Privacy Leakage and the Manipulation of Public Opinion in Online Social Networks

Inauguraldissertation
der Philosophisch-naturwissenschaftlichen Fakultät
der Universität Bern

vorgelegt von

Luca Luceri

von Italien

Leiter der Arbeit:
Professor Dr. Torsten Braun
Institut für Informatik
Professor Dr. Silvia Giordano
University of Applied Sciences and Arts of Southern Switzerland

Privacy Leakage and the Manipulation of Public Opinion in Online Social Networks

Inauguraldissertation
der Philosophisch-naturwissenschaftlichen Fakultät
der Universität Bern

vorgelegt von

Luca Luceri

von Italien

Leiter der Arbeit:
Professor Dr. Torsten Braun
Institut für Informatik
Professor Dr. Silvia Giordano
University of Applied Sciences and Arts of Southern Switzerland

Von der Philosophisch-naturwissenschaftlichen Fakultät angenommen.

Der Dekan:
Bern, March 31, 2020 Prof. Dr. Zoltan Balogh

Copyright Notice

Dedicated to my Family

Acknowledgements

The work here presented was carried out during my doctoral studies at the University of Bern and the University of Applied Sciences and Arts of Southern Switzerland (SUPSI), whose collaboration, under the frame of the Swiss National Science Foundation (SNSF) project "SwissSenseSinergy", allowed me to pursue a Ph.D.. Also, part of this work has been conducted at the Information Science Institute at the University of Southern California (USC). This thesis is the result of many years of work, during which I had the pleasure of meeting and collaborating with amazing researchers around the globe. This achievement would not have been possible without their cooperation and the support of many people to whom I would like to express my gratitude.

First of all, I would like to sincerely thank my supervisors, Prof. Dr. Torsten Braun and Prof. Silvia Giordano, for giving me the opportunity to pursue a Ph.D., guiding me towards this achievement, and allowing me to follow my own ideas.

In particular, I thank Prof. Braun for giving me the chance to join the Communications and Distributed Systems (CDS) group of the University of Bern, and for considering me in every stimulating activities he organized with his group, especially the amazing summer schools. I am also grateful for all the advice, ideas, and corrections he has given me, and which helped me in publishing my work and completing this thesis. The help he gave me to turn my intuitions into rigorous research has been of invaluable importance to me.

I cannot quantify the support of Silvia during my doctoral studies, especially during my most difficult periods. I have to express my deep gratitude to her for giving me inestimable aid in every day of the arduous path of this insidious undertaking, even when I was not behaving correctly or performing at my best. Without the motivation she has given me and the daily encouragement during these years, the way towards the Ph.D. would have been much more difficult. I cannot forget to mention her vision and intuitions, which have been of huge impact on the outcome of this work. I greatly thank her also, and enormously, for gifting me the most wonderful experience of my life: I will always be grateful to her for my working period and experience in US.

Besides my supervisors, my sincere gratitude also goes to Emilio Ferrara, for accepting me at the Information Science Institute at USC, without knowing me at all and just trusting his unfailing flair for *good people*. I have to thank him for all the (countless) things I have learned from him. He has been fundamental in broadening my horizons in research and in looking at problems from a different prospective.

A big thanks goes to all my CDS colleagues, in particular Mostafa, Gaetano, Ali, Jose, and Eirini, for the proficient conversations and advice about the Ph.D. life. An enormous thanks goes to Daniela Schroth, who has always been ready to help me with all the administrative procedures, especially during the completion of my Ph.D.

Many thanks also to my colleagues at SUPSI, Michela, Alessandro, Francesca, Daniele, Kamini, Steven, and Luigi, who were always there to discuss interesting ideas and propose interesting points of view. I would also take the opportunity to thank Roberto and Tiziano, for allowing me to pursue my Ph.D. without any pressure. A special mention goes to my life saver Felipe (and Anto). Among my SUPSI colleagues, I need to mention and thank Davide: My roommate, colleague, and friend (in a pure random order). I cannot summarize our personal and professional relationship in a sentence (another thesis would be needed for that!).

The time spent at USC would not have been the same without Adam, Anna, Ashok, Diana, Emilio, Goran, and Palash. Thanks for the good conversations, the proficient collaboration, and for making my time at USC a wonderful experience.

I would like to express my gratitude to the SNSF, which funded my doctoral studies with the projects “SwissSenseSinergy” and “Uprise-IoT”.

I cannot forget to thank all my friends. In particular, Vito V. Ivana, Mauro, Nigo, Marika, Barron (and his sweet half *C*) you have been fundamental in the moments of need. I cannot quantify the support you gave me and the funny moments spent together.

Enormous gratitude goes to my family in Milan: Michele and Virginia, Andrea and Elena, Pietro and Giwta, Marco, Mauro, Giulio, and Tommy, you have been the best company for the nights and dinners in Milan. Saretta and Babby, there are no words to express the help and support you gave me during these years.

Also, I need to mention my small family in Los Angeles: Amy and Trevor, Maja and Goran, and Chicco thanks for every moment spent together.

Least but not last, I would like to thank my family for all their support. You have been the most important source of encouragement and inspiration during my whole life. Thanks for always sticking by my side, even in the most complicated times.

Bern, March 31, 2020 Luca

Abstract

Online Social Networks (OSNs) are computer-based technologies that enable users to create content, share information, and establish social relationships in online platforms. The advent of OSNs have dramatically revolutionized the way we access the news, share opinion, make business and politics. Although the wide adoption of OSNs brought several positive effects, the combination of its technological and social aspects hides harmful effects for both the individual users and the entire society. Among the potential risks analyzed in the literature (e.g., security, health, etc.), in this thesis, we analyze the perils related to the *privacy* leakage and the *manipulation* of opinions in OSNs. In particular, we investigate the factors driving these perils, with the final objective of raising users' awareness of the risks behind their online activities. We show how, for both the privacy and manipulation perils, social connections play a central role in fostering and exacerbating such issues. In fact, social connections among OSN users result in a network structure, which enables the spreading of information, behaviors, and opinions across the OSN population through online interactions.

Along this research direction, we first explore to what extent an individual's privacy can be violated by leveraging information provided by other users in the OSN. In particular, we examine the problem of location privacy by developing methods to assess users' privacy risks and strategies to control the public exposure of their data. Then, we explore the privacy peril by considering the diffusion of behaviors and opinions in OSNs. In fact, social interactions can substantially affect the extent to which a behavior, an opinion, or a product is adopted by OSN users. This concept is a social phenomenon referred to as *social influence*. According to this concept, we investigate whether social influence modeling (i.e., learning influence strengths among subjects) can be used to accurately predict users' future activity and, in turn, violate their privacy. We present different approaches to model social influence and we show how such models can be employed to violate users' privacy.

Online interactions and social influence play also a crucial role in the manipulation of peoples' belief and opinion. Manipulation campaigns have raised particular concerns in the political context. Bots (i.e., software-controlled accounts) and trolls (i.e., state-sponsored human operators) are the main actors responsible for these campaigns. In this thesis, we analyze the activity of such malicious actors to enhance and enable countermeasures for their detection. More specifically, we first uncover the strategies employed by bots to avoid detection and manipulate human users. Then, we present an approach for detecting trolls' activity in OSNs that accurately identifies troll accounts and unveils their distinguishing behavior with respect to regular users. The results presented in this thesis confirm the privacy and manipulation risks in OSNs: On one hand, we prove that users' privacy is not under individual control as public information can be efficiently used to predict their behavior, and in turn, violate their privacy. On the other hand, we show that malicious actors have become increasingly sophisticated to escape detection and manipulate human users. However, the majority of OSN users are not conscious or underestimate the potential risks behind their online activity. Towards raising users' awareness of such perils and to mitigate this set of open problems, we propose an awareness service, based on a mobile application, to timely communicate users their current risks in OSNs. For this purpose, we deploy a framework to collect users' data in a privacy-preserving way and provide them feedback about their privacy and manipulation risks in real-time.

Keywords: Online Social Network (OSN), Privacy, Social influence, OSN manipulation.

Contents

List of Figures

List of Acronyms

AI Artificial Intelligence.

AUC Area Under the Curve.

DNN Deep Neural Network.

eWOM electronic Word of Mouth.

IC Independent Cascade.

IRL Inverse Reinforcement Learning.

LBS Location-Based Service.

LSTM Long Short-Term Memory.

LT Linear Threshold.

MAD Median Absolute Deviation.

MAE Mean Absolute Error.

MCS Mobile Crowd Sensing.

MDP Markov Decision Process.

MSE Mean Squared Error.

OSN Online Social Network.

RL Reinforcement Learning.

RNN Recurrent Neural Network.

RQ Research Question.

SIDL Social Influence Deep Learning.

SM Saliency Mask.

SSR Smallest Sufficient Region.

TNR True Negative Rate.

TPR True Positive Rate.

VHB VIVO Human Behavior.

WWW World Wide Web.

XAI eXplainable Artificial Intelligence.

1 Introduction

1.1 Background and Motivation

The advances of mobile and wireless technologies have given birth to a new era of human communication, where traditional contacts has been replaced by mobile and wireless encounters [112]. More recently, with the growing accessibility of the Internet, we have assisted to the rise of Online Social Networks (OSNs). In 1997, a generic OSN was defined as set of people or organization connected through a computer network [98]. Since then, OSNs have gained tremendous popularity worldwide. Thanks to the emergence of mobile device (e.g., smartphone) technology, this rise has sped up exponentially. Just as an example, as of July 2019, Twitter and Facebook count around 330 and 2,414 million active users, respectively[1], and Wechat, the Chinese OSN, reached over 1.5 billion mobile subscription[2].

Nowadays, OSNs do not only indicate a mere virtual connection among individuals. They represent the most relevant and influential combination of technological and

[1] https://www.statista.com/

[2] http://wearesocial.cn/digital-2019-china/

social paradigms. From the technical side, OSNs are interactive computer-mediated technologies that facilitate the creation and sharing of information, ideas, interests, and other forms of (personal) expression with virtual friends and online communities. The quantity and quality of data that users share in OSNs have a tremendous economic impact, which also determines the market value of an OSN. In fact, the availability of large amounts of data allows OSN providers to offer services that are increasingly-tailored towards the particular characteristics of each user. From the social side, OSNs are social structures resulting from a set of social actors, people or other entities, and the social interactions among them. This dual nature enabled the success and the establishment of one of the most disruptive communication platforms of the last fifteen years with high social, economic, and political value.

The rise of OSNs has dramatically transformed our society by revolutionizing the way we communicate, socialize, make business, and many other activities. Initially, OSNs have enabled social interactions that would be limited by physical constraints [109] and they facilitated the interplay between users with similar interests [14]. For example, OSNs allow users to create and subscribe to groups that are focused on specific topics, interests, or hobbies. During the last decade, OSNs have also changed marketing strategies by transforming the way most businesses operate and advertise their products. Political communication is no exception, as it has moved from the real (offline) world to the digital (online) one. OSNs play a central role in today's politics representing a fundamental asset for propaganda. In this regards, OSNs offer an exceptional channel to broadcast messages and information: They permit information spreading at a huge scale by offering online platforms where news can reach millions of people in a fraction of minute.

As a matter of fact, the complexity of today's online ecosystem captures the facets of the modern information society: Accessing the news, sharing opinions, and entertaining social connections are just a few examples of the variety of engagements that individuals regularly perform online. Such engagements appear harmless and do not require any explicit cost, as most OSNs are free. However, the online ecosystem, because of its dual technological and social nature, hides pitfalls still largely unknown to the final users, which may seriously impact their life and our entire society. Looking at the existing literature on this theme [6], [206], [34], [42], [82], [90], [229], we can classify these perils in the following five risk categories:

- *Security*: The major threats in this category are related to personal profile hacking, phishing (i.e., deceiving users to steal confidential data), ransomware (e.g., threatening users to publish their data or block the access to their system unless a ransom is paid), and fake links that aim to attain personal information.

- *Manipulation*: A phenomenon that has arisen in the last decade is related to the manipulation of public opinion over OSNs. The powerful influence that peers have on each other has been largely used to manipulate peoples' belief and opinion (from politics to finance) and, thus, poses a critical threat to public life. Also, the fact that the information diffusion in OSNs is really fast represents also a huge drawback as false information can easily and quickly spread across the online population, thus, empowering misinformation diffusion, which in turn contribute to the manipulation of public opinion.

- *Privacy*: By publicly sharing personal data and various contents, users' privacy is at risk. In fact, any third party interested in advertising products or manipulating users' belief can utilize such data for profiling and predicting users' interest, opinion, and future behavior. Also, users that share personal information in OSNs can be subject to crimes such as identity theft and stalking.

- *Safety*: Contents shared in OSNs might not be appropriate for kids, who can have access to a large variety of dangerous material, e.g., obscenity. Additionally, bullying and harassment have been largely diffused in OSN (a phenomenon referred to as cyberbullying) eliciting issues such as increased suicidal ideation, lower self-esteem, and a wide range of negative emotional responses (frustration, depression, anxiety, etc.)

- *Health*: Addiction, depression, sleep disorders, and cognitive absorption are just a few of the several health issues that the frequency of OSNs usage has on its users.

For every category, it is clear how both the technical facets of OSNs and the improper usage of their users contribute to such risks [85, 209], which can impact people personal and collective life. Motivated by these significant perils, and by the users' unconsciousness of such problems, in this thesis, we aim to explore the nature of OSN abuse and investigate both the users' misuse of online platforms and the OSN technical vulnerabilities that enable such risks. In particular, we examine the perils related to the *privacy leakage* and the *manipulation of opinions* in OSNs. In the next Section, we detail our effort towards such objectives and we describe the problems faced in this thesis.

1.2 Problem Overview

In this Section, we provide an overview of the problems faced in this thesis and we detail how they are linked to each other. As mentioned in Section 1.1, our objective is to investigate the factors driving the perils related to the *privacy leakage* and the *manipulation of opinions* in OSNs. These issues affect OSN users at a different granularity. While the former has repercussions at the individual level (e.g., the violation of personal information), the latter concerns our society as a whole (e.g., the risk of influencing the mass in voting events). Although these two perils appear to be disconnected, they share a significant feature. In both cases, social relationships and interactions play a significant role in fostering and exacerbating such risks. For instance, the fact that users more likely interact with and connect to people similar to them (e.g., with the same interests or opinions) [92, 172] may reveal users' undisclosed personal information and/or expose them to manipulation attempts. The most remarkable and recent example that encompasses both these issues is represented by the political scandal of Cambridge Analytica, a data analytics firm that worked with Donald Trump's election team and in the Brexit campaign. This company used Facebook users' personal information to build a system that could profile US voters, in order to target them with personalized political advertisements and influence their choices in voting events [47]. The personal data of about 87 millions Facebook users were acquired via the 270,000 Facebook users who used a third-party application for academic research. By giving this application permission to acquire their data, users also shared information about their friends. This resulted in the data of about 87 million users, the majority of whom did not consent to release their data [47]. This fact shows that although the privacy and manipulation perils affect users at a different granularity, their emergence involves the whole OSN community and, in turn, our society [97].

