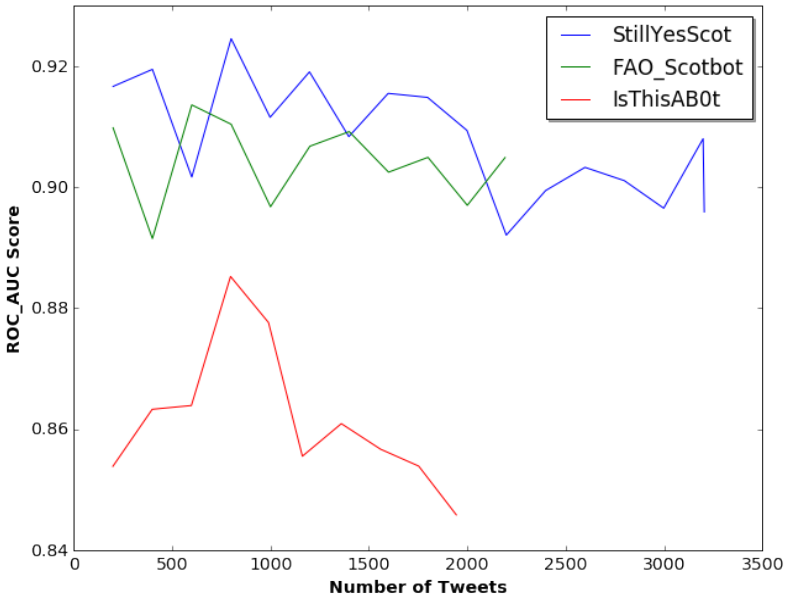


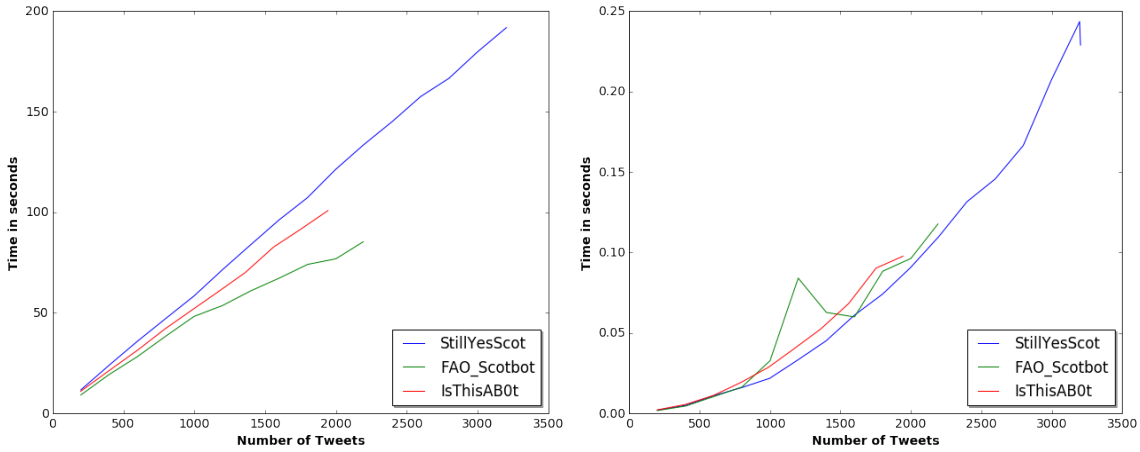
Figure 6.8: Relationship between number of tweets and accuracy

Figure 6.8 shows the results of the experiment on three bots, @StillYesScot, @FAO_Scotbot and @IsThisAB0t. The results show that an increase in the number of tweets does not always increase the accuracy. This is because of the behaviour in the previous observations

can be different from that in new observations. This change in accuracy score allows our approach to detect changes in the behaviour of a bot and report different rules for different observations. The accuracy curve of @StillYesScot is higher than that of @FAO_Scotbot and @IsThisAB0t because its pattern of behaviour is more deterministic.

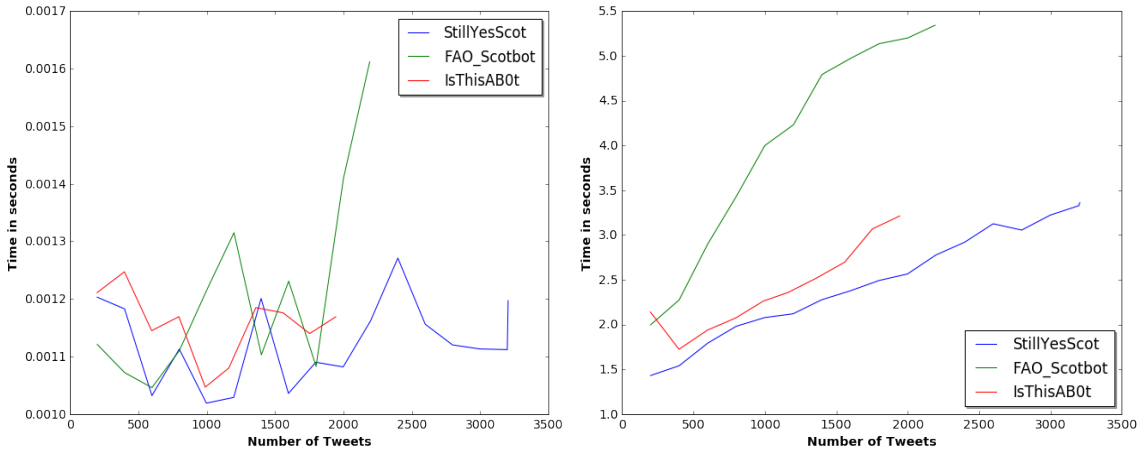
Time cost of the steps for rule learning

We investigate the time cost of each step in the rule learning process to understand its cost as the number of tweets increases. The results of our experiments with more than 1000 bot accounts shows that the time for the rule extraction approach does not increase proportionally with increase in the number of tweets, though the time for the rule visualisation increases as the number of tweets increases. This is due to the need for the visualisation approach to revisit the dataset to determine and encode the sentiment of the rule instances. Figure 6.9 shows the time cost of each step in the rule learning process as the number of tweets increases.



(a) Time for data preprocessing as the number of time increases

(b) Time for building a decision tree model as the number of time increases



(c) Time for rule extraction as the number of tweets increases

(d) Time for rule visualisation as the number of tweets increases

Figure 6.9: The time cost of each step in the rule learning process as the number of tweets increases

Figure 6.9a and Figure 6.9b show that the time for the data pre-processing and the time to build the decision tree model increase with the number of tweets. For 2,000 tweets, the average time pre-processing is about 1 minute 50 seconds and about 0.11 seconds to build the model. The time for the rule visualisation increases as the number of tweets increases, as shown in Figure 6.9d, but the time for the rule extraction does not increase with increasing in numbers of tweets, as shown in Figure 6.9c. From the Figure 6.9c, we may note that the rule extraction time for @FAO_Scotbot increases from 0.0011 to 0.00165 seconds when the number of tweets reaches 2,000 because of the increase in the size of the tree produce by the model. We found that the time for the rule extraction

ultimately depends on the size of the tree produced by the learning model. In Figure 6.9c, we note that with 2,200 tweets, the longest time for rule extraction is 0.00165, and 5.4 seconds for the rule visualisation all from @FAO_Scotbot. Our experiment with more than 1000 bots shows that for 600 tweets, the average time for the rule extraction is 0.0015 seconds and 2 seconds for the rule visualisation. Generally, it is well known in the area of data mining that data pre-processing takes some considerable time . However, the is evidence that our rule extraction approach is not affected by the size of the data, and that the visualisation takes few seconds; this provides us with a certain degree of confidence that our approach will scale on moderately sized number of bots with reasonable numbers of tweets.

6.3 Threats to Validity

Our evaluation is empirical in nature; however, empirical work is subjected to two different types of threats to validity, namely threats to internal validity and to external validity [16, 46, 112]. Threats to internal validity are factors that affect the sufficiency of the evidence to support the claim [42]. Threats to external validity are factors that affect the generalisation of the results [39, 112]. We discuss how these threats apply to our evaluation and how we address or mitigate them as follows.

Threats to internal validity

Instrument: The fact that the experiment was conducted using our tool could be a threat to the validity of our results in terms of correctness and completeness. However, the tool is built on a stable, popular machine learning framework, scikit-learn [86], which is used by many researchers in the area of bot analysis and detection [30,32,51]. We have also used a manual inspection to check the correctness of results produced by the tool. Another threat which applies to the performance results (time cost of steps in the rule learning process) is the simulation environment (computer). It is important to note that the experiments were

conducted on a personal computer, not a high-performance computer (HPC). The system runs on ubuntu 16.04 LTS, memory 8GB and processor Intel Core™ i5-4690S CPU @ 3.20GHz × 4. However, the time cost could be higher or lower if a different configuration is used. Hence, the results applied only to our implementation.

Selection: The accounts selected from the Gilani *et al.* [50] dataset were classified into human and bot in 2017. The behaviour of the accounts can change. It is highly likely that some of the human accounts are now acting like bots or that humans actually use some of the bot accounts. To mitigate this threat, we use recent data from the Brexit campaign (Dataset2). However, the classification of the accounts into bot and human was done using another tool (Botometer) which checks only 300 tweets from an account to classify as bot or human could be a concern. Botomoter has been tested and used by many researchers [5,6,8,30]. We crosscheck its results via a manual check of a sample of 200 accounts (including 100 accounts that we genuinely know are managed by humans).

Threats to external validity

The assumption that complex accounts are old accounts, with a large number of followers and tweets, may not always hold. Complex accounts can have few followers. Hence, detailed experimentation with complex accounts could not be performed. However, our experiments with a large number of accounts, including those of humans, could be considered as evidence of testing the approach on different types of accounts, and which indeed may include complex accounts. The results presented in Figures 6.5, 6.6, and 6.7 have a total of 1,100 bots and 1,100 human accounts. The results show that there is some difference between the performance of human and bot. However, we have no evidence to say this sample is sufficient to make any proper generalisation. Therefore, it is worth noting that the study does not claim that the approach can be used to differentiate between bot and human accounts, nor claim of an exhaustive test of complex accounts. The experiment presented purely shows the performance of the approach on different accounts and its scalability as the number of tweets increases according to our implementation.

6.4 Summary

In this chapter, we empirically evaluated the proposed approach. We created prototype bots and uses real Twitter bots to evaluate the correctness and completeness of the approach, both qualitatively and quantitatively. Results from the experiments show that the approach is practically applicable to real bots and succeeded in describing the behaviours of the bots correctly. Generally, the rules and representations of the behaviour produced by the approach are correct but incomplete because the automation of the approach relies on machine learning which discovers patterns and correlations based on a certain statistical level of occurrences of a behaviour. However, the completeness can be improved by incrementally observing the behaviour of the bots covering new behaviours and additional examples.

We have assessed the performance of the approach on different types of Twitter accounts, experimenting with more than 2000 accounts of different level of complexity and found that the rule learning achieved higher accuracy even in some human accounts with regular behaviour. However, the approach depends on the existence of patterns to achieve much higher accuracy. We have assessed how well the rule learning approach scales with a large number of observations by conducting experiments to check the effects of an increase in the number of tweets on accuracy and the number of rules produced, the results of which show that an increase in the number of tweets does not always lead to an increase in accuracy. This is because the behaviour in some previous observations can be different from that in newer observations. This change in accuracy score allows the detection of changes in the behaviour of a bot and reports different rules for different observations. We have assessed the time cost associated with each step in the rule learning and show that the scalability can be accepted for batch processing a moderately sized number of bots.

Overall, the evaluation provides a certain degree of confidence about the validity and scalability of the approach. In the following chapter, we present a prototype implementation of the approach and two case studies to demonstrate its application..

Chapter 7

Tool support and Applications

This chapter demonstrates the application and usability of the proposed approach. In section 7.2, we use the approach to study the behaviour of bots in debates after the 2016 UK referendum to exit from the European Union. Brexit has been one of the most controversial issues in the political history of the United Kingdom. We show how to identify strongly opinionated accounts from a large set of accounts participating in the same campaign and study their bias and policy position on other topics. We identify bots and show how we use the reverse engineering approach to uncover their implemented strategies. This reveals information useful for understanding their role in political debates. In Section 7.3, we present a case study of the 2019 Nigerian presidential election where we use the reverse engineering approach to study the online campaign strategies of the major political parties. This shows that the proposed approach could be used to study the behaviour of any social agents (automated or non-automated). Furthermore, we use the proposed differential sentiment analysis to study the relationship between Nigerian online users' opinions and information disseminated by the 12 top Nigerian newspapers.

7.1 Prototype Tool

As proof of concept, we create a prototype implementation of the approach. This assists us in the evaluation of the application and gives an indication of the usability of the approach. In this section, we describe the services provided by the tool, usage scenario and the architecture of the tool.

7.1.1 Services

The tool offers four different services to help with the analysis of Twitter accounts:

1. **Collection of Traces:** The tool allows collection of traces in two forms, based on account screen-names and on campaign keywords and hashtags. Given a Twitter account name, the crawler collects traces of activity up to the Twitter API limits, a maximum of 3,200 tweets from the account. For an activity such as retweets, replies, favorites, e.tc., it collects features of the original tweets and their authors. It also collects traces of tweets and account features of friends and followers of the given account.