Both these issues are inherently related to the structure and functionalities of OSNs. Differently from the World Wide Web (WWW), which is mainly structured around content, OSNs are organized around individuals, i.e., OSN users. Connections among such users results in a network, which provides a means to instantiate (and maintain)

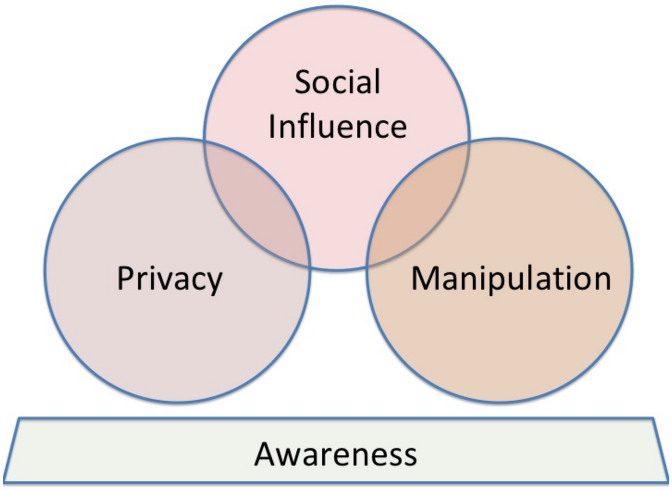


Figure 1.1: Overview of the four *sub-problems* faced in this thesis

social relationships, connect to users with similar interests, and discover content generated by others [176]. The network structures of OSNs foster the spreading of information and behaviors across the online population [52, 63, 237]. Considerable research has focused on the diffusion of information in OSNs, which is also referred to as electronic Word of Mouth (eWOM), demonstrating that eWOM plays a crucial role in influencing people's behavior [43, 103, 132]. The concept that the interplay among individuals may affect their behavior is a social phenomenon referred to as **social influence**, which is acknowledged as a fundamental factor governing human behavior. As a result, the network structure, in terms of who is connected to whom, can substantially affect the extent to which a behavior, an opinion, or a product, are adopted by individuals and diffuse across the online population [50]. Therefore, the understanding of how users influence each other can have an impact both on the privacy abuse (e.g., to predict users' interests, opinions, or activities) and on the manipulation of the public opinion (e.g., to maximize the spread of fake news and to target influential or impressionable users). For these reasons, social influence plays a relevant role in both the perils analyzed in this work and, thus, holds a central position in the discussion of this thesis, as also represented in Fig. 1.1. In particular, Fig. 1.1 shows an overview of the four *sub-problems* faced in this study and how they are interconnected among each other. The concept of *social influence* bonds the **privacy** and **manipulation** issues (as we explained above), while the sub-problem named **awareness** embraces these three sub-problems with the objective of increasing users' knowledge and understanding of the risks behind their online activities.

In the next Section, we detail the objectives of this thesis and the specific problems we aim to tackle throughout this work.

1.3 Problem Statement

As we described in Section 1.2, the network-oriented nature of OSNs, along with the tremendous volume of data released on online platforms, poses significant risks for both the individual users and the entire society. Therefore, an in-depth understanding of OSNs structure and functionality is necessary to evaluate the impact of users' online activity on such risks. In particular, in this thesis, we aim to investigate privacy and manipulation issues in OSNs with the final goal of raising users' awareness of such perils. As mentioned in Section 1.2, we divide this objective in four sub-problems. In the following subsections, we provide a detailed description of the Research Questions (RQs) we seek to answer for each sub-problem of this thesis.

1.3.1 Boundaries of Users' Privacy

OSN users can publish a variety of content (e.g., opinions, news, videos, music, etc.) and disclose information about themselves (e.g., age, sex, interests, residential address) in their profile. By flooding online platforms with these data, users leave a *digital trace* that, if properly analyzed, can provide very detailed information about them. The availability and the amount of such personal data in OSNs is of paramount importance in today's data-driven economy. In fact, such information can be exploited by a third party (e.g., a marketing company) that uses the OSN platforms to perform targeted advertisement and offer increasingly-tailored services to the users.

This fact raises inherent privacy issues that users can address by limiting the amount of content they share. However, this approach does not totally protect users from the disclosure of their personal data. Although OSN providers include in their platforms several privacy-preserving strategies (e.g., to restrict the access to the published contents to some users only), it has been shown [155] that such strategies are not fully effective in the protection of personal data in most popular OSNs.

Sensitive information about a given person can still be obtained from data released by other users within the OSN [22, 155]. In this respect, social cues (e.g., the strength of social connections and similarity patterns between users) are widely regarded as one of the main causes of privacy vulnerability in OSNs. This is due to the fact that users more likely interact with and connect to people similar to them (e.g., with the same interests, who visited common locations, etc...) [92, 172]. OSN functionality, in turn, amplifies this issue by supporting public messages and interactions among users. For instance, a user can be mentioned by others in relation to a given topic, which may unintentionally reveal interests for that subject. Hence, users are not in full control of the public exposure of their personal information and their privacy is not bounded by what they deliberately share [97].

Along this research direction, the goal of the privacy leakage sub-problem is to assess the vulnerability of users' personal information and provide them with strategies to control the level of their privacy. For this purpose, we aim to respond to the following RQs:

RQ 1.1: *Can an attacker infer users' personal information from publicly-available data of other OSN users?*

RQ 1.2: *Can we provide users with countermeasures to control the public exposure of their data?*

RQ 1.3: *Can we allow users to measure the impact of their characteristics, behavior, and online activity on their privacy?*

In Section 1.4.1, we summarize our efforts to answer the above RQs and we detail our contribution.

1.3.2 The Social Influence Phenomenon

Social relationships and interactions are widely recognized as a means to diffuse behaviors and information [50, 63]. As a result, the network structure of OSNs can critically influence the decision of a user to adopt a behavior, believe an information, or alter an opinion [51, 52]. For these reasons, the social influence phenomenon is considered a crucial factor governing human behavior. Although social interactions among OSN users occur online, social influence underlies real life spreading phenomena, such as the diffusion of opinions and the adoption of products, with inevitable repercussions on marketing, politics, health, and business. For this reason, various applications (e.g., viral marketing) rely upon the modeling of social influence among

OSN users (i.e., measuring the extent to which users influence each other). In fact, this analysis can provide useful indication to predict users' future interests, opinions, and activities. As a consequence, as we mentioned in Section 1.2, the manipulation of public opinion and the privacy leakage are issues inherently related to the social influence phenomenon.

Based on this discussion, the objective of this sub-problem is to model social influence among OSN users for measuring to what extent they are affected by others' activity and, accordingly, forecasting their future behavior. In particular, we aim to investigate whether social influence modeling can be used to predict users' future activities and, in turn, to violate their privacy. As we are interested in measuring whether and to what extent an individual is influenced by other subjects, we focus on social influence at the user-level. The latter provides, for each subject, prediction of a given spreading process (e.g., adoption of a product) based on others' behavior. Existing models [106, 208] have two main drawbacks, detailed in Chapter 2, which we aim to overcome by proposing a novel approach. Also, existing approaches rely only on direct social relationships among users (e.g., friendship) without considering other factors that might impact users' future decisions and activities (e.g., their location, interests, etc).

In summary, the sub-problem related to social influence aims to understand whether social influence modeling can be used to violate users' privacy and seeks to answer the following RQs:

RQ 2.1: *How is it possible to overcome the limitations of existing social influence models?*

RQ 2.2: *Can other factors (e.g., location) impact influence modeling other than social relationships?*

In Section 1.4.2, we describe our contribution and the proposed solutions to answer these RQs.

1.3.3 Manipulation of the Public Opinion

As we mentioned in Section 1.2, the issue related to the manipulation of public opinion leverages the social influence phenomenon to affect and alter peoples' belief. Various studies raised awareness about the risk of mass manipulation of public opinion, especially in the context of political discussion [174, 220, 234]. In particular, the massive diffusion of digital misinformation and the increasing presence of malicious accounts in OSNs have been identified as major threats to democracies, other than main factors contributing to OSN manipulation [82]. Misinformation denotes the spread of low-credibility content, such as false or misleading news reports, hoaxes, conspiracy theories, click-bait headlines, junk science [213]. Malicious actors, e.g., foreign agents or fake automated accounts, are OSN users that embed themselves in online social systems and interact with their users with the objective of influencing and manipulating the public opinion. Previous works found evidence that these malicious actors play a disproportionate role in manipulation and misinformation campaigns globally [130, 200, 219]. In turn, OSN users are vulnerable to this manipulation, resharing content posted by and interacting with malicious accounts [213].

The 2016 Brexit referendum and the 2016 US Presidential election represent recent remarkable examples of social media political manipulation, as both of them spotlighted a massive presence of malicious users and false information in OSNs (recognized by the research community, OSN providers, and government agencies) [9, 36, 71, 131]. Since then, OSN service providers have been increasing their efforts to suspend malicious actors and maintain a healthy conversation on their platforms. However, malicious activity on social media has not entirely stopped: social media bots (i.e., automated and software-controlled accounts [86]) and trolls (i.e., state-sponsored human operators [20]) are still active [70, 134, 167]. These malicious actors keep evolving and changing their strategies to escape detection and the resulting suspension from OSN platforms. Therefore, detection of coordinated campaigns is an open challenge for the research community [57, 87, 232]. While researchers offered different approaches for the detection of bots, the automated identification of troll accounts has been proven to be a challenging (yet unsolved) task. However, as the strategies of bots have been becoming increasingly sophisticated, it is of paramount importance to keep the pace of such malicious accounts in order to build and adapt effective countermeasures for their detection.

In this thesis, we propose to investigate the online activity of malicious actors in OSNs for enabling the detection of such accounts and curbing their manipulation attempts. Towards the objective of fighting the sub-problem related to the manipulation of public opinion in OSNs, we aim to respond to the following RQs:

RQ 3.1: *How are social bots evolving to mimic human behavior and avoid detection?*

RQ 3.2: *What are the strategies implemented by bots to manipulate OSN users and are those strategies effective?*

RQ 3.3: *In a analogous way to the detection of bots, is it possible to implement an automated approach for the identification of troll accounts in OSNs?*

In Section 1.4.3, we detail our work to answer the above RQs and tackle these challenges.

1.3.4 Challenges to Raise Users' Awareness of OSN Perils

From the discussion in Section 1.2, it is evident how both technical and user vulnerabilities contribute to privacy and manipulation issues. Whereas such perils are concrete and have significant effects on personal and collective lives, OSN users are not aware of or underestimate the potential risks behind their online activity. Moreover, there is a lack of countermeasures to timely communicate users their current risks in a clear and comprehensible way. However, building a service with such purpose presents different challenges, especially in the context of applications running onto mobile devices, which nowadays represent the main means to access OSNs [128]. The collection of users data, for example, is an operation that introduces additional privacy and security issues.

In the context of the SwissSenseSinergy Project[3], we realized that these issues especially affect application developers that aim to distribute crowd-sensing applications. On one hand, there is a need to provide a service (e.g., a location-based service). On the other hand, privacy and security of user data should be guaranteed (e.g., protect their location information). However, application developers do not always have the will and/or the skills to implement secure and privacy-preserving applications.

To tackle these challenges, this sub-problem aims to respond to the following RQ:

RQ 4: *Can we develop a secure and privacy-preserving service for assessing and communicating users' their privacy and manipulation risks in OSNs?*

In Section 1.4.4, we summarize our solution to answer RQ 4.

[3] http://www.swiss-sense-synergy.ch

1.4 Thesis Contributions

As mentioned in Section 1.3, the final goal of this thesis is to raise users' awareness of the abuse of OSNs. In particular, we focus on the perils related to the privacy leakages and manipulation of opinions in OSNs. To increase users' consciousness of such risks, it is needed to investigate these issues, assess their significance, and evaluate the factors impacting on their development. Therefore, in the following subsections, we first detail our contributions related to the sub-problems, and corresponding RQs, described in Section 1.3.1, Section 1.3.2, and Section 1.3.3. Finally, we introduce our proposed solution to raise users' awareness of OSNs abuse and tackle the challenges presented in Section 1.3.4.

1.4.1 Privacy Measurement and Control

The first contribution of this thesis is related to the exploration of the privacy issue in OSNs. Among all the sharable personal data, we consider the geographical location as it is considered both a highly-valuable and a very sensitive information. For example, user positions are often required for effective delivery of Location-Based Services (LBSs). However, to protect their privacy, users rarely reveal their location in OSNs (operation referred to as *geo-tagging*) [108]. Nonetheless, third parties that offer a LBS might have alternative strategies to obtain users' location (e.g., by implementing either an illicit or licit strategy to capture the geographical position of a certain user) and, thus, violate their privacy.

In this thesis, we explore the problem of geo-location privacy on Twitter. In accordance with the RQs discussed in Section 1.3.1, we present methodologies to measure the level of users' geo-location privacy and to control the public exposure of their sensitive information [163]. Specifically, to respond to RQ 1.1, we first assess the ability of an attacker to correctly estimate a target user's location by leveraging the information shared by other OSN users. We propose a novel deep learning architecture that can accurately infer users' location from a set of publicly-available geo-tags. We show that a deep learning approach is needed to model complex social relationships among OSN users. The obtained results confirm the serious concerns about location privacy in OSNs and motivate us to evaluate countermeasures applicable by users [101]. Thereby, to address RQ 1.2, we investigate the effectiveness of two data perturbation techniques that users can employ to control the public exposure of their geo-tags and, in this way, improve privacy. Finally, to shed light on the factors influencing privacy and to answer RQ 1.3, we propose a model that measures privacy based on several users' features (e.g., social and behavioral characteristics) and allows to quantify the impact that each feature has on privacy.

1.4.2 Social Influence Modeling

In line with the idea of exploring privacy leakages in OSNs, in the next contribution, we consider to examine the social influence phenomenon. In particular, we aim to model social influence among users to measure the level of influence they are subject to and, accordingly, predict their future activity (e.g., attending a social event).

To address RQ 2.1, we introduce Social Influence Deep Learning (SIDL) [164, 165], an approach based on Deep Neural Networks (DNNs), which learns influence strength in dyadic social connections from the history of users' activity and, accordingly, predict their future behavior. We show that an approach based on deep learning is suitable to overcome the limitations of state-of-the-art approaches. We then propose a step forward with respect to existing solutions in the literature. While previous approaches only consider direct social relationships among users (e.g., friends, relatives, etc.) to model social influence, we aim to investigate other factors that link users and might affect their behavior. In particular, we examine the collective influence that groups (or communities) of people can exert on an individual. Therefore, to respond to RQ 2.2, we present a novel interpretation of communities as sources of social influence by considering different factors impacting people behavior (i.e., the geographical area they live in, their interests, and their social ties) [162, 170]. Our results validate the idea of using social influence for predicting human behavior and, at the same time, raises further privacy concerns: Although users may not disclose information about their activity, we showed that their privacy is not only in their hands, as social influence modeling can be efficiently used to predict their behavior, and in turn, violate their privacy.

1.4.3 Detection of Manipulation Campaigns

In Section 1.3.3, we discussed the relevant perils of mass manipulation of the public opinion. In the next contribution, we analyze the activity of OSN users that play a pivotal role in manipulation campaigns, i.e., social bots and trolls, to improve their detection and adopt effective countermeasures. To respond to RQs 3.1 and 3.2, we first study the behavior of bot accounts that keep escaping detection and continuously act in OSNs. For this purpose, we explore the evolution of social bots during the last two US elections, i.e., the 2016 Presidential election and the 2018 Midterms [167, 168]. We examine the strategies implemented by bots to avoid the suspension from OSNs (RQ 3.1) and to manipulate the opinion of OSN users (RQ 3.2). Also, we analyze how human users deal with these automated accounts as it is of paramount importance to understand how humans handle the manipulation attempts. The results of our analysis reveal the effectiveness and the mutable nature of such increasingly sophisticated bot accounts. Our insights can inform actionable policies to detect social bots and fight online abuse.