2. **Rule extraction and Visualisation:** Given a bot account, the tool collects traces from the account as described above, pre-processes the data as described in Section 3.3.2, builds a decision tree model and extracts rules as described in Section 4.2. It produces a visualisation of the rules, an example shown in Figure 7.9. The interface for the rule learning takes screen-name of an account with an optional start and end date. This is to allow incremental learning and extraction of rules for specific date ranges.

3. **Differential Sentiment Analysis:** Given a bot account, the tool extract traces from the account, friends of the account, and followers of the account. It calculates the bias and antagonism of the bot as described in Section 5.3 and visualises the results as in the example shown in Figure 7.12a and 7.12b. As an input, it takes account'

screen-name and three optional parameters - *topic list, number of topics (n), start, and end dates* to calculate the bias and antagonism between specific dates. The tool can automatically find n-topics of interest of a bot and displays the bias and antagonism on those topics or accept a list of topics from the user via the optional parameter (topic list). The tool also computes and visualises the differential sentiment over time, as described in Section 5.4 and exemplify in Figure 7.16.

4. **Strongly Opinionated Accounts:** The tool provides an interface for the study of opinionated accounts. Given a set of campaign topics, keywords or hashtags. The tool collects tweets from the Twitter API periodically based on the given search terms, computes (cf. Section 5.2) and visualises the strongly opinionated accounts. This interface assists in providing interesting accounts that we may need to study their rules, bias or antagonism to understand their role in a campaign. Examples of outputs from this interface are shown in Figure 7.6, 7.8 and 7.14. Next is a usage scenario of the tool.

7.1.2 Tool Usage Scenario

Suppose we want to study the behaviour of Twitter accounts in a campaign; knowing the popular campaign keywords or hashtags, we can use the Twitter search section of the tool to collect tweets and visualise strongly opinionated accounts. We have shown a result of the opinionated accounts in Figure 7.9; here, will focus on the behaviour/rule learning and antagonism of an account. Suppose that AUOBSCOT is the account that we want to understand its behaviour and antagonism. Figure 7.1 shows a screenshot of the rule learning interface with a rule which represents the retweet behaviour of AUOBSCOT. The family of green colours in the rule represents positive tweets. The rule shows that AUOBSCOT is retweeting positive tweets which contain specific hashtags and are from specific set of users. From the rule we can learn:

- i) How the account conducts its campaign, i.e., promotes positive tweets with hashtags-

#AUOBGalashiels, #AUOB, #AUOBGlasgow, #Indyref2, #DissolveTheUnion, #AllUderOneBanner, and #AUOBOban.

- ii) Main URLs of its contents.
- iii) Its social masters, i.e., AUOBSCOT, AUOB_Kernow, AUOBCymru, liveIndyScot, ands Neil_MacKay5.

If we analyse the information provided by the rule, we would notice that the screen-names of three users (AUOBGalashiels, AUOB, AUOBGlasgow) which the account retweets appear to be related, which means they are accounts for the same campaign; they may have the same real master. Looking at the screen-name @Neil_MacKay5, we found that he is the founder of the AUOB campaign being promoted by the accounts. This could imply that @Neil_MacKay5 is the real master of the accounts and possibly created all the other related accounts to promote the AUOB campaign.

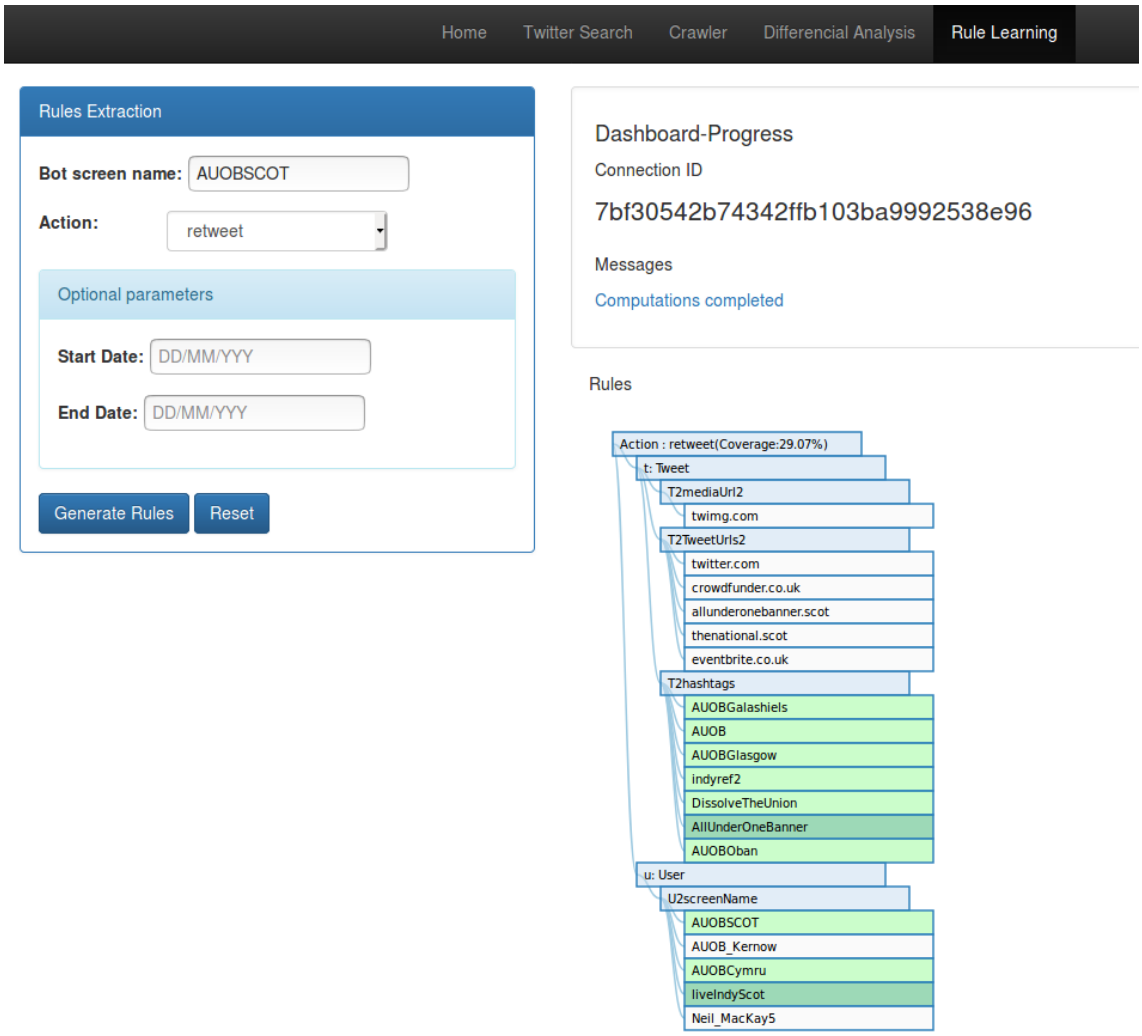


Figure 7.1: Rule learning interface

Figure 7.2: Twitter profile of @Neil_MacKay5, described as the AUOB founder

In Figure 7.2, the account @Neil_MacKay5 describes its profile as a musician, writer and AUOB founder. This allows us to conclude that @Neil_MacKay5 is the real master of the other three AUOB campaign accounts (AUOBGalashiels, AUOB, AUOBGlasgow). The subsequent section describes the antagonism of the AUOBSCOT account.

Antagonism of AUOBSCOT

Figure 7.3 shows the tool's interface for differential sentiment analysis with the results showing the relationship between the opinion of AUOBSCOT and that of its followers.

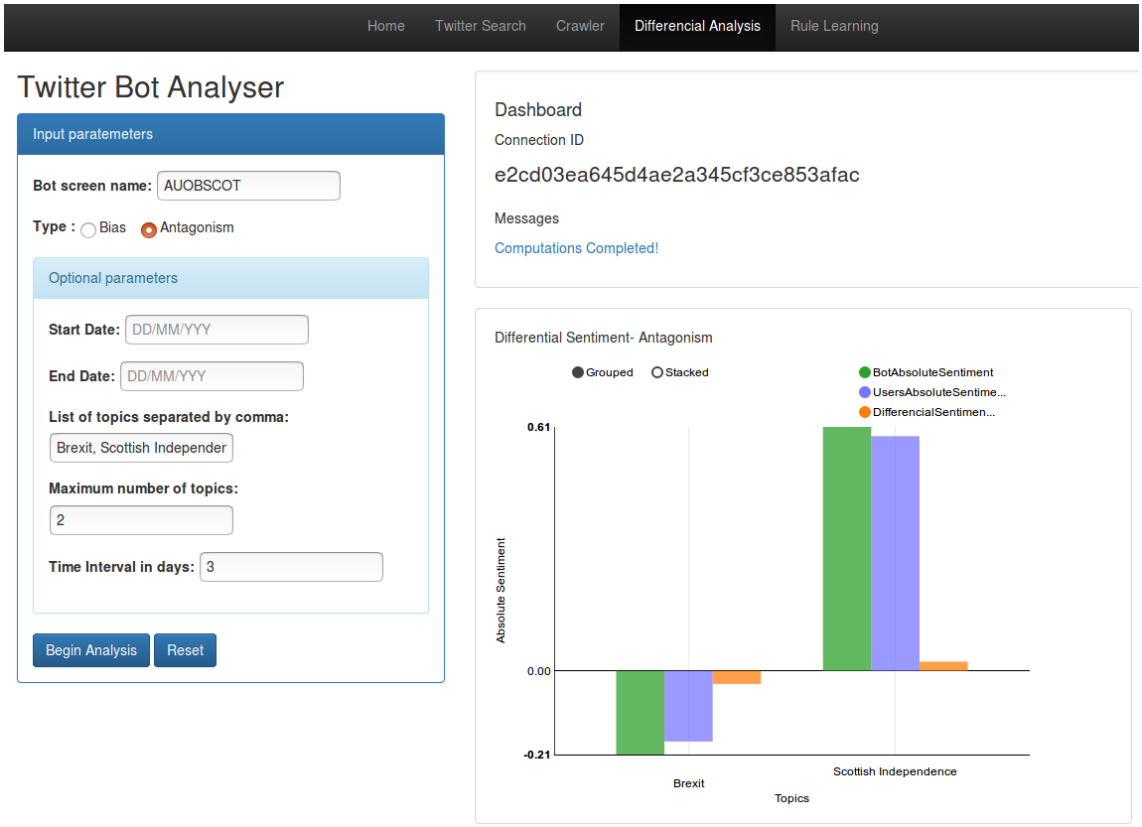


Figure 7.3: Differential sentiment analysis interface showing the antagonism of AUOBSCOT

In Figure 7.3, the tool computes the antagonism of the AUOBSCOT account. The result shows that AUOBSCOT and its followers have the same opinion, in that it opposes Brexit and promotes Scottish independence. The opinion of AUOBSCOT is in the green colour and that of its followers in purple. The orange colour represents the rate of antagonism between the two.

Figure 7.4 shows an example of how the tool visualises differential sentiment over time.

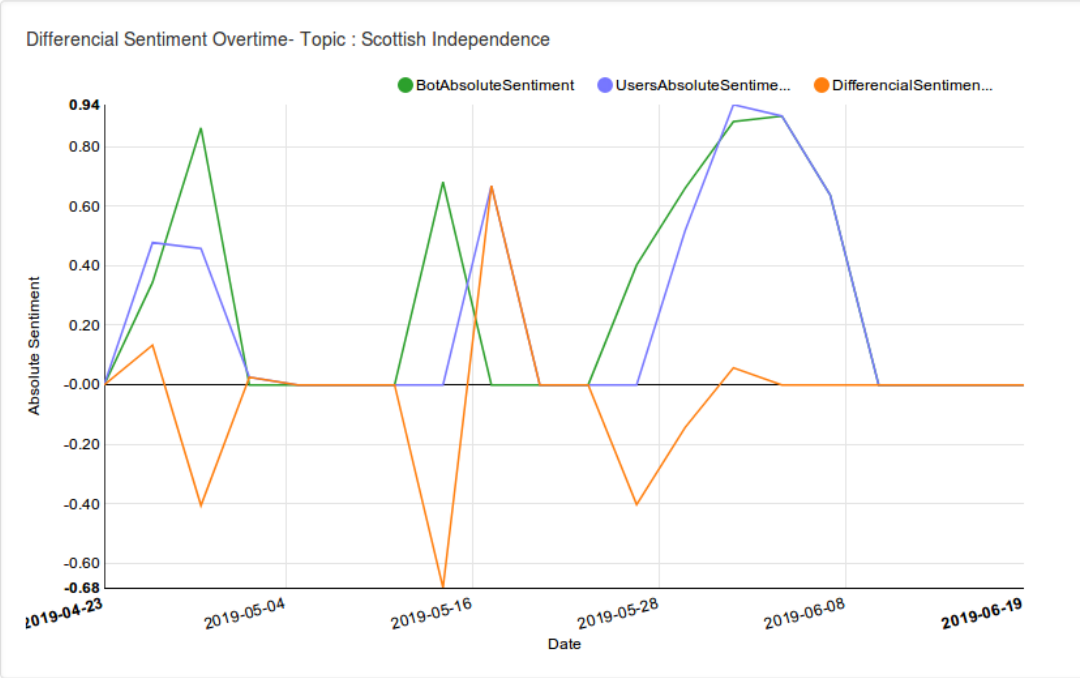


Figure 7.4: Antagonism of AUOBSCOT over time

Figure 7.4 shows how the opinion of AUOBSCOT and that of its followers changes over time. The orange-coloured line represents the rate of antagonism between the two. This helps to find periods where an account could have a significant impact on its followers' opinions. In the case of AUOBSCOT, based on results in Figure 7.4, these periods are 29-04-2019, 16-05-2019, 19-05-2019, and 28-05-2019. The following section describes the tool's architecture.

7.1.3 Architecture of the Tool

Figure 7.5 is a component diagram of the tool. The tool is designed in the form of loosely connected components to deal with the challenges of modern software systems, required to serve the same or different purposes in different ways and in a different format. The four major components, bsCrawler, RulesExtractor, DiffSentiment and Visualizer, interact with each other and provide the interfaces for users to perform various operations. The tool uses the Twitter API to access public tweets data. It uses a Natural Language Process-

ing Tool Kit (NLTK)[1] for data preprocessing and scikit-learn[2] for machine learning. The bsCrawler, function as a tracer which collects traces of bots' actions and tweet data from users which the bots interact with (followers and friends of the bots). The *Rules Extractor* implements the rule extraction process described in Section 4.2, and *DiffSentiment* implements the approach described in Section 5.3. The *Visualizer* provides interactive visualisation of the rules as described in Section 4.3, bots' opinions and behaviours described in Chapter 5 (See Section 7.2 for examples of the visualisations).

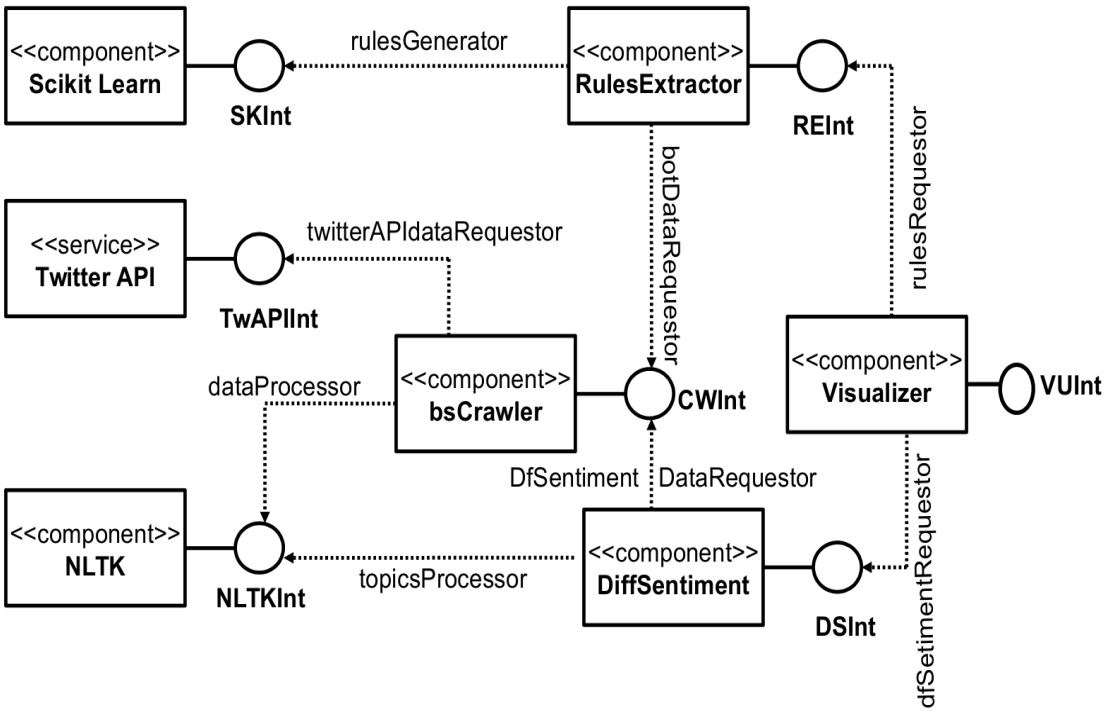


Figure 7.5: Component diagram of the tool

The tool is implemented as a web application in python and uses modern front end tools (HTML5 and JQuery). Celery, Redis, and Socket-IO are used for chainable task queues, and asynchronous/real-time event-based communication between the tool and connected clients. It uses D3.js[3], a JavaScript library for dynamic data visualisation in web browsers to provide interactive visualisations of the results. The following section presents a case study which demonstrates a particular application of the tool.

[1] https://www.nltk.org/
[2] https://scikit-learn.org/stable/
[3] https://d3js.org/

7.2 Case Study I: Analysing the Behaviour of Twitter Bots in Post-Brexit Politics

In the 2016 UK referendum on EU membership, bots were used by both parties, supporters of StrongerIn, and Brexit [64]. Bots were also used in the 2016 US presidential election [8]. Identifying their targets and strategies is a challenging task but can lead to a better understating of their use in political campaigns.

In this research, we study the behaviour of bots in the Brexit debates after the 2016 referendum, including issues such as a second Brexit referendum and Scottish independence.

We collected our data over a set of Brexit-related hashtags and searched for bot accounts by selecting strongly opinionated accounts, then analysed their strategies, and antagonism to understand their intended influence and policy position on Brexit issues. We found that there are more than 1,962 bot accounts currently engaged in the Brexit debate. Using the reverse engineering approach, we show how to uncover the bots' implemented strategies. We found that @StillYesScot, @IsThisAB0t and @FAO_Scotbot, all bots that promote Scottish independence, use similar strategies. Details of the analyses are as follows.

7.2.1 Research Questions

We aim to answer the following questions regarding the use of bots after the UK vote to leave the EU. They are prototypical of various political debates and can serve as a template for the use of our approach in their analyses.

- Are there bots promoting post-Brexit referendum issues? If yes, what are their strategies? In order to understand how bots carry out their campaigns and to infer their potential effects, we use our reverse engineering approach to study the strategies used by the bots to propagate information.

- What are their policy positions on specific issues, in addition to what they are generally known for? Here we investigate whether bots engage in discussing issues beyond their core purpose and discover their respective positions. This is important to understand their scope and to detect attempted cross-fertilization between topics.

- Who are the bots attempting to influence, and how? While it is difficult to evidence the actual impact of bots on their target audience (e.g., due to external factors such as news and political events), we analyse their antagonism to understand their intended influence and the possible impact.

- Do bots focus their messages on users from specific regions? We use the geolocation of bots' target audiences to gain insight into regional factors.

7.2.2 Data collection

We manually set a list of hashtags related to Brexit and Scottish independence. We include the twitter handles of the leaders of both the Conservative and the Labour parties. To form a comprehensive list, we first crawled data for three days, analysed the most frequent hashtags used by online users and then updated our list of search terms. The search terms used for the data collection are *brexit, brexit Second referendum, #NoDealBrexit, Brexit deal, Cancel Brexit, No Deal Brexit, #scottishindependence, Indyref2, scotref, scottish independence, #euref, @theSNP, #Brexit, #RoadtoBrexit, @NicolaSturgeon, @theresa_may, @Conservatives, @jeremycorbyn, @UKLabour*. We obtained a total of 2,520,663 tweets from 143,332 distinct users by querying the Twitter search API from 4 March to 9 May 2019. We chose to use the Twitter search API to ensure that we obtained all tweets related to the search terms rather than a sample of unfiltered tweets provided in real-time by the streaming API. This is to avoid issues reported in [81] about collecting data using the Twitter Stream API.

7.2.3 Bot detection

We used the Python API of Botometer[4] to check for accounts that are likely to be bots. Botometer [30] is a service available to check the likelihood of an account being a bot. The Botometer API uses the Twitter API to extract the top 300 tweets from a given account, analysing its content, temporal and network features to produce a bot score. Since the Botometer API incurs the limitations imposed by the Twitter API[5], it is difficult to test all user accounts. Instead, we obtained a bot score for the top 7,000 most active Twitter accounts ranked by volume of tweets in our dataset. A score of 50% has been shown to be sufficient to label an account as a bot [6, 8]. In this way, we detected a total of 1,962 as potential bot accounts out of the 7,000 accounts.

Identification of strongly opinionated accounts

We expect that strongly opinionated accounts would produce a high volume of tweets with a strong opinion. First, we use topic analysis to find the top five topics in the dataset, then analyse the sentiment intensity of the accounts over those topics. Figure 7.6 is a visualisation of these accounts based on their absolute sentiments and the volume of tweets produced. Our idea of using absolute sentiment (the sum of negative and positive sentiments) pushes all the less opinionated accounts on to the zero line while those with high polarity stand out. Since the majority of the accounts tend to be less opinionated, the further analysis concentrates on the top 100 accounts.

[4] https://botometer.iuni.iu.edu/!/api
[5] https://developer.twitter.com/en/docs/basics/rate-limits

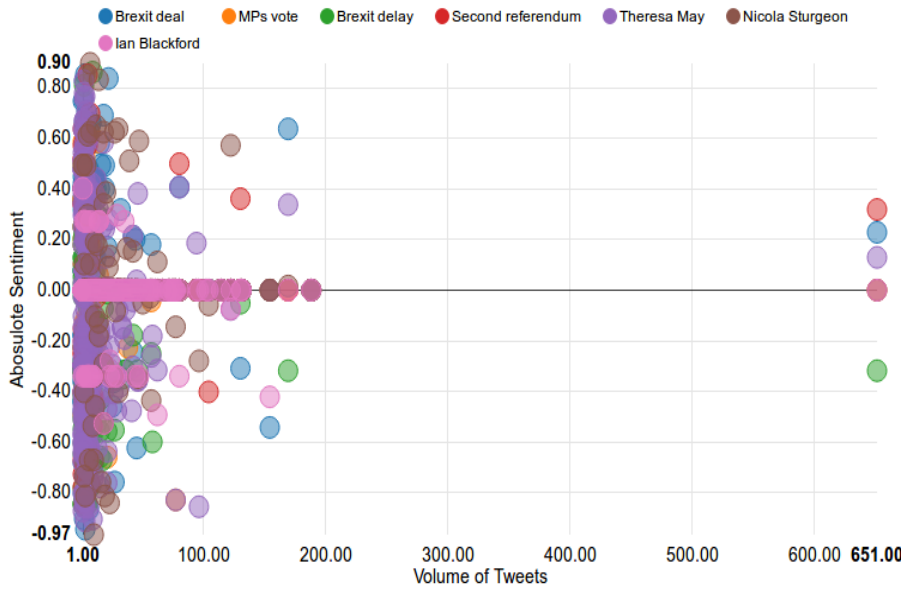


Figure 7.6: Opinion polarization of the accounts

Then, we investigate the relationship between accounts that promote opinions in favour of or against a Brexit deal with those that are calling for a second referendum. Figure 7.7 is a visualisation of these accounts.

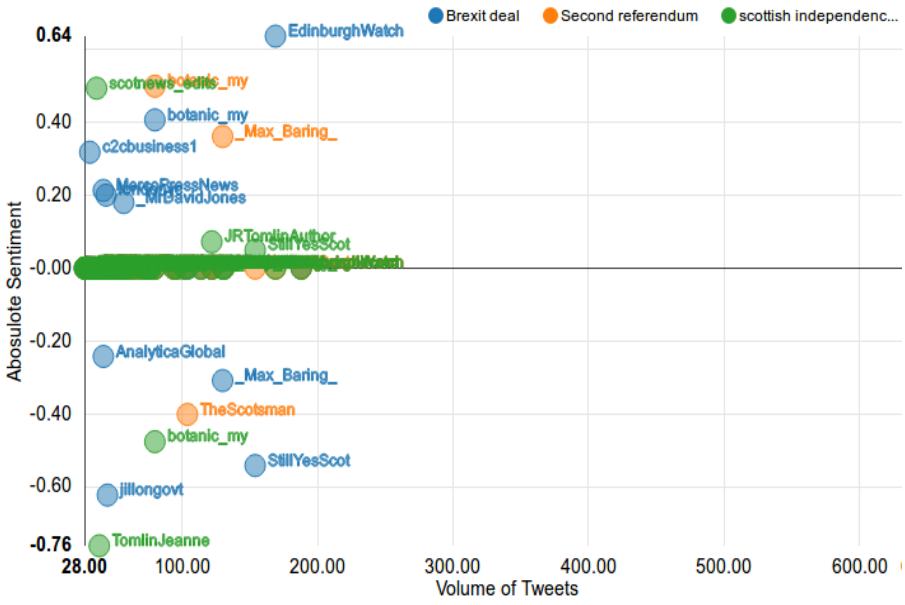


Figure 7.7: Opinion polarisation of top 100 accounts on the Brexit deal, Second referendum and Scottish independence

In Figure 7.7, we notice that @StillYesScot opposes Brexit deal while promoting Scottish independence. @EdinburghWatch supports a Brexit deal and @AnalyticaGlobal tweets

against it. @_Max_Baring_ is featured as an account that is against a Brexit deal and supports a second referendum. We can also note that, while @botanic_my promotes a Brexit deal and a second referendum, it is against Scottish independence. Figure 7.8 below shows opinions of the accounts on Scottish independence only.