Finally, as we detailed in Section 1.3.3, differently from bots, the automated identification of troll accounts is an open challenge for the research community. To address this challenge and answer RQ 3.3, we examine the online activity of Russian trolls during the 2016 US Presidential election and we propose a novel approach based on Inverse Reinforcement Learning (IRL) to capture troll behavior and identify troll accounts in OSNs [169]. We employ IRL to infer a set of online incentives that may steer user behavior, which in turn highlights behavioral differences between troll and non-troll accounts, enabling their accurate classification.

1.4.4 Awareness of OSN Perils

The final objective of this thesis is to increase users' awareness of OSN perils. For this purpose, we consider to implement a service to analyze and communicate to OSN users their current risks in real-time. We envision this service to be offered via a mobile application, since nowadays smartphones represent the main means to utilize OSNs and share sensitive data (e.g., location) in online platforms [128]. However, as highlighted in Section 1.3.4, this objective presents different challenges to be addressed.

To tackle these challenges and respond to RQ 4, we develop a crowd-sensing framework, called VIVO, which allows to collect data from mobile devices in a secure and privacy-preserving way [166], while enabling direct communication with mobile users. VIVO has a broader objective of gathering crowd-sensed data (e.g., from smartphone sensors) in real-time and supporting application development and testing. The fact that we can also rely on data collected by mobile sensors (e.g., location), but not published in OSNs, is an asset to our analysis as this allows us to extend and validate the models built relying only on OSN data to real life (offline) information. For example, we can evaluate the privacy of the location of OSN users also when they do not share their position on online platforms.

Overall, the VIVO framework is beneficial for the purpose of this thesis for three main reasons: (i) it provides a privacy-by-design facility for application developers, (ii) it allows us to deploy an application that collects heterogeneous data (from different sources, not only related to OSNs) in real-time, and (iii) it permits to directly communicate with the users, thus, enabling a mechanism of direct feedback of the risks behind their online activity (e.g., privacy leakage in an OSN).

1.5 Thesis Outline

The remainder of this thesis is organized as follows.

Chapter 2 provides an overview of the existing works in the context of privacy, social influence modeling, manipulation of public opinion, and awareness services. We also present some fundamentals on machine learning and network science, which represent relevant tools for our approaches, and we provide an overview of the OSNs analyzed in this thesis.

Chapter 3 shows a study case on the privacy risks in OSNs by analyzing the problem of geo-location privacy on Twitter. In particular, we present methods to assess the ability of an attacker to correctly infer users' locations and we propose strategies that users can adopt to measure and control their privacy.

Chapter 4 is devoted to social influence models for predicting human behavior. We present the SIDL approaches and a novel interpretation of communities as collective sources of social influence. Moreover, Chapter 4 discusses the relation between privacy and social influence modeling.

Chapter 5 explores the issue of the manipulation of public opinion in OSNs. In particular, we investigate the activity of the actors responsible for manipulation campaigns, i.e., bots and trolls, providing insights and proposing approaches for their automated detection.

Chapter 6 presents the VIVO framework, details its architecture, and describes the application we developed to raise users' awareness on OSN risks.

Chapter 7 concludes this thesis, discusses our findings, and presents future challenges for OSN users.

2

State of the Art

2.1 Overview

OSNs have rapidly grown from a clique aggregation channel to a global phenomenon that is, nowadays, responsible for a significant fraction of overall Internet browsing and user online engagement [146]. This has elicited a paradigm shift of our society, by transforming the way people communicate, interact, and socialize. These changes do not only concern social relationships. Politics, marketing, and advertisement are just a few of the several other activities that today take place in OSNs, such as Facebook, Twitter, Instagram, WeChat, etc.

For this reason, the advent of OSNs, and their intrinsic multi-relational data, has also offered the opportunity to study the dynamics of social relationships and human behavior at a huge scale. The availability and quality of OSN data on such dynamics allow researchers to experiment and validate offline studies (whose scalability was limited) at a low cost. An example is related to the “small world” phenomenon studied by Milgram in 1967 [175], and further identified in modern techno-social systems [4,237], according to which any two individuals on the planet are connected through a chain of no more than six intermediate acquaintances.

In this regard, an in-depth analysis of the graph structure of OSNs can lead to an understanding of individual and network features. The scientific discipline that studies the structure and properties of *complex* networks (i.e., social networks, telecommunication networks, biological networks, etc.) is referred to as *network science*, which is detailed in Section 2.7. For example, network science can be used to trace the flow of information among users, analyze social relationships, predict spreading phenomena, or detect influential users. Although this kind of investigation might appear harmless, and aimed only at describing network structures, it might be used for malicious purposes. For instance, the analysis of users' relationships and similarities can be used to reveal their personal information and violate their privacy [97]. A case in point is the Cambridge Analytica political scandal, where millions of Facebook users' data have been used (without their consent) to profile US voters in order to target them with personalized political advertisement and influence their voting choices [47]. Additionally, as OSNs have proven as effective tools to influence individuals' opinions and behaviors [14,15,24], the understanding of how influence spread among users can be beneficial for manipulating their opinion, understanding their belief, and predict their future behavior.

Along these research lines, in the next sections, we analyze the sub-problems (introduced in Chapter 1) related to privacy leakage, social influence, manipulation of opinions in OSNs, and users' awareness of OSN perils. Finally, we review the basics of the methodologies exploited in this thesis along with an overview of the OSNs considered in our analysis.

2.2 Geo-Location Privacy Leakage in OSNs

The unprecedented amount of personal information available in OSNs is of paramount importance in today's data-driven economy. Contextually, there is an increasing concern on the ability of users to effectively hide the personal information they are not willing to expose [247]. OSN providers include several privacy-preserving strategies in their platforms (e.g., to restrict access to the published content to some users only). However, it has been shown [155] that such strategies are not fully effective in the protection of personal data in the most popular OSNs. In this respect, social cues (e.g., the strength of social connections and similarity patterns between users) are widely regarded as one of the main causes of privacy vulnerability in OSNs. For example, it is shown that public data in OSNs can be effectively used to infer users' personal information and to predict future users' activities [17, 22, 26, 53, 144, 192, 249]. Similarly to these studies, this thesis shows how OSN users are not in full control of their privacy [97], as sensitive information can be obtained by analyzing other users' data. In particular, we introduce a deep-learning approach specifically targeted at the violation of users' location privacy from publicly-available data. We perform our analysis by considering the Twitter OSN as a study case. Therefore, in Section 2.2.1, we examine the role of location on Twitter and we compare our approach to existing solutions in the literature. Moreover, we propose a model that allows us to assess the extent to which several factors affect the privacy of users, therefore enabling its proper tuning and control. Along this research direction, in Section 2.2.2, we provide an overview of existing works that aim to provide privacy control for OSN users and we highlight their differences with respect to our proposed approach.

2.2.1 Geo-location Privacy on Twitter

Being Twitter one of the most used OSNs, the importance of both sharing and protecting location information on its platform is widely-recognized [250]. For example, various applications for emergency detection [13, 157], health monitoring [58], and event recommendations [186, 243] are based on the location information shared on Twitter. Recently, large efforts have been dedicated to the development of tools to perform location inference on Twitter. According to [250], location on Twitter can be of three main types: *home location, mentioned location,* and *tweet location.* The first one represents a user's long-term residential address, which may be published at several levels of granularity (e.g., city or village) in the user profile. The second refers to the locations that a user mentions in the text of her/his tweets. The third is the geo-tag that a user may publish as a meta-data attached to her/his tweets. The decision to either provide a geo-tag or not is done for each published tweet. On average, 1% of tweets are published with a geo-tag [108]. As described in [250], tweet location can be uncovered by relying on multiple sources of information: (i) tweet content [78, 142, 243], (ii) Twitter social network [207], and (iii) Twitter contextual information [61, 212] (i.e., meta-data related to both tweets and users' profiles).

In this thesis, we propose a novel deep learning architecture for unveiling users' geo-location based only on social network information. The proposed approach aims to infer the geo-tag of a generic user's tweet by only leveraging the geo-tags shared by other users on Twitter. The rationale is to investigate whether OSN users can effectively hide their location information. This intuition takes inspiration from [207], where a geo-tag published by a user is inferred considering information of her/his friends on Twitter (i.e., friends' geo-tag and the time when tweets are published). In [207], nearby locations are merged into a cluster and the location inference is framed as a classification task, where the objective is to maximize the classification accuracy. In this approach, the classification error does not carry information on the geographical distance between the target and estimated clusters. Differently from [207], in our work, we measure privacy as the geographical distance between estimated and actual locations and we frame the geo-location inference as a regression problem. The objective of this regression is the minimization of the aforementioned distance, i.e., the privacy of a user. Also, to the best of our knowledge, this is the first approach that attempts to assess users' location privacy based only on the locations shared by other OSN users.

2.2.2 Control of Geo-location Privacy

Although OSN users can employ privacy-control mechanisms (e.g., by allowing only their friends to view the content they share) to protect their sensitive information, location leakage in OSNs remains an open problem [155]. For example, Polakis et. al [194] identify several vulnerabilities related to location privacy in Facebook and Foursquare and propose a set of guidelines to limit them. A theoretical framework to evaluate privacy-preserving strategies against various types of attacks in LBSs has been proposed by Shokri et al. [214]. The most adopted technique to protect the location is based on the perturbation of the location. For instance, [32, 114, 127] propose to reduce the spatial and temporal resolution of location traces to protect users' anonymity. In this thesis, we apply similar data perturbation techniques on the privacy of Twitter users. In particular, we quantitatively measure the impact on users' privacy caused by the obfuscation and by the reduction of the shared geo-tags.

The topic of privacy control has recently gained attention. In [33], the authors develop a method to measure the level of users' anonymity in LBSs and propose countermeasures to increase their privacy. Baron et al. [29] propose a framework to assess the likelihood that the public exposure of a location leads to the leakages of personal information (e.g., political view). By using this framework, users can evaluate their vulnerability to privacy attacks and understand the factors behind it. Our work shares with these previous studies the objective of giving users methods to measure and control their privacy. But specifically, and differently from [29,33], we focus on protecting users' location information.

2.3 The Social Influence Phenomenon

Understanding how users influence each other can be beneficial to various applications, either favorable or harmful towards the final user. In this thesis, we focus on malicious exploitations of such social phenomenon. We aim to model social influence among users for measuring to what extent they are affected by others and, accordingly, inferring their future activity. For instance, this measure of influence can be used to disclose individuals that can be easily affected and manipulated, or be employed to predict human behavior and, in turn, violate users' privacy.

Being a driving factor in human behavior, a considerable amount of work has been conducted to investigate social influence and analyze its effects [14,24,147]. In [216] and [10], the authors propose how to qualitatively measure the existence of social influence, whereas in [65] the correlation between people similarity and influence is examined. Along with these qualitative studies, complementary research efforts have focused on developing predictive models of diffusion processes [117]. This broad research area presents two classes of social influence modeling. We can distinguish macro- and micro-level models according to the outcome granularity. While the former class [171,181] focuses on predicting the result of a diffusion process at the network level (e.g., number of adopters, spreaders, or infected individuals overall the network), the latter aims to study social influence at the user-level, providing prediction of a given spreading process for each subject [106,208]. In Section 2.3.1, we motivate the usage of a micro-level model for our purposes and we compare our proposed approach to other solutions in the literature. In Section 2.3.2, we describe the limitations of existing solutions and we introduce the rationale of our approach to overcome such limitations.

2.3.1 Micro-Level Social Influence Modeling

In this thesis, we focus on social influence at the user-level as we are interested in measuring whether and to what extent an individual is influenced by other subjects. More specifically, we leverage on dyadic social interactions between subjects to predict their behavior. Such micro-level analysis suits several real-life applications, such as targeted advertising, recommendation, and viral marketing. In particular, viral marketing is a convenient example of how to exploit social influence to maximize the adoption of a product or, more in general, the information spread. Although influence maximization [73, 139, 141, 205] is not the target of this thesis, the seminal models presented in [139] provide the underpinning of multiple existing approaches to model diffusion processes in social networks. The models proposed by Kempe et al. [139], referred to as Independent Cascade (IC) model and Linear Threshold (LT) model, map a spreading process to every single node of an underlying graph. In the IC model, each subject independently influences her/his friends with given influence probabilities (the power to influence neighbors [106]), while in the LT model, a subject is influenced by her/his friends if the combination of their total influence probabilities exceeds a threshold. Both models assume to have as input a social network whose edges are weighted by a measure of influence probability. However, these values are not known in practice and, thus, they need to be estimated.

Many efforts have been made to quantitatively measure the influence probability between pairs of friends [80, 106, 113, 159, 208, 225]. Existing works explored different forms of social influence. In [225], the authors proposed a graphical probabilistic model to measure influence strength. In their approach, they analyze influence propagation at the topic-level with the objective of learning influence probabilities with respect to given topics (e.g., politics). Similarly, Gruhl et al. [113] characterize information diffusion by tracking topics and individuals across different blogs. More recently, Zhang et al. [245, 246] studied social influence on Twitter and introduced the concept of social influence locality on users' retweet behavior. They proposed two approaches, i.e., a logistic regression classifier and a factor graph model, based on social influence locality and three kinds of hand-crafted features. On the other hand, other approaches [106, 208] offer more general models (topic and domain independent) by leveraging only the history of the actions performed by each subject in the OSN. These works focused on online actions, such as following, grouping, voting, tagging, etc. In particular, Goyal et al. [106] rely on an instance of the LT model introducing different metrics to estimate the pairwise influence between two individuals. In [208], authors focused on the IC model and employ the Expectation-Maximization (EM) algorithm to estimate the influence probability associated with each edge. Similarly to [106, 208] (details on the differences of these approaches with respect to the proposed solution are provided in Section 2.3.2), in this thesis, we propose a model that does not require any specific knowledge of the domain under analysis and can be generalized to every kind of human activity, both online (e.g., re-sharing a content) and offline (e.g., attending a social event). More specifically, we aim at learning influence strengths among subjects by leveraging the actions performed by users in their history and how such actions propagated between each other. In fact, our approach takes as input only the *raw data* related to users' actions in OSNs (i.e., the action log) and, thus, does not require any hand-crafted features, which in turn may depend on the specific OSN and on the availability of metadata. Although the models suggested in [106, 208] can be generalized to various contexts, they have some limitations, which we present in the next subsection.

2.3.2 Limitation of Social Influence Models

In this subsection, we first define the variables and formulation of existing social influence models. Then, we describe their limitations with an illustrative example and we introduce the rationale of our approach to overcome these drawbacks.