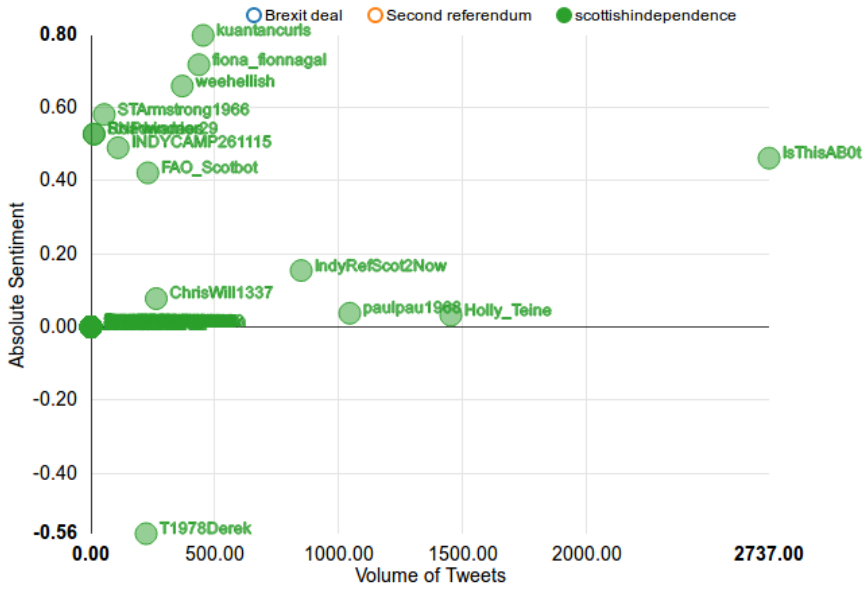


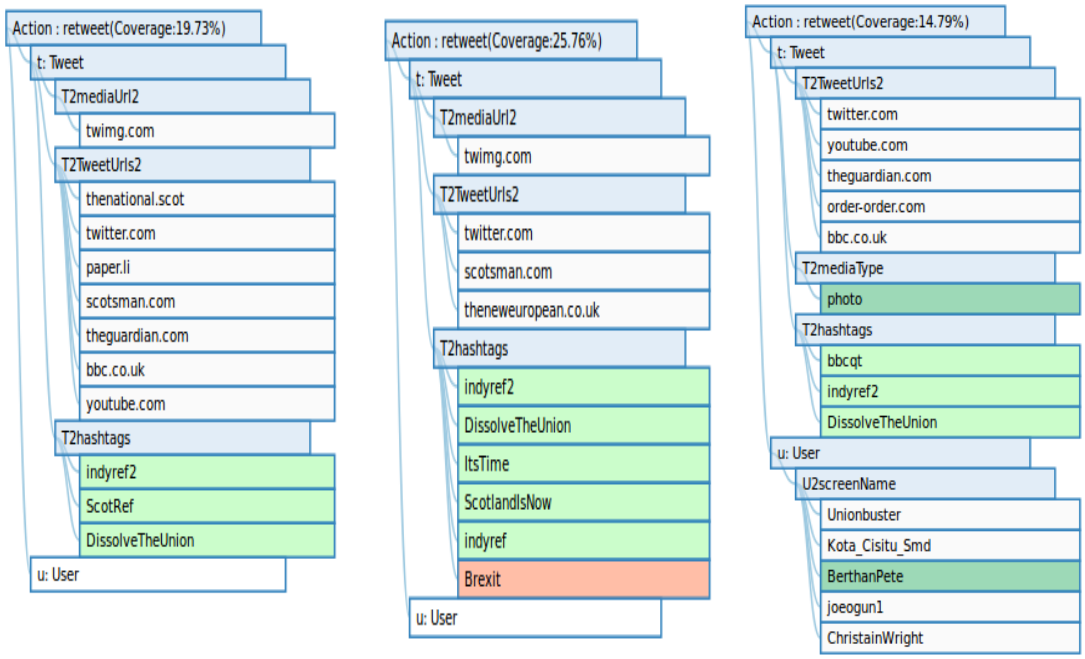
Figure 7.8: Opinion polarization of the top 100 accounts on Scottish independence only

Visualisation of the top 100 accounts in Figure 7.8 helps to identify interesting accounts such as @IsThisAB0t and FAO_Scotbot for further analysis. The next section shows detail campaign strategies of the accounts.

7.2.4 Implemented Strategies

Since the recent success in the detection of bots [22, 30, 101], the most challenging task is understanding their strategies, targets and antagonism [41, 64]. We contribute to this discussion by delivering the reverse engineering approach, which provides details about the behaviour of the bots. Figure 7.9a shows the result of applying our learning approach on @StillYesScot, a bot which supports Scottish independence. Apart from revealing the underlying construct of the bot's behaviour, it provides details of how it conducts its campaign. It shows that the account promotes Scottish independence by retweeting

positive tweets which contain Scottish independence hashtags (#indref2 and #ScotRef). The result also shows that tweets propagated by the bots contain photos and URLs of the Guardian and Scotsman newspaper. Analysing URLs can help us assess the credibility of the news an account is propagating based on its sources.



(a) @StillYesScot (b) @IsThisAB0t (c) @FAO_Scotbot

Figure 7.9: Retweets strategy

Our reverse engineering approach can help to identify bots with similar behaviour, which can lead to the identification of their masters. Using the reverse engineering technique, we found that @StillYesScot, @IsThisAB0t and @FAO_Scotbot, which are all bots promoting Scottish independence, have similar behaviour as shown in Figure 7.9. They all use similar hashtags (#indref2, #DisolveTheUnion) and essentially the same rule constructs; this suggests that they may have the same political masters. Further monitoring of these accounts through periodic analysis of rules describing their strategies could lead to the identification of their master, for example, if we found a rule indicating that these bots are all retweeting from a particular Twitter account which is known to belong to the leader of the campaign they are supporting.

7.2.5 Policy positions on specific issues

We use the approach proposed in Section 5.2 to investigate the policy position of the accounts on specific issues beyond what are popularly known. This helps to understand their scope and attempted influence in related areas. For example, we asked about the position of @StillYesScot on the Brexit deal and the relevant votes by MPs. Figure 7.10 shows absolute sentiments of @StillYesScot on these topics.

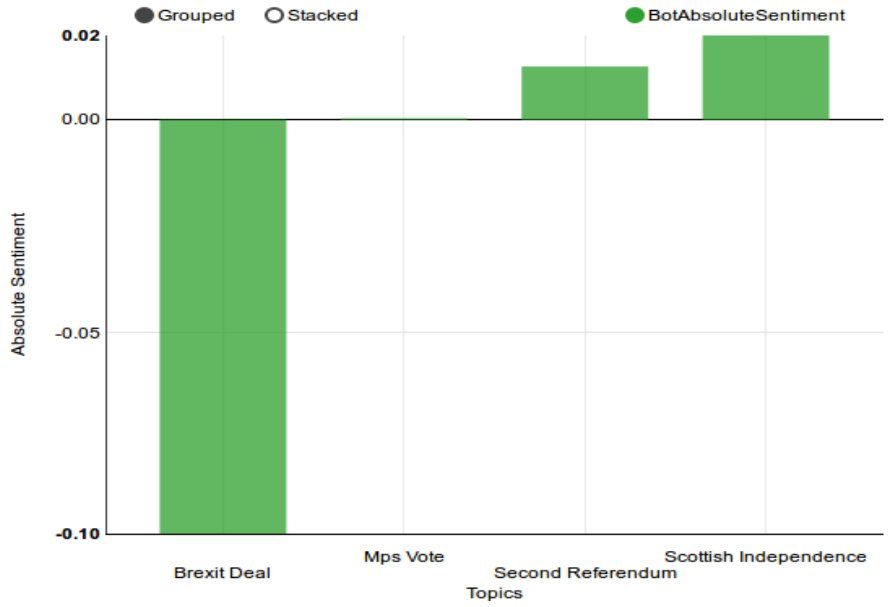


Figure 7.10: @StillYesScot policy on Brexit deal, MPs vote, Second referendum and Scottish Independence

The result in Figure 7.10 shows that @StillYesScot is against the Brexit deal more strongly than it promotes Scottish independence. This could be because the Brexit deal was the primary issue on social media during that period. Figure 7.11 shows how the bot's sentiments change over time with regard to those topics.

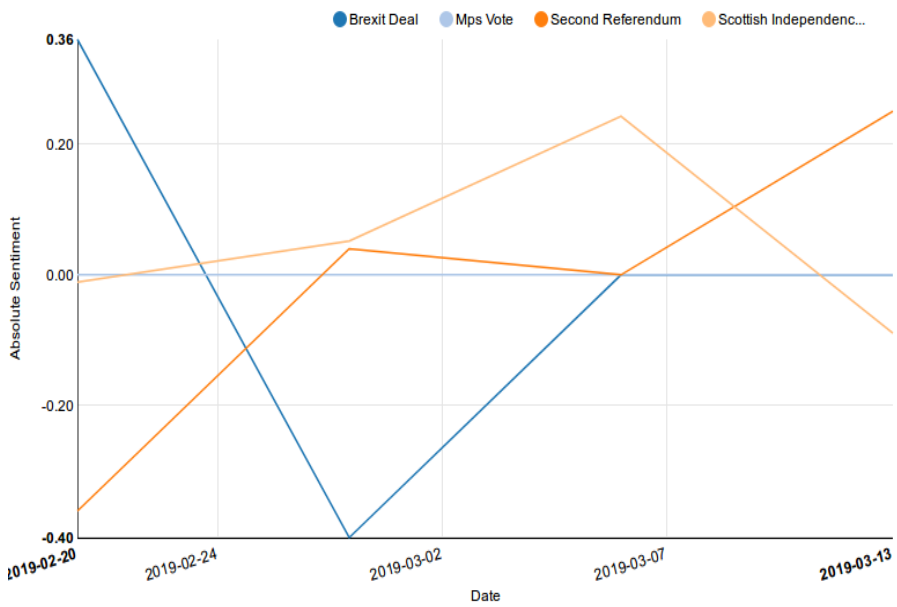
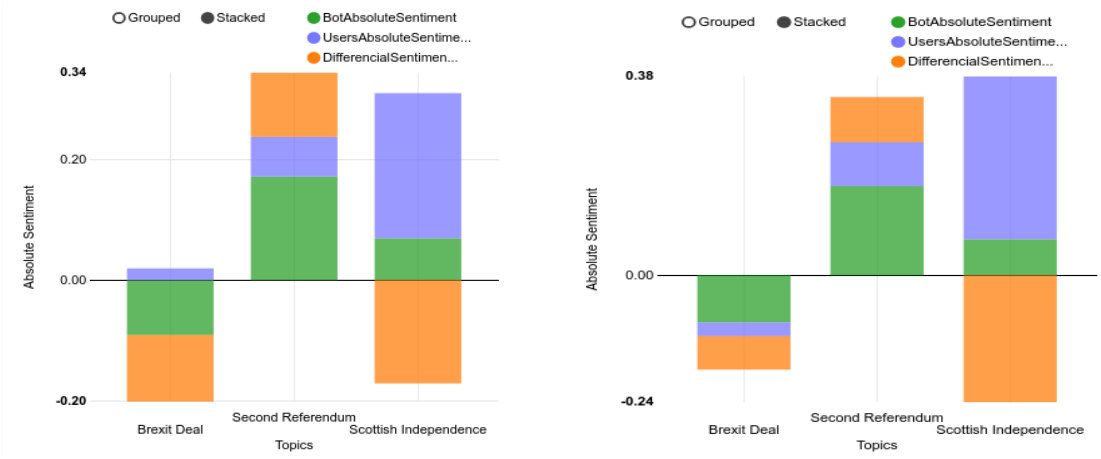


Figure 7.11: @StillYesScot's policy overtime

Figure 7.11 shows that between 20th February and 7th March, the period when the UK MPs began their deliberations on the Brexit deal, the account (@StillYesScot) campaigns strongly against the deal while from the 2nd to the 7 March the account continues with its usual campaign on Scottish independence. Next, is a study of bias and antagonism of the accounts.

7.2.6 Bias and antagonism

Social connections can be formed between parties of the same or different opinions. While the former is more common in a physical social network, in an online social network, the latter can be considered as an attempt by one party to influence the other. We analyse the opinions of users (friends) followed by the bots to investigate antagonism. Figure 7.12a shows the sentiments of @StillYesSot versus its friends. While we can not establish actual influence due to other external influencing factors, we note that among the users followed by the bot there are some who support the Brexit deal while the bot is known to be against it. This could be an attempt to share its negative opinion with these users while it promotes a second referendum and Scottish independence.



(a) Opinion of @StillYesScot vs Its Friends (b) Opinion of @StillYesScot vs Its followers

Figure 7.12: Bias and Antagonism of @StillYesScot

We also analysed the sentiments of tweets produced by the bots versus those of their followers to investigate the role of the bots in their networks. Figure 7.12b shows that the absolute sentiment of @StillYesScot on the Brexit deal and a second referendum is higher than that of all the users in the network. This is a sign that the account is trying to exert some form impact on other users in this direction.

7.3 Case Study II: Social Media Campaign Strategies in the 2019 Nigerian Elections

Twitter and other social media platforms play an important role in election campaigns. The use of social media in elections has been examined by numerous studies [33, 56, 68, 78, 90, 105]. However, these studies tend to place greater emphasis on the prediction of election outcomes or the nature of the interaction between online users and political candidates and/or parties. In this section, we show how our reverse engineering approach can be used to study the campaign strategies of a particular political party or candidate. This can support research for analysis and identification of good campaign strategies. It can also allow political candidates to improve their campaign strategies by knowing the strategies of their opponent.

News media plays a strategic role in shaping our daily discussion. During election campaigns, politicians can use news media to spread their political agenda. In this section, we also show how we use our proposed concept of differential analysis to study the correlation between the online users' opinions and information coming from the top Nigerian news media.

7.3.1 Data Collection and Analysis

We manually collected the Twitter handles of all political parties and presidential candidates. We used the Twitter API to continuously collect tweets related to the election keywords, hashtags and the handles from 4 December 2019 to 12 February 2019, covering the peak period of the election campaigns. The election was held on 23 February 2019 after being postponed at 3:00 on the initial day (16 February 2019). In addition to the data from political parties and presidential candidates, we collect tweets from the news media and the general public to understand their opinions on the parties and the candidates. The news media dataset was obtained from the following Twitter handles *@GuardianNigeria, @daily_trust, @nigeriantribune, @thesunnigeria, @SaharaReporters, @THISDAYLIVE, @PremiumTimesng, @LeadershipNGA, @MobilePunch, @NTANewsNow, @vanguardngrnews, @channelstv, @TheNationNews , and @nanonlineng* . The dataset is summarised in Table 7.1.

Table 7.1: Dataset statistics

Category	**Total Number Tweets**	**Proportion(%)**
Parties	13,873	0.73
Candidates	32,254	1.70
News Media	14,000	0.73
Users	1,835,613	96.82

7.3.2 Opinion polarisation

Our goal in this section is to understand the correlation between online users' opinions and the sentiments coming from the news media. In a political discussion, users may concentrate on promoting candidates of their interest taking a position against other candidates, or both. Understanding the nature of the online conversation is crucial in quantifying the support received by a political party or candidate. We used the proposed concept of mean absolute sentiment (MAS) to understand the complexion of the online discussion. Figure 7.13 shows the overall absolute sentiments of the users towards the political parties. We observe that the overall opinion of the users turns negative about the major parties @*officialAPCNig* and @*OfficialPDPNig*, with @*OfficialPDPNig* being more negative. This is an indication of a fierce confrontation between the supporters of the two major political parties. Similarly, in Figure 7.14, the candidates (@*atiku* and @*Mbuhari*) of the major parties (@*officialAPCNig* and @*OfficialPDPNig*, respectively) stood out above the other candidates. Despite the confrontation between their supporters, @*atiku* tends to have more positive sentiments than @*Mbuhari*.

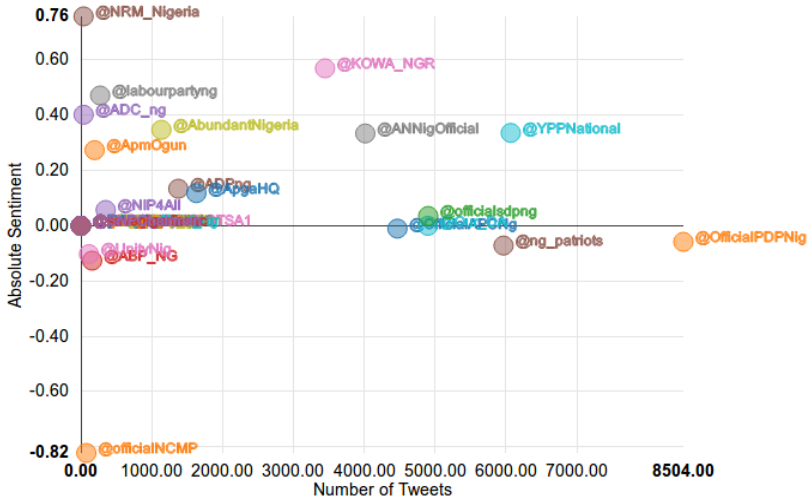


Figure 7.13: Overall opinions of the users on the political parties

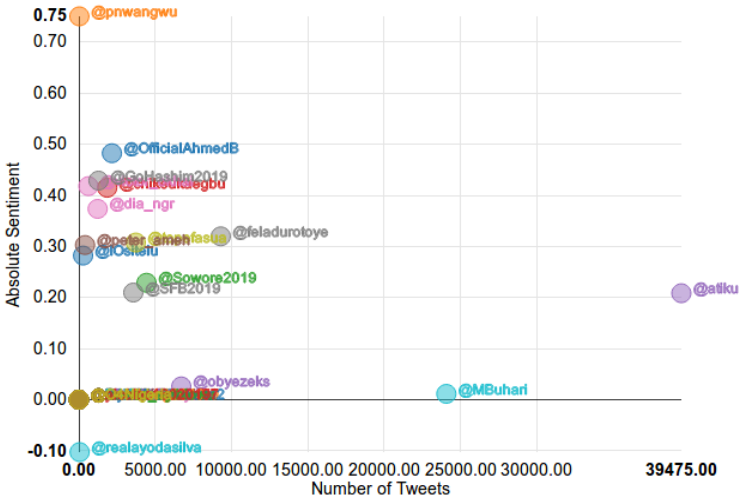


Figure 7.14: Overall opinions of the users on the candidates

We analysed the information disseminated by 12 major Nigerian media outlets and correlated it with the opinions of the users about the two major parties and their respective candidates. This was to assess the influence of the news media on online users during the election period. Figure 7.15 shows the absolute cumulative sentiment of the media and the users. The news produced by the media is mainly positive with regard to the candidates but negative with regard to their parties. A further investigation shows that a month before the election the media was full of news about the defection of members from one party to another and the dissatisfaction of members of parties over the primary elections in various states.

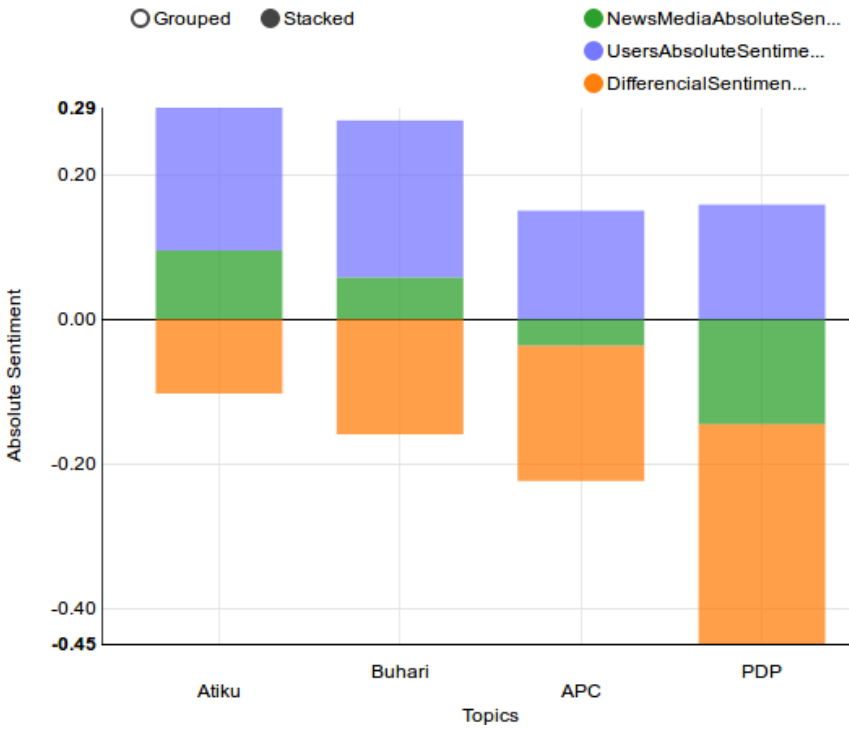
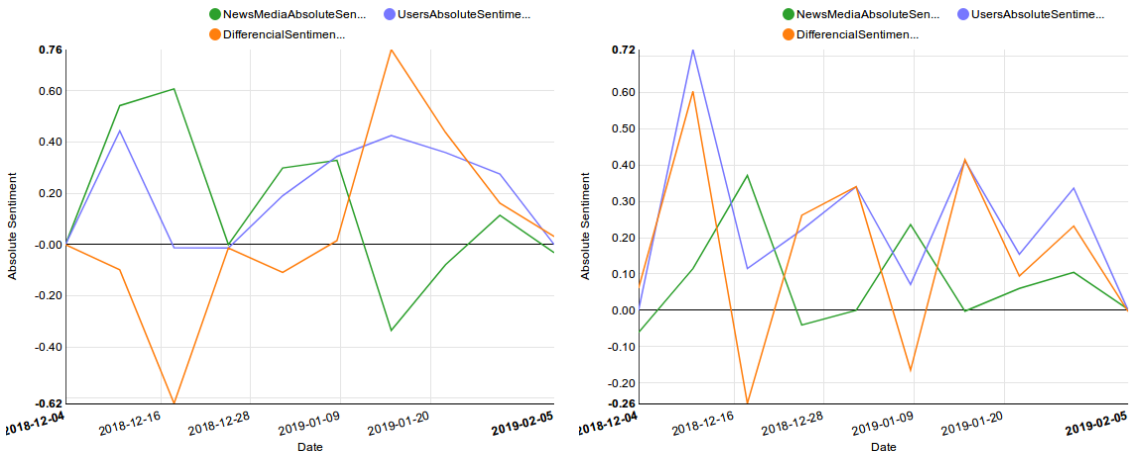


Figure 7.15: Opinion of news media and users on the two major parties and candidates



(a) Sentiment of news media and users on Atiku over time

(b) Sentiment of news media and users on Buhari over time

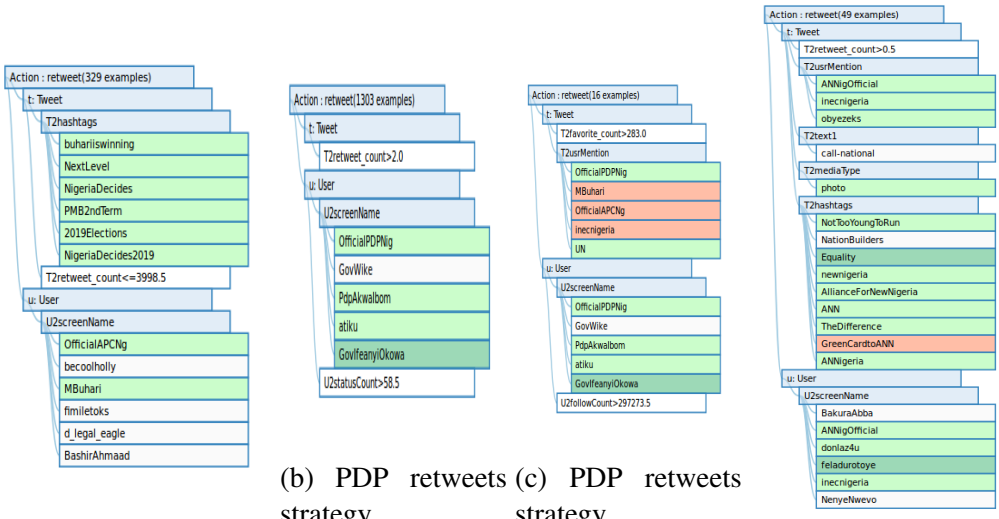
Figure 7.16: Differential sentiment over time of news media and users on the two major candidates

Figure 7.16a shows the development of sentiments over time. Differential sentiment analysis indicates the differences in sentiment between news media and online users. We found that from the 9 January to the 30 January 2019, negative news mostly dominated the media while the users' opinions were generally positive. Scrutiny of tweets from that period shows that the media was dominated with news about the ban on the main op-

position candidate (@*atiku*) from entering the United States of America and accusations of fraud. Despite this negative news, it is interesting to see that the overall sentiment about the opposition candidate (@*atiku*) was more favourable than that of the incumbent (@*Mbuhari*). This means the negative news did not have a significant impact on the online community, even if a slight decline in the users' sentiments can be observed. Figure 7.16b illustrates the sentiment of the media compared to that of the users about @*Mbuhari*. We note that there was not much negative news from the media about @*Mbuhari* as observed in @*atiku*. This gives a sense of what happened via public media channel during the election period, where supporters of the ruling party used earlier charges against @*atiku* to harm him on the news, while supporters of @*atiku* use the news to advertise their candidate, rather than attacking the ruling party.