Let u_i be a generic OSN user and A be the set of the actions performed by all the users in the OSN. For each action $a \in A$, each user is either *active*, if she/he has performed the action, or *inactive*, otherwise. We denote $S_{u_i,a}$ as the set of active friends of u_i for the action a. Existing approaches aim to predict whether a subject becomes active based on her/his active friends. To achieve this purpose, they determine the probability of activation $p_{u_i}(S_{u_i,a})$, i.e., the probability that subject u_i becomes active based on the active friends $S_{u_i,a}$, by exploiting the history of u_i actions. The main assumption in these works is that the probability of various friends influencing u_i are independent of each other. Thereby, the activation probability $p_{u_i}(S_{u_i,a})$ is computed as

$$p_{u_i}(S_{u_i,a}) = 1 - \prod_{u_j \in S_{u_i,a}} (1 - p_{u_j,u_i}), \tag{2.1}$$

where p_{u_j,u_i} is the influence probability (strength) of u_j on u_i.

Previous solutions [106, 208] learn the probability p_{u_j,u_i}, from the actions performed by both u_j and u_i. In particular, they consider u_i as influenced by u_j if the latter per-

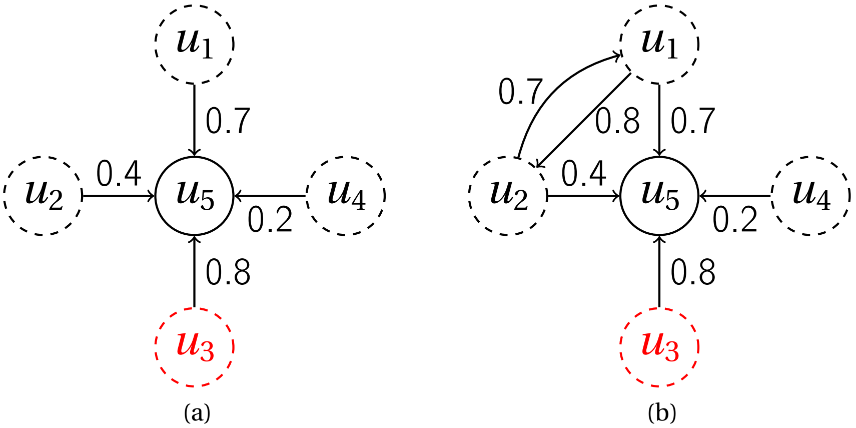


Figure 2.1: Example of influence probabilities in the social network of user u_5

formed the action before the former. These approaches differ from each other for the way the probabilities p_{u_j,u_i} are estimated. In the LT models of Goyal et al. [107], a node becomes active if $p_{u_i}(S_{u_i,a}) \geq \theta$, where θ is the activation threshold. They propose different probabilistic models to capture the influence probability p_{u_j,u_i}, referred to as Bernoulli Distribution (BD), Jaccard Index (JI), Partial Credits - Bernoulli (PC-B), and Partial Credits - Jaccard (PC-J). In the IC model of Saito et al. [208], each active subject independently influences her/his inactive friends with influence probabilities estimated by maximizing a likelihood function with the Expectation Maximization (EM) algorithm. Such approaches [106, 208] have two main drawbacks: (i) they assume that the probability of friends influencing a subject are independent of each other, and (ii) they do not consider the actions not performed by the subject (but performed by her/his friends) to learn the influence probabilities. Next, we detail such drawbacks with an illustrative example.

Figure 2.1a represents the social network of subject u_5. Each node drawn with a dashed line represents a friend of u_5, thus, edges indicate a friendship relation. Each edge is weighted by the influence probability p_{u_j,u_i}. To avoid confusion in the figure, only the incoming edges of node u_5 are represented. Therefore, every edge represents the influence exerted by a certain friend of u_5 on u_5. A red node represents an inactive subject, while a black node denotes an active subject. According to Eq. (2.1), previous approaches predict whether u_5 performs a generic action a as a function of the active friends (u_1, u_2, u_4) and corresponding influence probabilities. The first limitation of such a solution is related to the fact that the probability of friends influencing a subject are considered independent of each other. This assumption may not be always true. For example, the fact that subject u_1 and u_2 are both active can differently affect the decision of subject u_5, especially when nodes u_1 and u_2 are friends and influence each other, as displayed in Figure 2.1b. In this instance, the joint probability of influencing u_5 should be higher if compared to the combination of the independent probabilities (Eq. (2.1)). The second limitation of existing works is related to the fact that they learn the influence probability by considering only the actions performed (*positive instances*) by the subject under analysis (u_5 in our example). However, it may be relevant to take into account the actions not performed by the subject (*negative instances*), but performed by her/his friends, so as to understand who really affects the subject's decisions. For instance, we consider the case where subject u_5 does not perform a certain action, while some of her/his friends do (u_1, u_2, and u_4 in the example in Fig. 2.1a). In this case, by considering also negative samples we can improve and enrich the influence model, as u_5 may be affected by the friends that share the same *negative* decision (u_3 in the example in Fig. 2.1a).

In Chapter 4, we propose to overcome the above-described limitations by introducing a novel approach based on deep learning. The rationale is to learn complex social relationships by means of a DNN architecture, which simultaneously allows us to (i) consider dependencies among users' friends, and (ii) take into account negative instances in the training phase. Chapter 4 describes the different deep learning solutions we proposed and related performance, also comparing them with the results of existing social influence models.

2.4 OSN Manipulation

During the last decade, OSNs have become the conventional communication channel to share opinions and access the news [104]. Therefore, accuracy, truthfulness, and authenticity of the shared content are necessary ingredients to maintain a healthy online discussion. However, in recent times, OSNs have been dealing with a considerable growth of false content and malicious accounts. The resulting wave of misinformation highlights the pitfalls of OSNs and their potential harms to several constituents of our society, ranging from politics to public health. In fact, OSNs have been used for malicious purposes to a large extent [82]. Especially in the context of political discussion, there is a significant risk of mass manipulation of public opinion. Numerous studies concluded that OSNs can be a vehicle for political manipulation, citing factors such as the effect of fake news and misinformation [21, 40, 111, 115, 130, 193, 210, 215, 234], bots [36,38,178,213,231,239,242], trolls [5,19,134,244], and polarization [16,23]. In Section 2.4.1, we provide an overview of recent studies on the manipulation of public opinion in the political context. In Sections 2.4.2 and 2.4.3, we describe the malicious actors that play a pivotal role in manipulation campaigns, i.e., bots and trolls, and we compare our work to empower their detection with respect to other studies in the literature.

2.4.1 Political Manipulation in OSNs

The most remarkable example of political manipulation in OSNs is represented by the 2016 US Presidential election. Since then, there has been a big spotlight on the sovereignty of the US election system and, more in general, on manipulation campaigns in OSNs. This is further confirmed by the ongoing US Congress investigation of Russian interference during the election, where Russia is accused of using trolls and bots to spread propaganda and politically biased information. An in-depth analysis of the 2016 presidential election [36], showed how bots activity can affect democratic political discussion, which in turn can potentially alter public opinion and endanger the integrity of elections. In [20], the authors studied the effects of the manipulation campaign executed by the Russian troll accounts publicly disclosed by US Congress' investigation. Additionally, [75] analyzed Facebook ads allegedly purchased by the Russian government during the 2016 US election.

The online diffusion of false news is another issue in which the activity of bots and trolls is believed to be relevant [35, 149]. In particular, [213] analyzed the pivotal role of social bots in spreading articles from low credibility sources. US is not the only example of reported political manipulation. Similar instances involved other countries worldwide [130,200,219]. In [220], authors showed bots' strategy of targeting influential humans to manipulate online conversation during the Catalan referendum for independence. The presence of bots in Twitter was also analyzed during the 2017 French Presidential Election [83]. Interestingly, an overlapping set of bots that acted in the 2016 US election was also found in the French election, suggesting the existence of a black market for reusable political disinformation bots [242]. More recently (July 1st, 2019), California became the first State to attempt regulating the usage of bots. In fact, the *Bot Disclosure and Accountability Act*[1] requires a self-disclosure of automated accounts that intended to impersonate or replicate human activity in OSNs.

[1] https://leginfo.legislature.ca.gov/faces/billTextClient.xhtml?bill_id=201720180SB1001

In [218], it has been shown that social bots represent only one side of the problem. Human actors also play a key role in the spreading of false information. In particular, state-sponsored trolls activity have been largely studied in relation to the 2016 US election [5, 19, 21, 134]. These studies rely on a list of 2,752 (now-deactivated) Twitter accounts that have been identified as tied to the Russia's "Internet Research Agency" troll farm. The list of such accounts has been disclosed as part of the Congress' investigation of Russian interference in the election. Most recently, Twitter released [96] accounts and related content associated with potential information operations of four countries (i.e., Russia, Iran, Venezuela, and Bangladesh) to enable further investigation to researchers.

Although the attempts from social network providers to suspend suspected and malicious accounts, the presence of bots and trolls in OSNs do not show any sign of decline [70, 134, 242]. In the next subsections, we explore these malicious actors separately and we describe our objectives in relation to existing works.

2.4.2 Social Bots

Social bots are defined as accounts managed completely or in part by computer algorithms [86]. Such software-controlled accounts are designed to impersonate and possibly alter human behavior. Bots can automatically create, share, and spread content in OSNs with the objective of steering discussions and promote specific ideas.

While in the early days of OSNs (i.e., in the late 2000s) creating a bot capable to operate in a human-like manner was not a simple task, nowadays deploying bots has become increasingly simpler and, in some cases, no coding skills are required and several source codes can be found online [143]. Also, bots can run within cloud services (e.g., Amazon Web Services[2], Heroku[3], etc.) and can be provided as a service, i.e. Bot-as-a-Service (BaaS) [84]. More advanced conversational bots, powered by AI, are also provided by companies like ChatBots.io[4] and can be used to deploy spam campaigns, where bots are autonomously capable of engaging with human users. The advances of AI, especially in the area of neural-based natural language generation, has elicited an increasing sophistication of bots' capabilities of generating text content [84]. Also, the availability of large pre-trained language models, such as OpenAI's GPT-2 [198], have

[2] https://aws.amazon.com

[3] https://www.heroku.com

[4] https://home.pandorabots.com/

allowed to program bots that can automatically produce genuine-looking short texts on OSN platforms, making it harder to distinguish between human and automated accounts [7].

The massive scalability that bots can reach through automation represents a major concern in the fight against media manipulation [38,178,231], as further demonstrated by the recent suspension of millions of compromised accounts by Facebook[5] and Twitter[6]. For instance, multiple bots handled by the same human operator (referred to as bot master) can simultaneously generate the same content to amplify and create an illusion of consensus towards a (false) piece of information [86]. In such a way, bots can raise the popularity of a news or of an account and, thus, fool OSNs ranking algorithms [242]. Both the strategies and the algorithms behind the sharing activities of these software-controlled accounts are usually unknown and harder to detect unless their activity is strongly coordinated [55].

Nowadays, bots were estimated to being a consistent portion (about 9–15%) of the Twitter population [231], sharing around two-thirds of links to popular websites [238]. Researchers have brought to the table different tools for detecting malicious automated accounts [55, 57, 145, 222]. Such tools, based mainly on Artificial Intelligence (AI), have proven to achieve prominent results in the identification of bots [242]. In particular, Botometer[7], a publicly-available machine learning-based tool maintained by Indiana University [69, 231, 242], has demonstrated to be effective in bots identification on Twitter, with a classification (bot vs. human accounts) performance of 0.95 Area Under the Curve (AUC). Botometer is based on an ensemble classifier that aims to provide an indicator, namely bot score, used to classify an account either as a bot or as a human. To feed the classifier, the Botometer API extracts about 1,200 features related to the Twitter account under analysis. These features fall into six broad categories and characterize the account's profile, friends, social network, temporal activity patterns, language, and sentiment.

However, as we mentioned above, bots have been becoming increasingly sophisticated. They can interact with human users, engage in conversations, and also create original content in a fully automated way [85]. Most importantly, bots have evolved to further resemble the appearance of humans in order to escape detection. It is, therefore, of paramount importance to adapt existing countermeasures to the mutable and

[5] https://edition.cnn.com/2019/05/23/tech/facebook-transparency-report/index.html

[6] https://www.bbc.com/news/technology-44682354

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

growing sophistication of automated accounts. This, in turn, requires to investigate the strategies adopted by bots to keep interacting with humans and avoiding the suspension from OSN platforms. For this reason, in Chapter 5 (Sections 5.2 and 5.3), we aim to explore bots' activity over years to monitor their evolution and examine their current online behavior. In particular, we investigate bots' activity in Twitter during the last two US voting events, i.e., the 2016 Presidential Election and the 2018 Midterms. To the best of our knowledge, this is the first attempt to track the activity of malicious accounts over different elections for studying their evolution. More specifically, we analyze both the strategies implemented by bots to manipulate the opinion of the human population (while avoiding detection) and how human users have handled such manipulation attempts over the years. The rationale of our study is to provide insights related to bots' strategy and evolution for informing actionable policies in the detection of manipulation campaigns.

2.4.3 Trolls

Troll accounts denote a class of human users who aim to manipulate the public opinion and sow conflict on social or political issues by sharing inflammatory and false information in online platforms [44]. In the political context, and more specifically during the 2016 US Presidential election, trolls aimed to harm the political conversation and create distrust in the political system [19]. These trolls were allegedly funded by the Russian government to influence conversations about political issues, with the goal of creating discord and hate among different groups [100]. In [19], the authors characterized the effect of the manipulation campaigns carried out by Russian trolls during the 2016 US Presidential election. An analysis of the production and consumption of fake news generated by Russian Trolls on Twitter is offered in [20]. Along the same research line, [21] studies how to predict users who spread trolls' content with the goal of understanding how to contain their influence in the future.

In [244], state-sponsored trolls have been investigated on Twitter and Reddit. In particular, the authors focus on Russian and Iranian trolls trying to characterize their behavior and strategies over time. They found that the behavior of such trolls is not consistent over time. For example, they change their language, their (self-declared) location, and tactics according to the targeted manipulation campaign. Also, authors [244] found that trolls' activity does not appear completely disjoint from the activity of regular users. Thus, developing automated systems to identify and block such accounts is not a straightforward task and remains an open challenge [244]. In fact, differently from bots, the detection of troll accounts have not found any effective solution until recently.

Concurrently with our undertaking, two approaches for unveiling the activity of Russian trolls on Twitter have been proposed [5, 134]. To identify trolls' attempt to manipulate the public opinion during the 2016 US election, [5] identified 49 linguistic markers of deception and measured their use by troll accounts. They show that such deceptive language cues can help to accurately identify trolls. In [134], the authors proposed a detection approach that relies on users' metadata, activity (e.g., number of shared links, retweets, mentions, etc.), and linguistic features to identify active trolls on Twitter. Our work is similar to the above-mentioned efforts in the objective of unveiling troll accounts. However, differently from [5, 134], we do not aim at spotting differences in trolls' language or metadata. We do not exploit such signals to identify trolls, but we focus on uncovering hidden behavioral differences between troll and non-troll accounts. As we do not leverage either social network-dependent metadata or language features, our approach can be generalized on different OSNs and to state-sponsored trolls originating from different countries. In fact, our solution only relies on the sequence of users' activity on online platforms to capture the incentives the two classes of accounts respond to.

2.5 Awareness of OSN Perils

Raising users' awareness of OSN perils, along with providing them with some countermeasures, is the final objective of this thesis. In Section 2.5.1, we describe existing works in the literature that share with us the objective of increasing users' awareness of OSN risks and we present the challenges to be tackled to achieve this objective. In Section 2.5.2, we introduce our proposed approach to face these challenges and we compare it to existing solutions.