7.3.3 Campaign Strategies

We used the proposed reverse engineering approach to obtain a concise, descriptive description of the behaviour of the political parties. While the results shown in Figure 7.17 includes details of the underlying behavioural patterns, we will concentrate on the surface information they reveal. This is because the details are more relevant to the understanding of automated accounts.



(a) APC retweet strategy

(b) PDP retweets strategy

(c) PDP retweets strategy

(d) ANN retweets strategy

Figure 7.17: APC, PDP and ANN retweets strategy

According to Figure 7.17a, the ruling party's account, @*OfficialAPCNg*, is mainly promoting tweets which contain hashtags of support (*#NextLevel,#PMB2ndTerm*) for its candidate (@*Mbuhari*) and retweeting its candidate's tweets. On the other hand, the top opposition party, @*OfficialPDPNig*, operates differently, as shown in Figure 7.17c. In addition to promoting retweets of its candidate (@*tiku*), the party promotes negative tweets mentioning the ruling party (@*OfficialAPCNg*) and its candidate (@*Mbuhari*). We found that the less popular opposition parties mainly concentrate on promoting their candidate rather than opposing the ruling or other parties. Figure 7.17d shows an example of a party (*ANN*) that is mainly promoting tweets from its candidates and tweets with hashtags supporting its candidate. The subsequent section presents an analysis of the election from a Twitter perspective and compares it with the actual election outcomes in order to understand the extent of the online analysis.

7.3.4 Twitter Analysis of the 2019 Nigerian Election

Despite the growing number of studies concerning the role of social media in elections, little attention has been paid to Nigeria, Africa's largest democracy. Some of the most sig-

nificant challenges affecting election analysis using social media in developing countries are the availability of the data, because not many people are using social media, and the difficulty in obtaining the users' demographic information. In this section, we investigate the extent to which we can identify online users from various states of the country and the extent to which an online analysis could mirror election outcomes.

Users-location labelling: Identifying the location of each user will enable us to analyse the users' opinions across different parts of the country. In Twitter, the location of a given user can be identified through (1) analysing the tweets' geo-coordinates; this is only possible if this feature has been explicitly enabled by the user, or (2) identifying a *self-reported* location in the users' profile as provided during the account creation process. Since users rarely enable geo-tagging of their tweets, and different locations can be tagged, we utilised the second option for our analysis as we considered this to be the reliable. The account location is substantially noisy due to imprecise tagging by users. To mitigate this issue, we took an additional pre-processing step of labelling the location of each user. We created a dictionary of states and major cities in the country and implemented a program to automatically map the users' locations to a city or state in the country. A user location is recorded as *'None'* if neither a state nor city name can be found that is associated with their location.

Election Analysis: Here we describe how we computed the election tendencies across various states of the country using Twitter analysis and compared it with the election results.

To ensure a representative outcome, our analysis is based on replicating the idea of *one-citizen, one-vote*. Therefore, for each state, we aggregate all the tweets from each user and compute the overall sentiment of the user concerning the two major candidates – @*tiku* and @*MBuhari*. For instance, if the overall sentiment is positively in favour of any of the contestants, the user is assumed to be voting for that candidate. For each state, we sum up the overall votes for each candidate and return the candidate with the highest positive sentiment as the winner. Figure 7.18 shows the election outcomes which were announced

by the authorised electoral body (INEC), whilst Figure 7.19 shows the result based on our tweets analysis. The states won by the opposition party's candidate (Atiku of PDP) are highlighted in red whilst won by the incumbent (Buhari of APC) are highlighted in green.

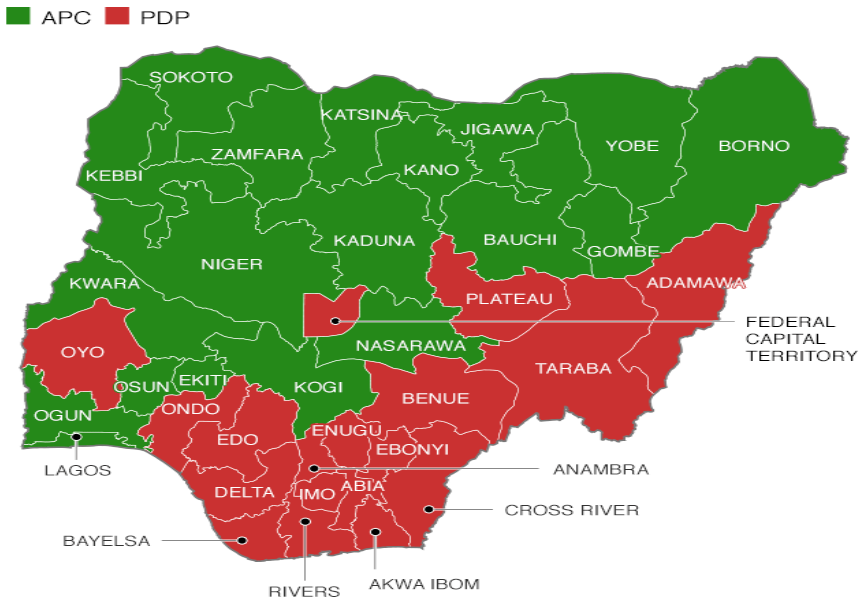


Figure 7.18: Actual election result [83]

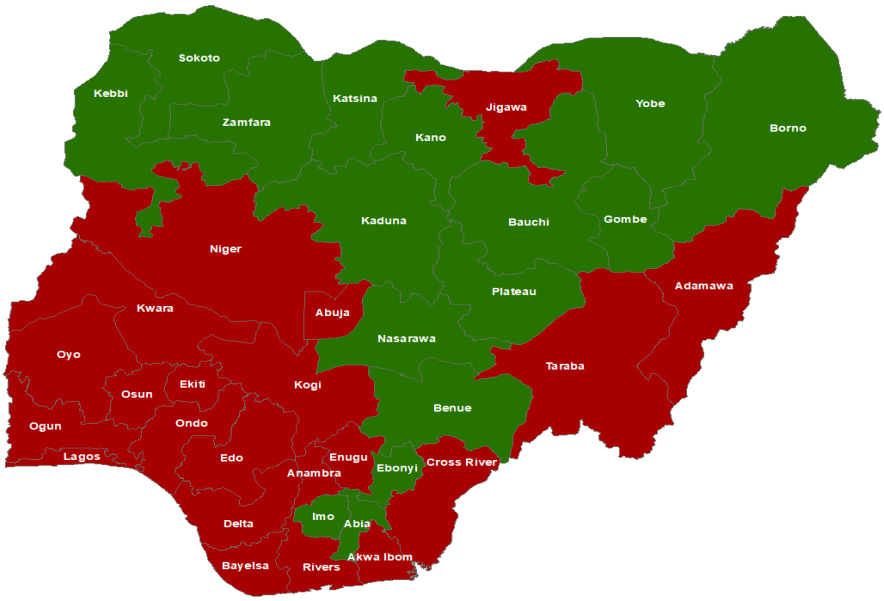


Figure 7.19: Results based on Twitter analysis. States won by Atiku (PDP party candidate) are highlighted in red colour and the states won by Buhari (APC candidate) are highlighted in green

Comparing the election results shown in Figure 7.18 with those of our analysis shown in Figure 7.19, shows that the analysis matches the outcome in 64.8% of the states. In other

words, out of the 37 states (including the Federal Capital Territory), the project results for 24 states reflect the results of the election. The states that contradict the election results – *Jigawa, Niger, Kwara, Ekiti, Osun, Ogun, Lagos* – were won by @*MBuhari* but won by the opposition candidate (@*atiku*) in our Twitter analysis. The other states, namely *Plateau, Benue, Ebonyi, Abia, Imo, Oyo* were won by @*atiku* (marked green in our results, red in the election result) not @*MBuhari*. Furthermore, we compute the overall online result, including users whose location is Nigeria, but without any particular state, we found that @*atiku* received 51.64% and @*MBuhari* 48.36% votes. However, in the election result, @*MBuhari* received 55.6% of the votes while @*atiku* received 41.2% of the votes.

Concluding, a correct election analysis from social media requires adequate sampling methods to correctly identify voters to obtain an unbiased representation of the sample [47]. The data used in this study is 6.9% of the total votes cast in the election, and thus the sample is not well representative of voters across all states. However, what this analysis does show is that we can get a sample of Twitter users from each state of the country; this was previously difficult but in future elections it appears we now have a valuable tool by which we can conduct more comprehensive analyses.

7.4 Summary

In this chapter, we have presented a prototype tool created as a proof of concept. The tool supports collection of traces and the reverse engineering of the behaviour of Twitter bots using the proposed approach for rules extraction and visualisation (cf. Chapter 4). It allows analysis of Twitter accounts using the proposed differential sentiment analysis (cf. Chapter 5).

With the aid of the prototype tool, we have used two case studies to demonstrate the applications of the proposed approach. In a case study of post-Brexit debates, we have shown how we used the concepts proposed in Chapter 5 to identify strongly opinionated

bot accounts, study their policy position and understand their bias and antagonism. We have shown how we used the rule learning approach proposed in Chapter 4 to understand how bots carry out their campaigns, infer their potential effects and potentially identify their masters. In a case study of the 2019 Nigerian election, we have shown how we used the concepts proposed in Chapter 5 to understand the nature of users' online conversations about the parties and candidates. We have also used the differential analysis to study the correlation between the online users' opinions and information disseminated by the news media. We have used the rule learning to study the parties' campaign strategies.

Chapter 8

Conclusion and Future Directions

Social media platforms such as Twitter play an important role in the public's participation in social issues and other democratic activities. The extent to which social media platforms are used for political campaigns and the use of social bots to promote political agendas raises concerns about the possibility of manipulating public opinion using bots. While there are approaches to distinguish automated accounts from regular user accounts, information about their masters, strategies, bias and antagonism on their target audience remains harder to obtain.

This thesis proposed an approach to reverse engineer the behaviour of Twitter bots to provide information that will help to identify their masters, targets and strategies on social media. It contributes to knowledge with respect to the understanding of their behaviour and strategies. The following sections summarise the main contributions of the thesis.

8.1 Summary of Thesis Achievements

8.1.1 About the Behaviour of Bots

We study the behaviour of Twitter bots and propose an approach for reverse engineering their behaviour. Reverse engineering requires extraction, analysis and mapping of different pieces of information to build a correct high-level representation of the system in question. To facilitate the extraction and analysis, by using the concepts of graph transformation we model the behaviour of Twitter bots by creating a meta-model of the Twitter API (cf. Chapter 3). We constituted a set of attributes for the collection of behavioural traces from Twitter accounts (cf. Chapter 3) and developed an extensible crawler to facilitate the collection of the traces (cf. Chapter 7). With the aid of these, we achieved the following contributions as presented in Chapter 4.

- An approach to observe the behaviour of a bot and represent it using a set of understandable rules (cf. Section 4.1)
- An approach to automatically extract rules from observations of a bot's behaviour (cf. Section 4.2).
- An approach for visualisation of the rules (cf. Section 4.3).

8.1.2 About Opinion, Bias and Antagonism of a Bot

Social bots are orchestrated to promote a specific agenda, or to propagate information in support of or against certain topics. In many cases, bots are not straight in their opinion on a topic. A bot that promotes a positive opinion on a topic can share some weak negative opinions to attract users with opposing opinions in an attempt to increase influence.

- We propose a model for identifying topics of interest to a bot and opinion of the bot on those topics (cf. Section 5.2). In Section 7.2, we have shown how we use

the proposed concept to identify strongly opinionated accounts in a large group of accounts with different opinions, and in Section 7.3, we used the concept to understand the level of confrontation between opponents and supporters of the political candidates.

- We proposed an approach for study of the bias and antagonism of a bot (differential sentiment) (cf. Section 5.3). In Section 7.2, we have shown how we used the proposed approach to study the bias and antagonism of bots after the United Kingdom (U.K.) voted to exit from the European Union (EU). In Section 7.3, we have shown how we used the proposed approach to study the relationship between the opinions of online users and the Nigerian news media to understand the relationship between the two and to detect antagonism.

- We propose the notion of differential sentiment over time as a means to track changes in the behaviour of bots and antagonism (cf. Section 5.4).

8.1.3 Tool support and Application

- As proof of concept, we provide an implementation of our proposals. i) Tool for the extraction and visualisation of the rules controlling the behaviour of a Twitter bot. ii) Tools for both absolute and differential sentiment analysis. These are available on GitHub [1].

- We demonstrate the application and utilization of the proposals using case studies: i) study of the behaviour of bots in politics after the UK vote to exit from EU, and ii) analysis of campaign strategies on social media during the 2019 Nigerian elections.