2.5.1 Awareness Services

The problem of raising users' awareness of OSN perils has recently gained attention with the explosion of political scandals, e.g., the Cambridge Analytica data breach [47] and the Russian interference in the 2016 US Presidential election [20]. The research community attempted to encourage a more cautious usage of OSNs, especially in the context of privacy. Several works tried to understand users' concern about their privacy and the initiative they act to protect their personal information [3, 105, 230]. Similarly with our objective, more recent works developed the idea of implementing services to support privacy awareness [118, 196]. In [196], the author provides guidelines and details requirements to develop a privacy awareness service. Gulenko [118] developed an application that inspects users' Facebook profile and provide them an assessment about the security and privacy of their account. The main difference of our work with respect to this effort is that in [118] users' privacy is evaluated by only checking the privacy-setting that they selected on the OSN platform (e.g., if they have a public or private profile). Our work, instead, has the objective of performing an evaluation of users' privacy not only based on privacy-settings but also, and mainly, on the publicly-available data shared by the online population. In particular, in this thesis (in both Chapters 3 and 4), we aim to investigate whether users' personal and sensitive information (e.g., their current location) can be inferred from data published by other users (e.g., their friends and acquaintances) in online platforms.

However, as we described in Section 1.3.4, developing an awareness service represents a challenging objective. This is especially true if the awareness service should be implemented on mobile devices, e.g., smartphones. The main problem is related to the collection of users' data, which in turn can introduce additional privacy and security risks. The collection of human data is not a simple task, which has struggled researchers for years. Different approaches have been proposed by the research community. Among these approaches, Mobile Crowd Sensing (MCS) represents a promising solution. MCS is an emerging paradigm that is based on the sensing capabilities of mobile devices [121]. Potential MCS applications span a wide spectrum in terms of application domain [228], ranging from traffic estimation [133, 177, 189], and place categorization [60] to smart cities [30, 179, 233] and social trend detection [120, 241]. Though these applications were established to pursue specific purposes, efforts have also been made towards formally characterizing the operation of MCS systems in an application-agnostic way. However, issues related to device heterogeneity, security, and privacy have limited the rise of MCS platforms [11, 12].

To address these challenges, in this thesis, we introduce VIVO, an open framework for crowd-sensed data gathering, where security and privacy are managed within the framework at the client side. In the next subsection, we present the novelty of VIVO with respect to existing solutions.

Table 2.1: Testbeds comparison

	LiveLabs	NetSense	PhoneLab	IoTLab	**VIVO**
standard smartphone OS	✓	✓	✗	✓	✓
simultaneous experiments	✗	✗	✓	✓	✓
open range of applications	✗	✗	✓	✗	✓
real-time data collection	✓	✗	✗	✓	✓
fixed and mobile sensors	✗	✗	✗	✓	✓
embedded security	✓	✓	✗	✗	✓
privacy-preserving	✓	✓	✗	✓	✓
multi-platforms	✓	✗	✗	✗	✗
incentives	✓	✓	✓	✗	✗

2.5.2 MCS Testbeds

In this Section, we show how VIVO differs and overcomes some of the limitations of existing MCS frameworks. Table 2.1 compares VIVO with existing solutions in the literature. VIVO allows an easy development and deployment of experimental software on mobile devices. More precisely, similarly to PhoneLab [182] and SmartLab [148], experiment developers can dynamically deploy their own application on each mobile device involved in the data collection. However, while PhoneLab [182] requires a modified version of the Android OS to run on the employed smartphones, thus limiting the set of potential participants (from now on referred to as *volunteers*), VIVO applications run on **standard smartphone OS** (i.e., Android), without any extra-hardware requirements and pre-deployment testing. SmartLab [148] is an architecture for managing a cluster of real and virtual devices. Users can install executables on devices, capture their screen, and issue UNIX shell commands. While Smartlab is targeted towards scenarios requiring low-level control over smartphones, e.g., deployment and debugging, VIVO is a framework focused on the gathering of crowd-sensed data.

Recent similar efforts are LiveLabs [25], NetSense [221], and IoT Lab [81]. Livelabs [25] is a mobile testbed that continuously collects sensor data from participant personal devices in four public spaces in Singapore. The goal of this data collection is to analytically extract context information to trigger consumer trials provided by retailers or service providers. NetSense [148] aims to understand the impact of the digital world (mobile communications and OSNs) on social relationships by collecting sensor data from instrumented smartphones distributed to hundreds of students at the University of Notre Dame. IoT Lab [81] has been developed with the purpose of researching the potential of crowd-sensing as an extension to the traditional IoT infrastructure. Through a smartphone application, the crowd was allowed to participate in experiments by contributing with sensory data and knowledge.

Unlike these previous efforts, where a single static application is installed on each smartphone to constantly save data collected from sensors, VIVO allows the deployment of multiple **simultaneous experiments** introducing an enrolled crowd-sensing model. In such a model, developers are not limited to a fixed set of experiments but they can build an **open range of application** without any constraint, in a more agnostic and generic way.

Differently from other approaches, the data collected through VIVO can be accessed both at the end of the experiment, as in traditional testbeds, as well as in **real-time**, as needed by several Big Data applications. This enables a broad range of applications that require low latency communication, e.g., navigation, recommendation, monitoring, and control. Moreover, VIVO is a human- and sensor-based testbed. It integrates two components: a crowd-sensing scheme composed of mobile devices (volunteer smartphones), and Syndesi [79], an IoT framework for smart buildings, which includes multiple fixed sensors. Syndesi [79] is a framework interconnecting heterogeneous devices from wireless sensor networks as well as mobile devices, providing central resource management and user-personalized smart automation. It is implemented in the premises of the University of Geneva, although it has been designed to support portability. One of its functionalities is the gathering of environmental data, such as temperature, illuminance, and humidity from the devices possessing sensors that are registered in its resource registry. The data are written into a database (Syndesi DB) hosted by the Syndesi server. The integration between VIVO and Syndesi empowers the seamless combination of resources coming from different sources, which allows to study the interaction between human beings and the surroundings.

Finally, the fundamental provisioning of **security** and **privacy** along with its capability of gathering heterogeneous data in real-time makes VIVO suitable for the purpose of this thesis, by allowing applications for monitoring and controlling users privacy and manipulation risks. We present our application to raise users' awareness of such risks, the VIVO framework and its architecture in detail in Chapter 6. Here, we just point out that, differently from other MCS frameworks, VIVO manages security and privacy within the framework, at the client side. In particular, we provide an API with all the methods necessary to secure and privatize the collected data before they leave the smartphone. VIVO also supports *differential privacy* [76], which is a method to share information about a set of data by describing aggregate statistics of the dataset while preserving and limiting the disclosure of private information about individuals. Differential privacy is also defined as a property that provides an upper bound on the information a third party can obtain from the data after the release [77]. This means that, if a *differentially* private method is used to protect data, then any third party that sees a statistical result over the data would not be able to identify individuals' information up to a certain specified privacy parameter (referred to as privacy loss).

The current limitations of VIVO concern mainly two aspects. The first limitation is related to the fact that, as of today, VIVO applications can only run on Android smartphones. A **multi-platform** capability (e.g., both iOS and Android) is desirable to extend the set of participants and experiments. For instance, Livelabs [25] supports iOS, Android, and Windows Phone (WP) smartphones. The second limitation concerns the recruitment of volunteers, which is a typical issue in crowd-sensing platforms. Crowd incentives, as well as ensured Quality of Information (QoI) of crowd-sensed data, are considerably important aspects for the success of MCS applications. Existing solutions (e.g., NetSense [221], PhoneLab [182], Livelabs [25]) propose to offer smartphones and discounted mobile planes to encourage volunteers to participate in their experiments. Currently, VIVO does not provide similar **incentives** in exchange of volunteers' participation. To reward volunteers we launched the context "Volunteer of the Year", where each participant is encouraged to participate in the largest number of experiments to win a prize. However, the recruitment of a large set of volunteers requires more effective incentives. Therefore, we plan to develop a reward-based mechanism that allows the experiment developer to advertise prizes to the volunteers according to their involvement in the experiment.

2.6 Machine Learning

In this Section, we provide theoretical basics of machine learning algorithms employed throughout this thesis. Section 2.6.1 details deep learning architectures based on deep neural networks, which we use to assess users' privacy in Chapter 3 and to model social influence in Chapter 4. In Section 2.6.2, we describe the fundamentals of reinforcement learning algorithms that we utilize for detecting malicious troll accounts in Chapter 5.

2.6.1 Deep Neural Networks

In this Section, we provide the background needed to comprehend the *deep learning* paradigm, which is fundamental in several approaches presented throughout this thesis. Deep Learning is a fancy marketing name for artificial neural networks, a class of machine learning algorithms inspired by biological nervous systems. In recent years, neural networks [150, 211] have found successful applications in a growing number of areas ranging from speech and image recognition to natural language processing and computer vision. This is confirmed by the intensive research and development that has been carried out during the last years in numerous fields, not only directly related to Artificial Intelligence (AI) applications.

An artificial neural network is defined by a combination of three layers: input layer ($\mathbf{x}$), hidden layers ($\mathbf{h}$), and output layer ($\mathbf{y}$). A Deep Neural Network (DNN) is an artificial neural network with multiple hidden layers ($\mathbf{h} = \mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_L$) between the input and output layers. Each layer is composed of multiple processing units (*neurons*), which use the output from the previous layer as input. The cascade of multiple layers is connected as

$$\mathbf{h}_j = \begin{cases} \phi_j(\mathbf{x}\mathbf{W}_{xh_j}) & \text{if } j = 1 \\ \phi_j(\mathbf{h}_{j-1}\mathbf{W}_{h_{j-1}h_j}) & \text{if } 1 < j \leq L \end{cases}$$

$$\mathbf{y} = \phi_o(\mathbf{h}_L \mathbf{W}_{h_L y}) \ ,$$

where $\mathbf{W}_{kl}$ indicates the weights of the connections between layer k and l, while ϕ_j is a non-linear activation function (e.g., sigmoid, ReLU, tanh, softmax) of each hidden node at layer j, and ϕ_o is a non-linear activation function of each output node.

The cascade of multiple layers consisting of non-linear neurons allows a DNN to approximate any continuous function. Moreover, DNN replaces the manual feature extraction procedure by building up a complex hierarchy of concepts (abstractive features) through the multiple layers to automatically extract relationships embedded in the input data [125]. The predictive model of a DNN can be formulated as $\hat{\mathbf{y}} = f(\mathbf{x}|\Theta)$, where $\hat{\mathbf{y}}$ denotes the predicted output, Θ represents the model parameters (i.e., the inter-layers weights), and f indicates the function that maps the input $\mathbf{x}$ to the output $\hat{\mathbf{y}}$ based on the DNN architecture, i.e., $f(\mathbf{x}) = \phi_o(\phi_L(\ldots\phi_2(\phi_1(\mathbf{x}))\ldots))$.

We employ the DNNs' capability of learning relationships embedded in the input data to model social relationships and dependencies among OSN users. In particular, we developed two approaches for (i) assessing users' location privacy based on the locations shared by other OSN users (presented in Chapter 3) and (ii) modeling social influence among users to forecast their future activities (presented in Chapter 4).

2.6.1.1 Recurrent Neural Networks

In Section 2.6.1, we described a general architecture of a DNN. As in such architecture there are no loops (or cycles) between the nodes of the network and the flow of information moves *forward* from the input to the output neurons (through the hidden layers), this class of DNNs is referred to as *feed-forward neural networks*. However, feedback connections have been extensively used in Recurrent Neural Networks (RNNs) for modeling the temporal dynamics of sequences in a wide range of fields, such as machine translation, speech, handwriting, and image recognition. The feedback loop allows RNNs to have memory of previous instances and makes them suitable to learn sequential data, such as human activities. The most commonly used RNNs are referred to as Long Short-Term Memory (LSTM) [126] and Gated Recurrent Unit (GRU) [64]. In this thesis, we rely on RNN architectures in Chapter 3 to learn recurrences within a sequence of locations, and in Chapter 4 to model recurrent patterns in social activities.

2.6.1.2 DNN Challenges

Although DNNs have reached outstanding performance on different tasks, they produce an obscure model and, for this reason, they are referred to as black boxes. A black box is defined as a predictor, whose internals are either unknown to the observer or they are known but are not understandable by humans [116]. As an example, we can consider a machine learning model fed by a certain number of features, whose relevance for the outcome prediction is not known or cannot be computed. Contrarily, an *interpretable* (i.e., understandable) solution is desirable as deep learning is nowadays used in critical areas, such as justice and medicine, where the understanding of the model logic, functionality, and results can be necessary. However, according to the analysis provided by Lipton [158], the word *interpretability* holds no agreed-upon meaning, despite numerous papers assert the interpretability of their models. In this study [158], the author claims that interpretability is not a monolithic concept, but it reflects several distinct ideas. For this reason, to be meaningful, any assertion regarding interpretability should fix a specific definition. Lipton describes two categories of interpretability. The first is related to the concept of transparency (as the opposite of blackbox-ness) and focuses on the understanding of the model, its components, and training algorithms. The second category, namely *post-hoc explanations*, does not aim to explain how a model works but has the objective of extracting information from a learned model. In particular, given the output of a black box predictor, the problem consists in reconstructing an explanation for it, without necessarily elucidate the logic behind the model. This problem is called *outcome explanation problem*, and it is the notion of interpretability we study in Chapter 4 to model social influence among OSN users. Overall, our attempt is in line with the emerging paradigm of eXplainable Artificial Intelligence (XAI) [119, 191], which aims to create a suite of techniques for producing more explainable models, while maintaining a high level of performance.

2.6.2 Reinforcement Machine Learning Algorithms

In this subsection, we present the fundamentals of reinforcement machine learning algorithms, which we utilize in Section 5.4 for the detection of troll accounts in OSNs. Reinforcement Learning (RL) is defined as the problem faced by an agent that learns how to behave through trial-and-error interactions with an uncertain environment [136]. The objective of the agent, in this game-like situation, is to achieve a goal by performing a sequence of actions in the environment. The agent, through a trial-and-error strategy, learns the actions to take according to some rewards or penalties (associated with the actions it performs). RL represents, along with supervised learning and unsupervised learning, one of the major field in machine learning. As this approach relies on the experience that the agent acquires in the trials and does not exploit any training examples, this machine learning field is not considered a supervised learning solution. RL algorithms proved their effectiveness in various fields, such as telecommunications, automation and control, and gaming as chess, Atari, and Alpha Go. Nowadays, this approach is also capable of beating the world champion e-sports team in strategy games as Dota 2 (a multiplayer online battle arena game).

In this thesis, we utilize a variant of RL that aims at solving the inverse problem of Reinforcement Learning, namely Inverse Reinforcement Learning (IRL). Therefore, in this subsection, we first formalize the RL paradigm, which will guide the subsequent presentation of IRL.

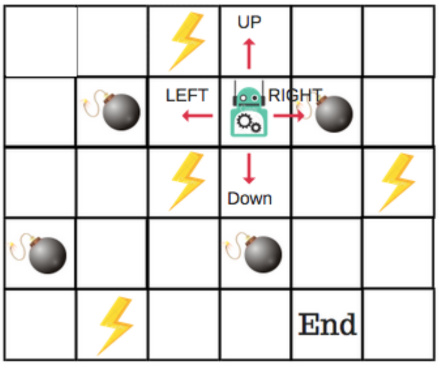


Figure 2.2: Reinforcement Learning (RL) example [94]

2.6.2.1 Reinforcement Learning

The easiest example to understand RL is related to a maze scenario, where a robot has to reach the endpoint in the shortest possible time. Figure 2.2 displays the environment of such example. The robot has the possibility to move one tile at a time. It gets numerical rewards or penalties (e.g., +/- 100 points) if it encounters lightnings or mines, respectively. Also, the robot is penalized for every move, thus, it has to find the fastest way to the endpoint in order to maximize the cumulative reward.

The most used and straightforward way to formulate a RL problem is to frame the environment as a finite Markov Decision Process (MDP), which includes:

- A finite set of states S, which represents the situation in which the agent finds itself in the environment (e.g., a tile of the maze);
- A finite set of actions A that the agent can perform in the environment (e.g., turn left in the maze);
- Transition probabilities T between states (e.g., the probability of transition from state $s \in S$ to state $s' \in S$ after performing action $a \in A$);
- A finite set of scalar rewards R associated with each transition (e.g., the robot gets +100 points if it runs into a lightning);
- A discount factor γ, which determines the importance between short-term and long-term rewards.

The general framework of a RL problem is displayed in Figure 2.3. The agent (e.g., a robot) has to learn how to achieve a goal (e.g., the endpoint of a maze) from the

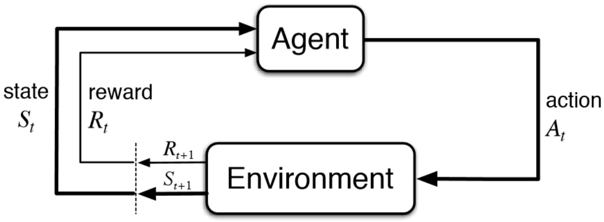


Figure 2.3: RL schema [223]

interaction with the environment (e.g., a maze with rewards and penalties). The agent and the environment interact at each step t of a sequence of discrete time steps ($t = 0, 1, 2, \ldots$) [223]. The interaction flow follows the schema displayed in Fig. 2.3: At each step t, the agent selects an action $A_t \in A$ based on its current state $S_t \in S$. The environment responds to such action and presents to the agent a new state $S_{t+1} \in S$ and a reward $R_{t+1} \in R$ associated with the transition (S_t, A_t). The sequence of state-action pairs that arises from the interaction of the agent with the environment is referred to as *trajectory* $\zeta = \{(S_0, A_0), (S_1, A_1), (S_2, A_2), \ldots\}$.

The MDP schema can be applied to various RL problems in many different ways. It is out of the scope of this thesis to distinguish the different algorithms (e.g., Q-learning, policy learning, etc.) and notions (e.g., the trade-off between exploration and exploitation) discussed in the literature to solve RL problems. However, we conclude this discussion by formalizing the common objective of RL problems. In a RL approach, the agent seeks to select its actions in order to maximize the cumulative rewards it expects to receive in the future. Therefore, the objective can be formulated as a maximization of the expected (discounted) return G_t, which is defined as follows:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}, \tag{2.2}$$

where $\gamma \in [0, 1]$ is the discount rate introduced above.

2.6.2.2 Inverse Reinforcement Learning

IRL is a machine learning framework that aims at solving the inverse problem of RL. While in RL the agent is provided with a reward function, which is used to achieve a given goal, the rationale of IRL is to find the rewards that explain the observed behavior of the agent. This objective has two main motivations [185]. The first reason is for imitation purpose (i.e., to replicate the agent's behavior), when a reward function is not explicitly determined [199]. The second reason is to support behavioral studies, where computational models are needed for understanding human and animal behavior [67, 227, 236].

Formalizing, the objective of IRL is to estimate a reward function that could have led to agent actions. More specifically, and relying on the MDP frame described above, the observed behavior of the agent is expressed as the history of its trajectory ζ. The goal is to uncover the hidden reward function that maps each state-action pair (s, a) to a real value $r_{s,a} \in R$. Therefore, IRL aims to find a function g such that: $g(S, A) = R$. In many applications, however, the size of the state space does not allow to compute a reward function for every state-action pair. Hence, current IRL techniques leverage features that approximate the state-action space to capture the structure of the reward function [1, 153, 185, 201].

Maximum Entropy IRL [251] represents the most popular solution to solve IRL problems. This approach maps a set of features f to the reward R as a weighted linear combination of the feature values:

$$g(f) = g(f, \theta) = \theta^{\mathrm{T}} f = R. \tag{2.3}$$

Most recent approaches propose non-linear models to represent the reward structure [59, 154, 240]. In particular, [240] introduces the usage of Convolutional Neural Networks in the context of IRL in an approach called *Maximum Entropy Deep IRL*. This approach showed comparable performance with respect to Maximum Entropy IRL and is considered particularly suited for life-long learning scenarios [240].

In Section 5.4, we employ such techniques (both Maximum Entropy IRL and its non-linear variation Maximum Entropy Deep IRL) to infer the rewards that could have led troll and non-troll accounts to perform their online activity. We then consider to exploit the estimated rewards as features of a supervised learning algorithm aimed at classifying such accounts.

2.7 Network Science

The scientific field of *network science* focuses on the study of complex networks (e.g., social networks, telecommunication networks, biological networks, etc.) [8, 49, 184]. The term *complex* captures the fact that it is difficult to derive the collective behavior and functionality of such systems by merely analyzing their components. However, behind each complex system, there is an intrinsic network that encodes the interactions between the system's components. As a consequence, an in-depth understanding of the network structure behind complex systems is needed for their collective comprehension [28].

Consider for example the society that requires cooperation between billions of individuals, or the activity of billions of neurons in our brain. Complex networks have also a fundamental role in the most groundbreaking technologies, ranging from Google's search mechanism to the functionality of OSNs and telecommunication technologies. The fact that networks pervade nature, technology, and business elicited an increasingly growing interest in network science during the last decade [28]. A key discovery of network science is that the properties of networks emerging in different domains are similar to each other. This allowed network scientists to use a common set of mathematical tools to explore such systems. In the next subsections, we present network science fundamentals and the mathematical tools used throughout this thesis.

2.7.1 Network Science Fundamentals

In this subsection, we detail several properties and measures of complex networks that we utilize to analyze OSNs. Networks are usually represented by means of a graph composed of a set of nodes and the connections among them, referred to as edges. The edges of a graph can be either directed or undirected. In the former case, a source node points to a destination node indicating a unidirectional form of interaction (e.g., links in the WWW or phone calls). In the latter case, connections are not directed and indicate bidirectional relationships (e.g., friendship). According to this difference, we can distinguish directed or undirected graphs. Additionally, weighted networks are a particular kind of graph, where each link (i, j) has a unique weight w_{ij} representing a property of the connection. For example, in a social network weights can represent the total number of interactions (e.g., phone calls) between pairs of individuals.

Given a generic graph $G = (V, E)$, where $V = (v_1, v_2, \dots, v_N)$ is the set of nodes with cardinality N, and $E = ((v_i, v_j), (v_i, v_k), \dots, (v_k, v_l))$ is the set of edges with cardinality L, we can define:

- **Degree** k_i is the number of connections a node i has with other nodes (e.g., number of friends). In directed graphs, we define the in-degree k_i^{in} (resp. out-degree k_i^{out}) as the number of in-coming (resp. out-coming) edges of node i.

- **Degree distribution** represents how nodes' degree are distributed. The distribution is obtained by considering the degree probability p_d, which represents the probability that a node in the network has degree equal to d and, thus, it is computed as:
$$p_d = \frac{n_d}{N}, \tag{2.4}$$
where n_d is the number of nodes with degree d.

- **Average degree** is the mean value of the nodes' degree over all the network. In an undirected graph is computed as:
$$\langle k \rangle = \frac{1}{N} \sum_{i=1}^{N} k_i = \frac{2L}{N}, \tag{2.5}$$
where the factor 2 corrects for the fact that in the sum each edge is counted twice. The average degree of a directed networks is calculated as:
$$\left\langle k^{in} \right\rangle = \frac{1}{N} \sum_{i=1}^{N} k_i^{in} = \left\langle k^{out} \right\rangle = \frac{1}{N} \sum_{i=1}^{N} k_i^{out} = \frac{L}{N} \tag{2.6}$$

- **Adjacency Matrix A** provides a mathematical and complete description of a graph. It is a $N \times N$ matrix, whose generic element A_{ij} is equal to 1 if nodes i and j are connected, and is 0 otherwise.

- **Density** D is the ratio between the number of edges L connecting the nodes and the possible number of edges in a network with N nodes, which is $N(N-1)/2$.

- **Diameter** d_{max} is the longest of the shortest paths in the network, where the shortest path between nodes i and j is the path with the fewest number of edges.

- **Clustering coefficient** C_i is the ratio between the number of edges connecting i's neighbors to each other and the total number of possible connections among them. It captures the extent to which the neighbors of a given node are connected among each other. It is computed as

$$C_i = \frac{2L_i}{k_i\,(k_i - 1)}, \tag{2.7}$$

where L_i represents the number of links between the k_i neighbors of node i.

- **Average clustering coefficient** $\langle C \rangle$ is the degree of clustering of the entire network, which is computed by averaging C_i over all the nodes in the network.

2.7.2 Network Structures

One of the most interesting findings about complex networks is that these structures are not random graphs. These networks show big *inhomogeneities* and a high level of order and organization. The degree distribution is broad, with a tail that often follows a power law. Thus, many nodes with low degree coexist with some nodes with large degree. The distribution of edges is not only *inhomogeneous* globally, but also locally, with high concentrations of edges within groups of nodes, and low concentrations between these groups. This feature of real networks is called **community** structure [93, 102] or mesoscale level structure (as the scale lies between the scale of the nodes and that of the entire graph).

Usually, a community is defined as a group of nodes that have a higher likelihood of connecting to each other than to nodes from other communities. In social networks, for example, communities can represent circles of friends, or people living in the same geographical area, or a group of individuals who pursue the same interests. In particular, the tendency of individuals to associate and bond with similar others is a social phenomenon known as **homophily** [172]. The rationale is that nodes with similar attributes (e.g., age, opinions, interests, etc.) are more likely to attach to each other than dissimilar ones. The concept of homophily is often used to weight the edges of a weighted network to indicate and quantify the similarity between pair of users [170].

Different applications exploit the concept of homophily, for instance to cluster individuals in communities [170] or to infer personal information (e.g., political leaning) [167]. For example, **label propagation** is a widely used algorithm that leverages homophily among individuals to assign them a label (e.g., left- or right-wing leaning). Label propagation is a network-based algorithm, where each node is assigned a label, which is updated iteratively based on the labels of node's network neighbors. A node takes the most frequent label of its neighbors as its own new label. The algorithm proceeds by updating labels iteratively and stops when the labels no longer change.

Another network structure of pivotal importance in the study of social networks is represented by the **ego network**. The ego network is a star network, where the ego is the center of the star and the other nodes, referred to as *alter*, are ego's direct social connections (e.g., friends, relatives, etc). This means that every edge in the ego network represents a social tie between the ego and the alter.

2.7.3 Centrality Measures

Centrality measures aim to quantify the extent to which a node is central in a graph G. These measures can provide a ranking that identifies the most important nodes within the graph. The word importance can have different facets, which might vary with the nature of the network and its application. For example, nodes can be important for their prominence or structural importance within the graph, while others can be important for the influence they have over the other nodes in the network. For this reason, different centrality measures have been introduced. We here present the following centrality measures:

- **Degree centrality**: every node is scored based only on its number of edges. The degree centrality for a node i is the fraction of nodes it is connected to. Therefore, it is computed as:

  $$c_d(i) = \frac{k_i}{N-1} \tag{2.8}$$

  The same formulation holds for the in-degree (resp. out-degree) centrality measure $c_{d_{in}}(i)$ (resp. $c_{d_{out}}(i)$), which in turn replaces k_i with k_i^{in} (resp. k_i^{out}).

- **Betweenness centrality** measures the number of times a node lies on the shortest path between other nodes in G [41]. Formally, the betweenness centrality of a node i is the sum of the fraction of all-pairs shortest paths that pass through i:

  $$c_b(i) = \sum_{s \neq i \neq t \in G} \frac{\sigma(s,t|i)}{\sigma(s,t)}, \tag{2.9}$$

where $\sigma(s,t)$ is the total number of shortest paths from node s to node t, and $\sigma(s,t|i)$ is the number of those paths that passes through node i.

- **Closeness centrality** scores every node based on their "closeness" to all the other nodes within G. The closeness centrality of node i is the reciprocal of the average shortest path distance to i over all the $n-1$ reachable nodes [95]:

$$c_C(i) = \frac{n-1}{\sum_{v=1}^{n-1} d(v,i)}, \tag{2.10}$$

where $d(v,i)$ is the shortest-path distance between nodes v and i.

- **Eigenvector Centrality** is a measure of influence of a node in a graph by computing the centrality for a node based on the centrality of its neighbors [39]. The eigenvector centrality tries to generalize the degree centrality by taking into account the importance of the neighbors. Specifically, the eigenvector centrality for node i is the i-th element of the vector x defined by:

$$\mathbf{A}x = \lambda\mathbf{A}, \tag{2.11}$$

where $\mathbf{A}$ is the adjacency matrix of G with eigenvalue λ. In general, there exist different eigenvalues λ for which a non-zero eigenvector solution exists. However, given that every element of the adjacency matrix is non-negative, according to the Perron–Frobenius theorem there is a unique largest eigenvalue, which is real and positive. This greatest eigenvalue results in the eigenvector centrality measure.

- **PageRank**: it is a variant of EigenCentrality, which also considers the direction and the weight of each edge. PageRank is the algorithm developed by Google to rank web pages in their search engine [188].

Other techniques are also employed in the literature to explore the centrality (also referred to as "embeddedness") of nodes in a network. For instance, the **k-core decomposition** aims to determine the set of nodes deeply embedded in a graph by recursively pruning nodes with degrees less than k. In other words, the k-core is a subgraph of the original graph in which every node has a degree equal to or greater than a given value k. The k-core decomposition assigns a core number to each node, which is the largest value k of a k-core containing that node. The higher the core number k is the more embedded the node is in the network.

2.8 OSNs

In this Section, we provide an overview of the different kinds of OSNs considered in this thesis and their relation with our analyses and approaches.

Microblogging Platforms Microblogging is a combination of blogging (e.g., to add content to a blog) and instant messaging that allows users to create short messages to be shared in OSNs. These short messages can include different kinds of content, including text, images, video, audio, and hyperlinks. Twitter represents the most popular microblogging platform. In Twitter, users can share content and interact with each other through messages referred to as "tweets", which can have a maximum length of 280 characters. Specifically, Twitter users can create original tweets, re-share others' tweets (i.e., retweet), respond to others' tweets (i.e., reply), involve other users in their tweets (i.e., mentions), and like others' tweets. In this thesis, we use Twitter data to assess user' location privacy in Chapter 3 and to capture the online discussion during US political elections in every analysis related to the manipulation peril in Chapter 5.

Location-Based Social Networks Location-Based Social Networks (LBSNs) are OSNs that are built upon the notion of bringing together the places we visit with the friends we connect to [187]. In fact, in recent years, LBSNs have become popular services that allow users to register (*check-in*) at named places and share their location with their friends. Check-in information includes latitude, longitude, category of the place (e.g., restaurant), the ID of the location, and time of the check-in. Historical check-ins provide useful hints about user interests and preferences, and represent a promising source of activity data to study human behavior and social dynamics [187]. In this thesis, we use data from Foursquare[8], which is the most popular LBSN, to test our proposed social influence approaches in Chapter 4.