[1] https://github.com/bellobichi2/botscope

8.2 Scope and Limitations

This thesis proposed a reverse engineering approach to understand the behaviour of Twitter bots. The automation of the approach relies on machine learning, which itself depends on the availability of patterns to discover behaviour. Regardless of the type of accounts, the approach achieves higher accuracy if there is a pattern, repetitive content structure or behaviour, or an association between tweet features and/or user features. Although the approach can reveal patterns of content produced by bots, it does not cover how the bots actually generate their contents, e.g., the algorithm used to generate the text content.

With regard to applications, as demonstrated using running examples in Chapter 5 and case studies in Section 7.2 and Section 7.3, the approach could be useful to researchers in the study of the role of social bots in online campaigns. Businesses and politicians could use the reverse engineering approach to discover the campaign strategies of their opponents and improve their own campaigns. It could assist security agencies in the discovery of online extremists or activism activities that could become a threat to national security.

8.3 Future Directions

This section presents a range of possible future research to extend the work proposed in this thesis and my further research direction in general.

8.3.1 Rule Analysis and Inference

This thesis aims to describe the behaviour of Twitter bots. The rules produced could be used for further analysis. In Section 7.2, we have shown bots with similar rule constructs. We want to study the utilization of the exported rules for the detection of bots with similar rule constructs and investigate how to extrapolate the relationship obtained for the detec-

tion of their masters. We would also like to explore the utilization of the rules for the detection of similar campaigns and botnets.

8.3.2 Link Content Analysis

The proposed approach in this thesis analyses the contents of tweets and exposes the URLs which a bot uses as a source of its content or to direct users. However, the approach does not analyse the content of the links. The description of a bot's behaviour produced by the approach can be enhanced by incorporating the sentiment and content of the links into the description of the behaviour. We want to study methods of detecting fake news and incorporate it in such a way that the approach can automatically indicate if a URL contain fake news.

8.3.3 Improve the utilization of the prototype tool

We have developed a prototype tool as proof of our proposals. The prototype tool has been presented to social scientists at the Department of Media and Communication, University of Leicester, UK. The feedback that we received includes the need for data extraction from the models, e.g., exporting data at any point where the differential sentiment over time indicates antagonism. We would like to explore how to improve the utilization of the tool through collaboration with social scientists. Using the tool, we would like to: i) study how bots' rules changes over time, and detect and interpret such changes; ii) study the behaviour of humans and bots in different scenarios, outside and during campaign periods to analyse and interpret the results. These could show if the approach could assist in distinguishing bot and human accounts and if humans behave in a similar manner to bots during a campaign period, as shown by our results in Figures 6.5, 6.6, and 6.7.

8.3.4 Detect the use of Algorithms and Automation for social manipulation

Social media has become a significant part of our lives. It has become a major source of information and opinions about food, health matters (e.g., vaccinations and medical care), products, politics, and various other social issues. The development of algorithms and technologies for marketing strategies and user preferences has endangered our social values, and taken our freedom to receive free and fair information. The use of such algorithms for political propaganda will endanger our democracy especially in Africa where social media is now seen as a gateway to freedom of information without realising the use of algorithms for social manipulation. Many people tend to believe in what is popular on social media. Recently, specifically May 2019, an Israel firm was found spreading misinformation targeting Africa, using fake accounts to post news on politics and elections in various countries in the continent[2]. Furthermore, Facebook was reported to have removed 265 Facebook and Instagram accounts originating from Israel that were targeting Nigeria, Senegal, Togo, Angola, Niger and Tunisia, along with some other countries in Asia and Latin America[3]. I would like to focus on developing models and algorithms that can be used to detect and study the use of algorithms on social media for social manipulation. Considering that many researchers have focused on the U.S and Europe, I would like to pay special attention to Africa. To pursue my dreams of using computer science models and algorithms to address social and mental health problems, I would further like to collaborate with social scientists.

[2]https://www.bbc.co.uk/news/business-48305032

[3]https://newsroom.fb.com/news/2019/05/removing-coordinated-inauthentic-behavior-from-israel/

Bibliography

[1] Abdullah Alshanqiti, Reiko Heckel, and Timo Kehrer. Inferring visual contracts from java programs. *Automated Software Engineering*, 25(4):745–784, 2018.

[2] Sinan Aral, Lev Muchnik, and Arun Sundararajan. Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks. *Proceedings of the National Academy of Sciences*, 106(51):21544–21549, 2009.

[3] Sinan Aral and Dylan Walker. Identifying influential and susceptible members of social networks. *Science*, 337(6092):337–341, 2012.

[4] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. In *Lrec*, volume 10, pages 2200–2204, 2010.

[5] Adam Badawy, Aseel Addawood, Kristina Lerman, and Emilio Ferrara. Characterizing the 2016 russian IRA influence campaign. *Social Network Analysis and Mining*, 9(1):31:1–31:11, 2019.

[6] Adam Badawy, Emilio Ferrara, and Kristina Lerman. Analyzing the digital traces of political manipulation: The 2016 russian interference twitter campaign. In *IEEE/ACM 2018 International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2018, Barcelona, Spain, August 28-31, 2018*, pages 258–265, 2018.

[7] Eytan Bakshy, Jake M. Hofman, Winter A. Mason, and Duncan J. Watts. Everyone's an influencer: quantifying influence on twitter. In *Proceedings of the Forth*

International Conference on Web Search and Web Data Mining, WSDM 2011, Hong Kong, China, February 9-12, 2011, pages 65–74, 2011.

[8] Alessandro Bessi and Emilio Ferrara. Social bots distort the 2016 us presidential election online discussion. 2016.

[9] Sajid Yousuf Bhat and Muhammad Abulaish. Analysis and mining of online social networks: emerging trends and challenges. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 3(6):408–444, 2013.

[10] David W. Binkley. Source code analysis: A road map. In *International Conference on Software Engineering, ISCE 2007, Workshop on the Future of Software Engineering, FOSE 2007, May 23-25, 2007, Minneapolis, MN, USA*, pages 104–119, 2007.

[11] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[12] Robert M Bond, Christopher J Fariss, Jason J Jones, Adam DI Kramer, Cameron Marlow, Jaime E Settle, and James H Fowler. A 61-million-person experiment in social influence and political mobilization. *Nature*, 489(7415):295, 2012.

[13] Margaret M Bradley and Peter J Lang. Affective norms for english words (anew): Instruction manual and affective ratings. Technical report, Citeseer, 1999.

[14] Axel Bruns and Jean E Burgess. The use of twitter hashtags in the formation of ad hoc publics. In *Proceedings of the 6th European consortium for political research (ECPR) general conference 2011*, 2011.

[15] Erik Cambria, Catherine Havasi, and Amir Hussain. Senticnet 2: A semantic and affective resource for opinion mining and sentiment analysis. In *FLAIRS conference*, pages 202–207, 2012.

[16] Donald Thomas Campbell and Thomas D Cook. *Quasi-experimentation: Design & analysis issues for field settings*. Rand McNally College Publishing Company Chicago, 1979.

[17] Gerardo Canfora, Massimiliano Di Penta, and Luigi Cerulo. Achievements and challenges in software reverse engineering. *Commun. ACM*, 54(4):142–151, 2011.

[18] Qiang Cao, Michael Sirivianos, Xiaowei Yang, and Tiago Pregueiro. Aiding the detection of fake accounts in large scale social online services. In *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*, pages 15–15. USENIX Association, 2012.

[19] Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28, 2014.

[20] Nikan Chavoshi, Hossein Hamooni, and Abdullah Mueen. Debot: Twitter bot detection via warped correlation. In *ICDM*, pages 817–822, 2016.

[21] Nikan Chavoshi, Hossein Hamooni, and Abdullah Mueen. On-demand bot detection and archival system. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 183–187. International World Wide Web Conferences Steering Committee, 2017.

[22] Nikan Chavoshi, Hossein Hamooni, and Abdullah Mueen. Temporal patterns in bot activities. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 1601–1606. International World Wide Web Conferences Steering Committee, 2017.

[23] Elliot J. Chikofsky and James H Cross. Reverse engineering and design recovery: A taxonomy. *IEEE software*, 7(1):13–17, 1990.

[24] Zi Chu, Steven Gianvecchio, Haining Wang, and Sushil Jajodia. Who is tweeting on twitter: human, bot, or cyborg? In *Twenty-Sixth Annual Computer Security Applications Conference, ACSAC 2010, Austin, Texas, USA, 6-10 December 2010*, pages 21–30, 2010.

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

[26] Michael D. Conover, Emilio Ferrara, Filippo Menczer, and Alessandro Flammini. The digital evolution of occupy wall street. *CoRR*, abs/1306.5474, 2013.

[27] Michael D. Conover, Bruno Gonçalves, Jacob Ratkiewicz, Alessandro Flammini, and Filippo Menczer. Predicting the political alignment of twitter users. In *PASSAT/SocialCom 2011, Privacy, Security, Risk and Trust (PASSAT), 2011 IEEE Third International Conference on and 2011 IEEE Third International Conference on Social Computing (SocialCom), Boston, MA, USA, 9-11 Oct., 2011*, pages 192–199, 2011.

[28] Michael D Conover, Jacob Ratkiewicz, Matthew Francisco, Bruno Gonçalves, Filippo Menczer, and Alessandro Flammini. Political polarization on twitter. In *Fifth international AAAI conference on weblogs and social media*, 2011.

[29] George Danezis and Prateek Mittal. Sybilinfer: Detecting sybil nodes using social networks. In *Proceedings of the Network and Distributed System Security Symposium, NDSS 2009, San Diego, California, USA, 8th February - 11th February 2009*, 2009.

[30] Clayton Allen Davis, Onur Varol, Emilio Ferrara, Alessandro Flammini, and Filippo Menczer. Botornot: A system to evaluate social bots. In *Proceedings of the 25th International Conference Companion on World Wide Web*, pages 273–274. International World Wide Web Conferences Steering Committee, 2016.

[31] Morton Deutsch and Harold B Gerard. A study of normative and informational social influences upon individual judgment. *The journal of abnormal and social psychology*, 51(3):629, 1955.

[32] John P Dickerson, Vadim Kagan, and VS 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, pages 620–627. IEEE Press, 2014.

[33] Joseph DiGrazia, Karissa McKelvey, Johan Bollen, and Fabio Rojas. More tweets, more votes: Social media as a quantitative indicator of political behavior. *PloS one*, 8(11):e79449, 2013.

[34] Jack Dorsey and Ned D. Segal. Twitter 2019 third quarter earnings report. `https://s22.q4cdn.com/826641620/files/doc_financials/2019/q3/Q3'19-Earnings-Press-Release-FINAL-(1).pdf`, October 24 2019. Accessed: 30/12/2019.

[35] Sara El-Khalili. Social media as a government propaganda tool in post-revolutionary egypt. *First Monday*, 18(3), 2013.

[36] Aviad Elyashar, Michael Fire, Dima Kagan, and Yuval Elovici. Homing socialbots: intrusion on a specific organization's employee using socialbots. In *2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2013)*, pages 1358–1365. IEEE, 2013.

[37] Gunn Sara Enli and Eli Skogerbø. Personalized campaigns in party-centred politics: Twitter and facebook as arenas for political communication. *Information, communication & society*, 16(5):757–774, 2013.

[38] Michael D. Ernst. Static and dynamic analysis: synergy and duality. In *Proceedings of the 2004 ACM SIGPLAN-SIGSOFT Workshop on Program Analysis For Software Tools and Engineering, PASTE'04, Washington, DC, USA, June 7-8, 2004*, page 35, 2004.

[39] Linda Ferguson. External validity, generalizability, and knowledge utilization. *Journal of Nursing Scholarship*, 36(1):16–22, 2004.

[40] Emilio Ferrara. Disinformation and social bot operations in the run up to the 2017 french presidential election. *First Monday*, 22(8), 2017.

[41] Emilio Ferrara, Onur Varol, Clayton Davis, Filippo Menczer, and Alessandro Flammini. The rise of social bots. *Communications of the ACM*, 59(7):96–104, 2016.

[42] Kevin J Flannelly, Laura T Flannelly, and Katherine RB Jankowski. Threats to the internal validity of experimental and quasi-experimental research in healthcare. *Journal of health care chaplaincy*, 24(3):107–130, 2018.

[43] Michelle Forelle, Philip N. Howard, Andrés Monroy-Hernández, and Saiph Savage. Political bots and the manipulation of public opinion in venezuela. *CoRR*, abs/1507.07109, 2015.

[44] Carlos Freitas, Fabricio Benevenuto, Saptarshi Ghosh, and Adriano Veloso. Reverse engineering socialbot infiltration strategies in twitter. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pages 25–32. ACM, 2015.

[45] Jennifer Fromm, Stefanie Melzer, Björn Ross, and Stefan Stieglitz. Trump versus clinton: Twitter communication during the US primaries. In *Network Intelligence Meets User Centered Social Media Networks [4th European Network Intelligence Conference, ENIC 2017, Duisburg, Germany, September 11-12, 2017]*, pages 201–217, 2017.

[46] NL (Nathaniel Lees) Gage and Julian C Stanley. *Experimental and quasi-experimental designs for research*. Chicago: R. McNally, 1963.

[47] Daniel Gayo-Avello, Panagiotis Takis Metaxas, and Eni Mustafaraj. Limits of electoral predictions using twitter. In *Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain, July 17-21, 2011*, 2011.

[48] Rachel K Gibson and Ian McAllister. Does cyber-campaigning win votes? online communication in the 2004 australian election. *Journal of Elections, Public Opinion and Parties*, 16(3):243–263, 2006.

[49] Zafar Gilani, Reza Farahbakhsh, Gareth Tyson, and Jon Crowcroft. A large-scale behavioural analysis of bots and humans on twitter. *TWEB*, 13(1):7:1–7:23, 2019.

[50] Zafar Gilani, Reza Farahbakhsh, Gareth Tyson, Liang Wang, and Jon Crowcroft. Of bots and humans (on twitter). In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017, Sydney, Australia, July 31 - August 03, 2017*, pages 349–354, 2017.

[51] Zafar Gilani, Ekaterina Kochmar, and Jon Crowcroft. Classification of twitter accounts into automated agents and human users. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, pages 489–496. ACM, 2017.

[52] Zafar Gilani, Liang Wang, Jon Crowcroft, Mario Almeida, and Reza Farahbakhsh. Stweeler: A framework for twitter bot analysis. In *Proceedings of the 25th International Conference Companion on World Wide Web*, pages 37–38. International World Wide Web Conferences Steering Committee, 2016.

[53] CJ Hutto Eric Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth International Conference on Weblogs and Social Media (ICWSM-14). Available at (20/04/16) http://comp. social. gatech. edu/papers/icwsm14. vader. hutto. pdf*, 2014.

[54] Sandra González-Bailón, Javier Borge-Holthoefer, and Yamir Moreno. Broadcasters and hidden influentials in online protest diffusion. *CoRR*, abs/1203.1868, 2012.

[55] Sandra González-Bailón, Javier Borge-Holthoefer, Alejandro Rivero, and Yamir Moreno. The dynamics of protest recruitment through an online network. *CoRR*, abs/1111.5595, 2011.

[56] Todd Graham, Dan Jackson, and Marcel Broersma. New platform, old habits? candidates' use of twitter during the 2010 british and dutch general election campaigns. *New media & society*, 18(5):765–783, 2016.

[57] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003.

[58] Mark A. Hall and Lloyd A. Smith. Feature selection for machine learning: Comparing a correlation-based filter approach to the wrapper. In *Proceedings of the Twelfth International Florida Artificial Intelligence Research Society Conference, May 1-5, 1999, Orlando, Florida, USA*, pages 235–239, 1999.

[59] Mark Andrew Hall. Correlation-based feature selection for machine learning. 1999.

[60] Reiko Heckel. Graph transformation in a nutshell. *Electronic notes in theoretical computer science*, 148(1):187–198, 2006.

[61] Tin Kam Ho. Random decision forests. In *Document analysis and recognition, 1995., proceedings of the third international conference on*, volume 1, pages 278–282. IEEE, 1995.

[62] Liangjie Hong and Brian D Davison. Empirical study of topic modeling in twitter. In *Proceedings of the first workshop on social media analytics*, pages 80–88. ACM, 2010.

[63] Philip N Howard et al. *New media campaigns and the managed citizen*. Cambridge University Press, 2006.

[64] Philip N Howard and Bence Kollanyi. Bots,# strongerin, and# brexit: Computational propaganda during the uk-eu referendum. *Available at SSRN 2798311*, 2016.

[65] Pik-Mai Hui, Kai-Cheng Yang, Christopher Torres-Lugo, Zachary Monroe, Marc McCarty, B Serrette, Valentin Pentchev, and Filippo Menczer. Botslayer: real-time detection of bot amplification on twitter. *Journal of Open Source Software*, 4(42):1706, 2019.

[66] Brownlee J. (2017). A tour of machine learning algorithms,[online] machine learning mastery. available at: https://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/ [accessed 4 sep. 2017].

[67] Elıas Jónsson and Jake Stolee. An evaluation of topic modelling techniques for twitter.

[68] Vadim Kagan, Andrew Stevens, and VS Subrahmanian. Using twitter sentiment to forecast the 2013 pakistani election and the 2014 indian election. *IEEE Intelligent Systems*, 30(1):2–5, 2015.

[69] Bente Kalsnes, Arne H. Krumsvik, and Tanja Storsul. Social media as a political backchannel: Twitter use during televised election debates in norway. *Aslib J. Inf. Manag.*, 66(3):313–328, 2014.

[70] Kenneth A Kaufman and Ryszard S Michalski. 2-from data mining to knowledge mining. *Handbook of Statistics*, 24:47–75, 2005.

[71] Herbert C Kelman. Compliance, identification, and internalization three processes of attitude change. *Journal of conflict resolution*, 2(1):51–60, 1958.

[72] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

[73] Ron Kohavi and George H John. Wrappers for feature subset selection. *Artificial intelligence*, 97(1-2):273–324, 1997.

[74] Raffi Krikorian. Map of a twitter status object. *The Wall Street Journal*, 18, 2010.

[75] Pat Langley and Herbert A. Simon. Applications of machine learning and rule induction. *Commun. ACM*, 38(11):54–64, 1995.

[76] Hui Li, Sourav S Bhowmick, and Aixin Sun. Casino: towards conformity-aware social influence analysis in online social networks. In *Proceedings of the 20th*

ACM international conference on Information and knowledge management, pages 1007–1012. ACM, 2011.

[77] Luca Luceri, Ashok Deb, Adam Badawy, and Emilio Ferrara. Red bots do it better: Comparative analysis of social bot partisan behavior. In *Companion of The 2019 World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019.*, pages 1007–1012, 2019.

[78] Christopher Mascaro, Denise Agosto, and Sean P Goggins. One-sided conversations: The 2012 presidential election on twitter. In *Proceedings of the 17th International Digital Government Research Conference on Digital Government Research*, pages 112–121. ACM, 2016.

[79] Yan Mei, Youliang Zhong, and Jian Yang. Finding and analyzing principal features for measuring user influence on twitter. In *2015 IEEE First International Conference on Big Data Computing Service and Applications*, pages 478–486. IEEE, 2015.

[80] Panagiotis T Metaxas and Eni Mustafaraj. Social media and the elections. *Science*, 338(6106):472–473, 2012.

[81] Fred Morstatter, Jürgen Pfeffer, Huan Liu, and Kathleen M. Carley. Is the sample good enough? comparing data from twitter's streaming API with twitter's firehose. In *Proceedings of the Seventh International Conference on Weblogs and Social Media, ICWSM 2013, Cambridge, Massachusetts, USA, July 8-11, 2013.*, 2013.

[82] Michael L. Nelson. A survey of reverse engineering and program comprehension. *CoRR*, abs/cs/0503068, 2005.

[83] BBC News and INEC Nigeria. Nigerian election results. *BBC News World-Africa)*, 2019.

[84] Finn Årup Nielsen. A new anew: Evaluation of a word list for sentiment analysis in microblogs. *arXiv preprint arXiv:1103.2903*, 2011.

[85] Abigail Paradise, Rami Puzis, and Asaf Shabtai. Anti-reconnaissance tools: Detecting targeted socialbots. *IEEE Internet Computing*, 18(5):11–19, 2014.

[86] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake VanderPlas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[87] James W Pennebaker, Martha E Francis, and Roger J Booth. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, 71(2001):2001, 2001.

[88] J Ross Quinlan. Generating production rules from decision trees. In *ijcai*, volume 87, pages 304–307. Citeseer, 1987.

[89] J. Ross Quinlan. Simplifying decision trees. *International journal of man-machine studies*, 27(3):221–234, 1987.

[90] Dean Apriana Ramadhan, Yani Nurhadryani, and Irman Hermadi. Campaign 2.0: Analysis of social media utilization in 2014 jakarta legislative election. In *2014 International Conference on Advanced Computer Science and Information System*, pages 102–107. IEEE, 2014.

[91] Jacob Ratkiewicz, Michael D. Conover, Mark R. Meiss, Bruno Gonçalves, Alessandro Flammini, and Filippo Menczer. Detecting and tracking political abuse in social media. In *Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain, July 17-21, 2011*, 2011.

[92] Jacob Ratkiewicz, Michael D. Conover, Mark R. Meiss, Bruno Gonçalves, Snehal Patil, Alessandro Flammini, and Filippo Menczer. Truthy: mapping the spread of astroturf in microblog streams. In *Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28 - April 1, 2011 (Companion Volume)*, pages 249–252, 2011.

[93] Bertram H Raven. Social influence and power. Technical report, CALIFORNIA UNIV LOS ANGELES, 1964.

[94] Marnie E Rice and Grant T Harris. Comparing effect sizes in follow-up studies: Roc area, cohen's d, and r. *Law and human behavior*, 29(5):615–620, 2005.

[95] Sayad Saed. Data mining map,an introduction to data mining,retrieved 23 july 2017,http://www.saedsayad.com/data_mining_map.htm.

[96] Steven Salzberg. Book review: C4.5: programs for machine learning by j. ross quinlan. morgan kaufmann publishers, inc., 1993. *Machine Learning*, 16(3):235–240, 1994.

[97] Andrew Schaap. Political theory and the agony of politics. *Political Studies Review*, 5(1):56–74, 2007.

[98] Samantha Shorey and Philip N Howard. Automation, algorithms, and politics: A research review. *Int. J Comm*, 10, 2016.

[99] Philip J Stone and J Kirsh. Cambridge computer associates. 1966. the general inquirer: A computer approach to content analysis.

[100] Gianluca Stringhini, Christopher Kruegel, and Giovanni Vigna. Detecting spammers on social networks. In *Twenty-Sixth Annual Computer Security Applications Conference, ACSAC 2010, Austin, Texas, USA, 6-10 December 2010*, pages 1–9, 2010.

[101] VS Subrahmanian, Amos Azaria, Skylar Durst, Vadim Kagan, Aram Galstyan, Kristina Lerman, Linhong Zhu, Emilio Ferrara, Alessandro Flammini, Filippo Menczer, et al. The darpa twitter bot challenge. *arXiv preprint arXiv:1601.05140*, 2016.

[102] Mike Thelwall, Kevan Buckley, and Georgios Paltoglou. Sentiment strength detection for the social web. *Journal of the American Society for Information Science and Technology*, 63(1):163–173, 2012.

[103] Mike Thelwall, Kevan Buckley, Georgios Paltoglou, Di Cai, and Arvid Kappas. Sentiment strength detection in short informal text. *Journal of the American Society for Information Science and Technology*, 61(12):2544–2558, 2010.

[104] Zeynep Tufekci and Christopher Wilson. Social media and the decision to participate in political protest: Observations from tahrir square. *Journal of communication*, 62(2):363–379, 2012.

[105] Andranik Tumasjan, Timm Oliver Sprenger, Philipp G. Sandner, and Isabell M. Welpe. Predicting elections with twitter: What 140 characters reveal about political sentiment. In *Proceedings of the Fourth International Conference on Weblogs and Social Media, ICWSM 2010, Washington, DC, USA, May 23-26, 2010*, 2010.

[106] Onur Varol, Emilio Ferrara, Christine L Ogan, Filippo Menczer, and Alessandro Flammini. Evolution of online user behavior during a social upheaval. In *Proceedings of the 2014 ACM conference on Web science*, pages 81–90. ACM, 2014.

[107] Gang Wang, Tristan Konolige, Christo Wilson, Xiao Wang, Haitao Zheng, and Ben Y Zhao. You are how you click: Clickstream analysis for sybil detection. In *Presented as part of the 22nd {USENIX} Security Symposium ({USENIX} Security 13)*, pages 241–256, 2013.

[108] Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. Twitterrank: finding topic-sensitive influential twitterers. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 261–270. ACM, 2010.

[109] Yinglian Xie, Fang Yu, Qifa Ke, Martín Abadi, Eliot Gillum, Krish Vitaldevaria, Jason Walter, Junxian Huang, and Zhuoqing Morley Mao. Innocent by association: early recognition of legitimate users. In *the ACM Conference on Computer and Communications Security, CCS'12, Raleigh, NC, USA, October 16-18, 2012*, pages 353–364, 2012.

[110] Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. A biterm topic model for short texts. In *Proceedings of the 22nd international conference on World Wide Web*, pages 1445–1456. ACM, 2013.

[111] Chao Yang, Robert Harkreader, and Guofei Gu. Empirical evaluation and new design for fighting evolving twitter spammers. *IEEE Transactions on Information Forensics and Security*, 8(8):1280–1293, 2013.

[112] Chong-ho Yu and Barbara Ohlund. Threats to validity of research design. *Retrieved January*, 12:2012, 2010.

[113] Eva Zangerle and Günther Specht. Sorry, i was hacked: a classification of compromised twitter accounts. In *Proceedings of the 29th annual acm symposium on applied computing*, pages 587–593. ACM, 2014.

[114] Zizhu Zhang, Weiliang Zhao, Jian Yang, Cecile Paris, and Surya Nepal. Learning influence probabilities and modelling influence diffusion in twitter. In *Companion Proceedings of The 2019 World Wide Web Conference*, pages 1087–1094. ACM, 2019.