Event-Based Social Networks An Event-Based Social Network (EBSN) is a web platform where users can create events, promote them, and invite friends to participate. Events range from small get-together activities, e.g., Sunday brunch or movie night, to bigger events, e.g., concerts or conferences [160]. The rationale behind the choice of

[8] www.foursquare.com

utilizing an EBSN is the intrinsic agglomerative power of the events. In fact, participation in an event represents a direct and explicit form of social interaction, other than a personal interest. In this thesis, we use data from Plancast[9], an EBSN that provides a social network service to connect friends and users with common interests. In the event main page, a Plancast user can see the information related to the event, e.g., date, location, and description, along with the confirmed participants. This information may activate processes of social influence, which can drive user participation in the events [99, 170], as we describe in Chapter 4.

[9] www.plancast.com

3

Geo-Location Privacy Leakage in OSNs

3.1 Introduction

It appears evident from Chapters 1 and 2 that the protection of users' sensitive data in OSNs is nowadays far from being effective. In this respect, geographical location is considered both a very sensitive and highly-valuable information. On one hand, users tend to protect their privacy by rarely publishing their location in OSNs. This operation is referred to as *geo-tagging*. On the other hand, users' position is a business-relevant information that third parties, e.g., LBS providers, may be interested to obtain. In previous studies, it has been shown that users' unexposed location can still be accurately inferred by combining different sources of information, such as users' generated contents (e.g., public messages), mobility patterns, and social cues [78, 207, 212].

Along this research line, we explore the problem of geo-location privacy in OSNs. We consider a scenario where a third-party entity (e.g., a LBS provider that uses OSN platforms to perform targeted advertisement) is interested to obtain users' locations. This third party is the attacker that tries to violate users' privacy by inferring the locations that users are not willing to expose. In such a scenario, our first objective is to measure the level of geo-location privacy of a user. We define the *geo-location privacy* of a user as the geographical distance between the actual location and the one estimated by the attacker. Based on this definition, we investigate if a user's privacy can be violated by using information shared by other OSN users. More specifically, we examine whether a user's undisclosed location can be uncovered by leveraging other users' locations. Notice that such an attack does not require the implementation of any illicit strategies, as it is based only on the information attainable from publicly-available geo-tags. As users are generally not aware of the potential privacy risks behind this public data exposure, it is crucial to provide them with tools to measure the geo-location privacy and control it (i.e., to be capable of setting the level of privacy a user is comfortable with). Summarizing, in this Chapter, we aim to respond to the following RQs:

RQ 1.1: *Can an attacker infer users' personal information from publicly-available data of other OSN users?*

RQ 1.2: *Can we provide users with countermeasures to control the public exposure of their data?*

RQ 1.3: *Can we allow users to measure the impact of their characteristics, behavior, and online activity on their privacy?*

Towards the objective of addressing the above RQs, we provide the following three main contributions [163] by considering the Twitter OSN as a study case:

- To assess users' geo-location privacy (RQ 1.1), we propose a novel deep learning architecture that, starting from publicly-available geo-tags, attempts to violate users' privacy by unveiling their unexposed locations.
- We propose two data perturbation techniques that users can employ to control the public exposure of their geo-tags (RQ 1.2) and, in this way, improve privacy. These strategies are called geo-tag obfuscation and geo-tag reduction and, by using them, a user can provide noisy locations or diminish the number of shared geo-tags, respectively.
- We model the privacy value obtained with the deep learning approach as a function of users' characteristics related to mobility, social network, and enforced

data perturbation (RQ 1.3). This model, which is based on a machine learning algorithm, allows us to measure the impact of each feature on privacy, thus enabling their proper control.

This Chapter is structured as follows. Section 3.2 presents the methodology to measure users' geo-location privacy. In Section 3.3, we describe the proposed strategies to control the level of users' privacy. Finally, experimental settings and results are presented in Section 3.4 and 3.5, respectively.

3.2 Geo-Location Privacy Measurement

In this Section, we initially present the formulation of the privacy measurement problem. Then, we describe the deep-learning methodology that we employ to solve it.

3.2.1 Geo-Location Privacy Definition and Measurement

As mentioned in Section 3.1, social cues represent relevant information to violate users' privacy. We model social relationships within the OSN platform as an undirected graph $G = (V, E)$, where V is the set of users and E is set of edges encoding friendship relations. Notice that, on Twitter, social connections are based on the *followee/follower* paradigm. Hence, we consider two users to be friends if both follow each other and we denote the 1-hop neighborhood of u as the set of u's friends, the 2-hop neighborhood as the set of u's friends of friends, etc.

We now describe how we leverage other users' (e.g., neighbors') data to violate a target user's privacy. Privacy is generally considered a problem-dependent and subjective metric. This holds true for location privacy as well. In fact, each user may have a different perception of the intrusiveness of a precise localization. However, to perform our analysis, we require an objective measure of privacy. We define the privacy P_u of the generic user u as the average geographical distance between the locations visited by u and the ones estimated by a certain attacker. In the following formula

$$P_u = \frac{1}{N_u} \sum_{t_i} Dist(l_{t_i}^{(u)}, \hat{l}_{t_i}^{(u)}) \tag{3.1}$$

$l_{t_i}^{(u)}$ and $\hat{l}_{t_i}^{(u)}$ are the actual and estimated geo-tags of the tweet published by user u at time t_i. $Dist(x, y)$ is the geographical distance between locations x and y. Notice that such distance is always greater than (or equal to) zero as it is computed as the Haversine distance [62] between pairs of latitude and longitude coordinates. Notice that the Haversine distance measures the angular distance between two points on the surface of a sphere and, therefore, considers the curvature of the Earth. N_u is the total number of tweets published by u and provided with a geo-tag. It should be noticed that, to assess users' privacy, we use as ground truth the locations actually shared by the users on Twitter. However, when we simulate the location inference task (Section 3.2.3), we hide this information as it represents the target to be inferred by the attacker.

Based on this formulation, we examine whether a user's undisclosed location can be violated by using other OSN users' information. More specifically, we leverage the location information shared by other OSN users. The rationale is to understand if other users' locations can provide useful information to infer the location of a given user. As an example, the question we aim to answer is: Can we infer the location of Alice given that Bob is in his office in Manhattan, Carol is in a gym in Brooklyn, and Ted is in a bar in Times Square? Notice that Bob, Carol, and Ted are not necessarily Alice's friends.

Formally, the objective of the attacker is to estimate the geo-tag of the content (in the considered case, a tweet) published by the generic user u from the set of geo-tags that have been shared on the OSN platform by other users within a given period of time. Such period of time is discretized into time slots of fixed duration Δt and we refer to t_i to indicate the i-th time slot. To perform this inference, we need an estimator Θ that models both the spatial and temporal dependencies between target user u and the other users within the OSN. In the following formula,

$$\hat{l}_{t_i}^{(u)} = \Theta(l_{t<t_i}^{(u)}, \mathbf{l}_{t\leq t_i}^{(F)}) \tag{3.2}$$

$\hat{l}_{t_i}^{(u)}$ is the estimated location of the tweet published by u at time slot t_i; $l_{t<t_i}^{(u)}$ is the set of geo-tags published by u before time slot t_i and $\mathbf{l}_{t\leq t_i}^{(F)}$ is the set of geo-tags published by other users up to time slot t_i. Notice that we include also the geo-tags provided by other users within time slot t_i itself because we expect these to be highly-informative

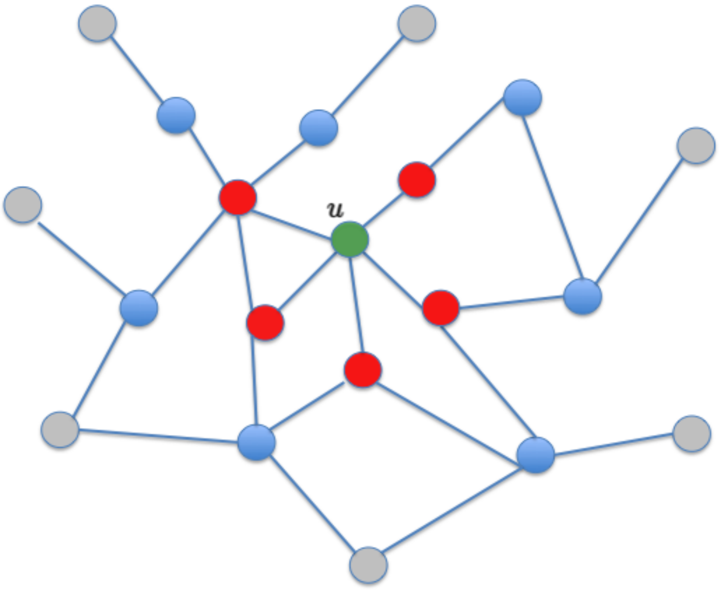


Figure 3.1: Example of social proximity for the valid users selection phase

of the current location of u (as we explained above in the example related to Alice). In the next subsection, we describe how we take into account both temporal and social proximity to design the estimator Θ.

3.2.2 Geo-tags Selection

From a theoretical standpoint, the estimator Θ could consider the whole historical information available in the OSN, i.e., all geo-tags published up to t_i. However, this amount of data grows linearly with the number of users and with the considered period of time. Hence, this approach cannot scale to large instances of OSN (which can count up to several hundreds of millions of users in real scenarios). Also, most of the geo-tags are expected to be uninformative with respect to the location of u at time t_i and can be safely discarded. In particular, tweets that have been published (i) far away in time and (ii) by users with weak social relationships with u are expected to be the least relevant ones [180]. To reduce the volume of available data, we perform a data selection process that undergoes two subsequent phases, namely the valid slots selection and the valid users selection.

In the*valid slots selection* phase, we select the last T time slots before t_i when user u has published at least a geo-tagged tweet. Once this phase has been performed, in the *valid users selection* phase, N valid users are chosen, in each valid slot, among those who have published a geo-tagged tweet within that slot. The valid users selection phase is carried out privileging users with high social proximity with u. We denote as social proximity the distance, in terms of hops, between users in the graph G (defined in Section 3.1). In the example in Fig. 3.1, we represent the social network of u (green node), where edges represent social connections in G. Nodes in red are 1-hop away from u, nodes in blue are 2-hops away from u, and nodes in grey are 3-hops away from u. According to the concept of social proximity defined above, we first select the users that are 1-hop-away from u (e.g., red nodes in Fig. 3.1). If the number of such users is less than N, the process is carried out for users that are 2-hops-away from u (e.g., blue nodes in Fig. 3.1), etc. The process concludes when N valid users are found. We refer to this set of users as *neighbors*. It should be noticed that when it is not possible to assign u a set of N valid neighbors, e.g., in the case of disconnected network components, we assign specific missing values to $\mathbf{l}^{(F)}_{t\leq t_i}$, as detailed in the following subsection. We stress the fact that other users' geo-tags are considered if provided in valid slots, i.e., when also the target user u has published at least one geo-tag. This choice is motivated by the objective of learning a consistent spatio-temporal dependency between u and her/his neighbors. Notice also that the N selected users differ at each time slot, as not in every time slot the same users have geo-tagged their tweets. In the next subsection, we present the deep-learning model that we use to learn the estimator Θ.

3.2.3 Deep Learning Model for Geo-Location Privacy Measurement

The deep learning architecture proposed to learn the estimator Θ is trained separately for each user. The architecture has two main inputs, namely the information relative to the geo-tags published by the target user before time slot t_i and by her/his neighbors up to time slot t_i (included). The model is trained to map these inputs to the output, i.e., the geo-tag of the target user at time slot t_i. Notice that the generic geo-tag l_{t_i} is represented as a vector with 2 components, the first for the latitude and the second for the longitude. The architecture is designed to initially process its inputs separately and to perform a successive downstream elaboration of their representations. This approach is common in the machine learning community to learn an effective representation of the inputs [138]. To realize such design objective, the deep learning architecture is composed of the following four main building blocks: i) *Target User Transform*, ii) *Neighbors Aggregator*, iii) *Concatenator*, and iv) *Regres-*

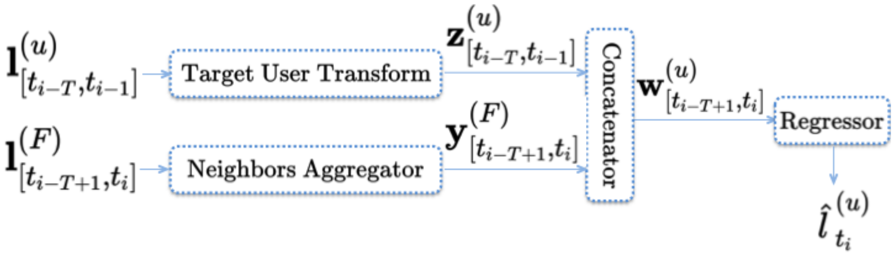

$\mathbf{l}^{(u)}_{[t_{i-T},t_{i-1}]}$
Target User Transform
$\mathbf{z}^{(u)}_{[t_{i-T},t_{i-1}]}$
$\mathbf{l}^{(F)}_{[t_{i-T+1},t_i]}$
Neighbors Aggregator
$\mathbf{y}^{(F)}_{[t_{i-T+1},t_i]}$
Concatenator
$\mathbf{w}^{(u)}_{[t_{i-T+1},t_i]}$
Regressor
$\hat{l}^{(u)}_{t_i}$

Figure 3.2: Overview of the deep learning architecture

sor. A representation of the architecture is depicted in Fig. 3.2. The transform and aggregator blocks perform a first processing of previous geo-tags of the target user and her/his neighbors, respectively; the concatenator simply juxtaposes the outputs of the previous processing and provides a single input for the regressor; finally, the downstream regressor returns the inference of the target user's location. The overall deep learning architecture is trained to minimize the Mean Squared Error (MSE) between the estimated and actual geo-tags of user u in the selected slots. Each block of the deep learning architecture is described in detail in the next paragraphs.

Target User Transform This block takes as input the sequences of u's geo-tags provided in the last T valid time slots, i.e., $l^{(u)}_{t_{i-T}}, ..., l^{(u)}_{t_{i-1}}$. Each geo-tag is processed separately by a feed-forward neural network, which returns an abstract representation of its input, denoted as $z^{(u)}_{t_{i-T}}, ..., z^{(u)}_{t_{i-1}}$. The considered neural networks are independent, i.e., they do not share any parameter. The output of this block is a sequence of T elements containing the representations of the locations geo-tagged by u in the selected T valid slots.

Neighbors Aggregator This block computes an overall representation of the information relative to the N neighbors who have been previously selected (in the *valid users selection* phase, described in Section 3.2.2) in each valid time slot, i.e., $\mathbf{l}^{(F)}_{t_{i-T+1}}, ..., \mathbf{l}^{(F)}_{t_i}$. Notice that $\mathbf{l}^{(F)}_{t_k}$ is the list of all the geo-tags provided by the selected users within the k-th time slot. The list corresponding to each selected slot is individually processed by a Long Short-Term Memory (LSTM) [126] and then by a feed-forward neural network. The choice of the LSTM (introduced in Section 2.6.1.1) in this phase is inspired by [122], in which it is suggested to use it as a tool to learn the representation of a node's neighbors within a graph. In our approach, topological information is considered since, as explained in Section 3.2.2, neighbors are chosen according to their proximity with respect to the target user. Apart from this, the LSTM guarantees higher expressive capabilities with respect to other data aggregation methods, e.g., mean aggregator [122]. The output of this block is denoted as $\mathbf{y}^{(F)}_{t_{i-T+1}}, ..., \mathbf{y}^{(F)}_{t_i}$, which is a sequence of T elements containing the representations of the locations geo-tagged by the selected neighbors in the T valid slots.

Concatenator This block receives in input two sequences of T elements, i.e., $\mathbf{z}^{(u)}_{[t_{i-T}, t_{i-1}]}$ and $\mathbf{y}^{(F)}_{[t_{i-T+1}, t_i]}$, which encode the representations of the geo-tags provided in the selected T slots by the target user and by the N neighbors, respectively. The concatenator block juxtaposes the elements in corresponding positions of the sequences and returns a single sequence of T elements, denoted as $\mathbf{w}^{(u)}_{[t_{i-T+1}, t_i]}$, which can be processed by the downstream regressor.

Regressor The final block is composed of a LSTM and feed-forward neural network module. This block processes $\mathbf{w}^{(u)}_{[t_{i-T+1}, t_i]}$ (i.e., the output of the concatenator) and returns $\hat{l}^{(u)}_{t_i}$ (i.e., the inferred geo-location). Notice that, even though the architecture of this block is quite similar to that of the neighbors aggregator block, the design strategies behind them are significantly different. In fact, in this phase, we want to exploit the ability of the LSTM to learn recurrences within a sequence of temporal data, which are useful to perform the inference task.

The proposed deep learning architecture returns an estimate location $\hat{l}^{(u)}_{t_i}$ for each geo-tagged tweet shared by user u. Finally, the privacy of u is measured according to Eq. (3.1), i.e., by computing the geographical distance between actual and estimated locations. We remark here that, to compute users' location privacy, we use as ground truth the locations actually shared by the users on Twitter. However, we have hidden these geo-tags in the location inference task as they represent the target values that the attacker aims to estimate.

3.3 Control of Privacy Level

In this Section, we describe two strategies that users can implement to enhance their privacy, i.e., to reduce the ability of an attacker to correctly infer their location. Then, as the measured privacy is likely to be affected by other factors beyond data perturbation, we also propose a privacy model that captures the impact on privacy of several users' behavioral characteristics (e.g., data perturbation level and users' mobility) and that can therefore be employed as a more comprehensive privacy control tool.

3.3.1 Strategies to Tune the Level of Privacy

Every time a user publishes a tweet, she/he can decide to either provide it with a geo-tag or not. In this respect, each user is characterized by a particular behavior, which can be defined as the percentage of tweets that she/he normally geo-tags. As location privacy can be violated by relying on the information released by other OSN users, to prevent this violation and preserve the secrecy of their location, users may purposely alter the geo-tags of a portion of the tweets they normally publish. We refer to this strategy as *data perturbation* and we introduce a variable p, the data perturbation probability, i.e., the probability that a user deviates from her/his normal behavior. For instance, a user who publishes, on average, 10 geo-tags per month, can set $p = 0.3$ and reduce the number of normally geo-tagged tweets to 7. The remaining 3 tweets are perturbated. We propose to use two data perturbation strategies, namely *data obfuscation* and *data removal* strategy and we describe them in the following:

- **Data Obfuscation**: According to this strategy, a geo-tag is shared, with probability p, by randomly selecting a location within the boundaries of the city where the user is sharing content (New York City, in our dataset).
- **Data Reduction**: Following the data reduction strategy, with probability p, a user does not geo-tag a tweet that would have been provided with a location if no perturbation were applied.

Other approaches to perturb geo-tags can also be considered, e.g., to delay the time in which the tweet is published. This, and other possible techniques to perturb geo-tags, will be evaluated in future work. Overall, the rationale of the perturbation strategies is that, by increasing p, i.e., the level of data perturbation, a user can improve her/his privacy. However, we shall still provide a quantitative answer to an important pending question: how much privacy a user should expect to gain by increasing p? A user who is interested to have a quantitative assessment of her/his privacy can follow the approach described in Section 3.2.3 to infer her/his visited locations (as an attacker would do) and, from this, evaluate her/his own level of privacy. This approach is effective to estimate users' expected level of privacy but has several drawbacks. First of all, to assess the impact of the data perturbation level on the resulting privacy, the user has to train and test the deep learning model using data perturbed for several values of p. This process, besides being highly-time consuming, does not allow users to understand the impact that several factors (e.g., related to her/his mobility behavior)

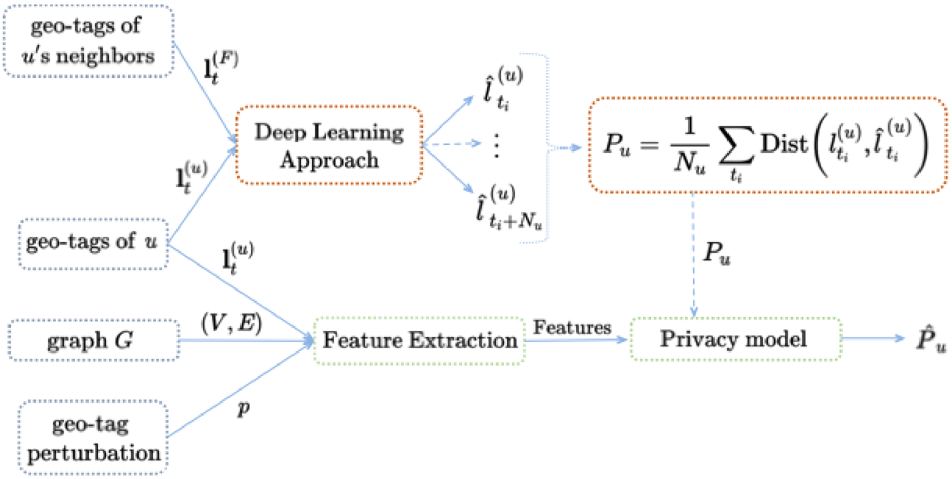


Figure 3.3: Block diagram of the approach followed to learn the privacy model

beyond p have on her/his privacy. In fact, this model takes as input only raw data (i.e., geo-tags) that do not explicitly capture the characteristics of users' behavior. Moreover, there is no possibility to interpret the impact of each feature on privacy, since the output of the model is itself a location (and not a value of privacy). Lastly, deep learning models are widely considered as black boxes whose outcomes are difficult to interpret in relation to the input features [116, 158]. Hence, the proposed deep learning approach is only capable to measure privacy, but it does not give users a proper understanding on how to control it. In the next subsection, we describe a *privacy model* that quantifies the impact of various factors on privacy and allows to directly measure its level (i.e., without performing an intermediate location inference).

3.3.2 Privacy Model

In this subsection, we present a privacy model that provides users with an estimate of their privacy level given a set of features that explicitly capture their characteristics and behavior. This model is based on a machine learning algorithm trained in a supervised manner to map diverse users' features to their value of privacy. Machine learning algorithms are commonly trained using ground truth values. In our case, however, the target privacy cannot be regarded as a ground truth value in the traditional sense. In fact, privacy is not an attribute of the users, but rather a value derived from the estimation of their location, which in turn depends on many factors (e.g., the ability of the attacker, amount of available data, etc...). In this study, we consider as target value the measurement of privacy obtained with the deep learning approach explained in Section 3.2.3, as represented in Fig. 3.3. Notice that, in principle, any estimation of privacy could be used as a target value P_u.

We train several machine learning algorithms (e.g., random forest and decision tree) to obtain an estimator of users' privacy (results in Section 3.5.3). These algorithms are trained to minimize the Mean Absolute Error (MAE) with respect to the target privacy value. The MAE is defined as:

$$MAE = \frac{1}{|V|} \sum_{u \in V} |P_u - \hat{P}_u|, \tag{3.3}$$

where P_u and $\hat{P}_u$ are the target privacy value of user u (computed according to Eq. (3.1)) and the privacy value estimated by the privacy model, respectively. $|V|$ is the total number of users present in our dataset.

To obtain the privacy model, such machine learning algorithm is fed with a set of features, which are listed in Table 3.1. These features cover a broad range of parameters that users can tune to control their privacy, and fall into three main categories, namely *mobility-related*, *topology-related*, and *data-perturbation-related*. Mobility-related features are considered to statistically describe the mobility (e.g., in terms of variability of the visited locations [190]) of the users. Topology-related features aim to provide a characterization of users' position within the social network (e.g., in terms of the network centrality measures described in Section 2.7.3). The data-perturbation-related feature is p, i.e., the level of data perturbation the user is willing to adopt for tuning her/his privacy.

Given these features for a certain user, the model outputs an estimate of her/his location privacy. If, for example, a user is not satisfied with her/his privacy level, she/he can evaluate how the secrecy of her/his information enhances by varying her/his sharing activity (e.g., by geo-tagging less frequently), changing her/his social clique (e.g., by diminishing the number of friends in the OSN), or perturbing her/his geo-tag (i.e., by increasing the data perturbation probability p). We envision the utilization of this model in a client-server scheme, where the client is a user who requires an estimate of her/his privacy, while the server is a third-party entity that offers a service of privacy awareness. Notice that this approach is also privacy-preserving since it

Table 3.1: Features of the user's geo-location privacy model

Category	Feature	Description
Mobility-related	tweet frequency	Frequency of geo-tagged tweets per day
	avg. distance	Average distance of geo-tags
	var latitude (longitude)	Variance of the geo-tags latitude (longitude)
	corr. latitude (longitude)	Autocorrelation of the geo-tags latitude (longitude)
	median latitude (longitude)	Median of the geo-tags latitude (longitude)
	kurtosis latitude (longitude)	Kurtosis of the geo-tags latitude (longitude)
	skew latitude (longitude)	Skewness of the geo-tags latitude (longitude)
Topology-related	no. of friends	Number of friends
	PR	PageRank
	deg-centrality	Degree centrality measure
	eig-centrality	Eigenvector centrality measure
	cl-centrality	Closeness centrality measure
	bw-centrality	Betweeness centrality measure
Data perturbation-related	p	Data perturbation probability

requires users to disclose to the server only the aforementioned features, which, as detailed in Table 3.1, represent aggregate information (e.g., the variance of their visited locations) and do not need the sharing of the visited locations.

3.4 Experimental Setup

In this Section, we show the setup of our experiments for measuring and controlling users' geo-location privacy in OSNs. We first describe the data employed in our analysis and we then detail the implementation of the proposed deep learning architecture.

3.4.1 Twitter Data

We perform our evaluations on the Twitter dataset presented in [207], which includes the information about social connections among users (i.e., pair of followees, followers) and $2,173,681$ tweets collected within 100 kilometers from New York city center for 31 days. Figure 3.4 shows the spatial distribution of the tweets over all the collection period. In Table 3.2, we summarize the statistics about the data and network properties related to the social graph. Notice that we detailed the network properties (e.g., average degree, diameter, etc.) in Section 2.7.

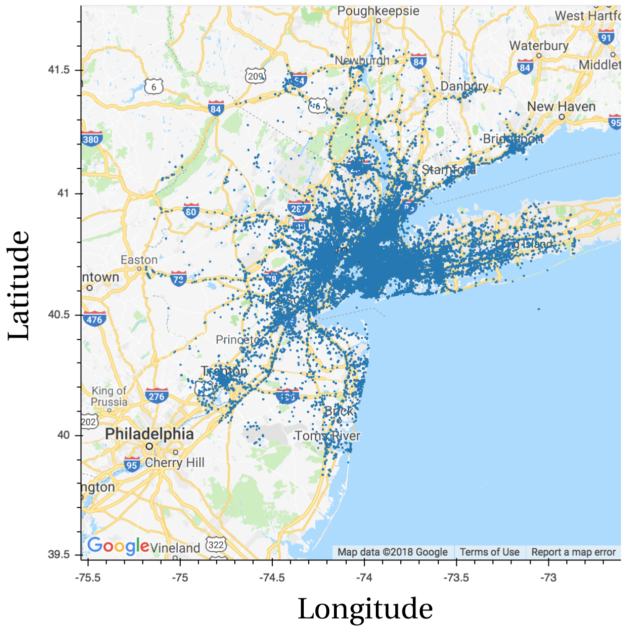


Figure 3.4: Geographical distribution of the geo-tagged tweets in New York City

Table 3.2: Statistics of the Twitter dataset

	New York City Dataset
Users $\lvert V\rvert$	6,082
Friendship relationships $\lvert E\rvert$	31,874
Average degree $\langle k\rangle$	10.22
Diameter d_{max}	19
Average clustering coefficient $\langle C\rangle$	0.15
Density D	0.001

3.4.2 Experimental Settings

We evaluate the privacy of a user (defined as the geographical distance between her/his actual and estimated location) at a given time slot t_i, given the information of the geo-tags published on the past T slots. After preliminary evaluations, we found that $T = 3$ was the best compromise between accuracy and training time of the deep learning algorithm. Each slot represents a period of 3 hours, which is the average time between two consecutive tweets in the dataset. Each location published during the slots is expressed as a pair of latitude and longitude. Both latitude and longitude have been normalized according to the standardization technique, which is widely employed to perform feature scaling in machine learning.

The deep learning architecture includes a *target user transform* (i.e., a feed-forward neural network composed of a single layer with 5 neurons), a *neighbors aggregator* (i.e., a LSTM layer with 5 neurons followed by a feed-forward neural network layer with 2 neurons) and a regressor (i.e., a LSTM layer with 5 neurons and a feed-forward neural network layer with 2 neurons). Among the configurations of hyper-parameters that we have considered, the aforementioned one proved to achieve the best balance between inference effectiveness and training time efficiency. The deep learning model is trained to estimate the location of the target users at time slot t_i from the previous T geo-tags provided by the target user u (up to time slot t_{i-1}) and those of the N neighbors (up to time slot t_i) chosen in the valid users selection phase. The number of neighbors N is set to 10, which is the approximate network average degree (see Table 3.2). The first 75% of the time slots are used for training purposes, while the remaining 25% are used to test the learned model. The 75% of training slots are further divided into pure training slots (60%) and validation slots (40%).

3.5 Evaluation

This Section shows the results of the presented approach to measure users' geo-location privacy and provides an evaluation of the proposed countermeasures and methodologies to control the public exposure of users' locations.

3.5.1 Privacy Measurement

In this subsection, to address RQ 1.1, we present an overview of the results related to the privacy measurement obtained by using the proposed deep-learning approach described in Section 3.2.3. To motivate the use of a machine learning strategy, we firstly show the average location inference error obtained using several alternative approaches. In particular, we propose three baselines that perform the location inference of a target user based on the same inputs of our deep-learning algorithm, i.e., her/his most recent available location and the geo-tags of other users. Specifically, the three baselines compute a value $\hat{F}_{loc}$ from the geo-tags of $N = 10$ target user's neighbors as follows:

- *mean-based*: $\hat{F}_{loc}$ is obtained by computing the average latitudes and longitudes of the N neighbors.

Table 3.3: Comparison between the deep-learning and the baseline approaches to infer users' locations

Approach	Average Error [km]
Deep Learning	2.3
Median-Based	7.3
Mean-Based	7.7
Cluster-Based	9.2

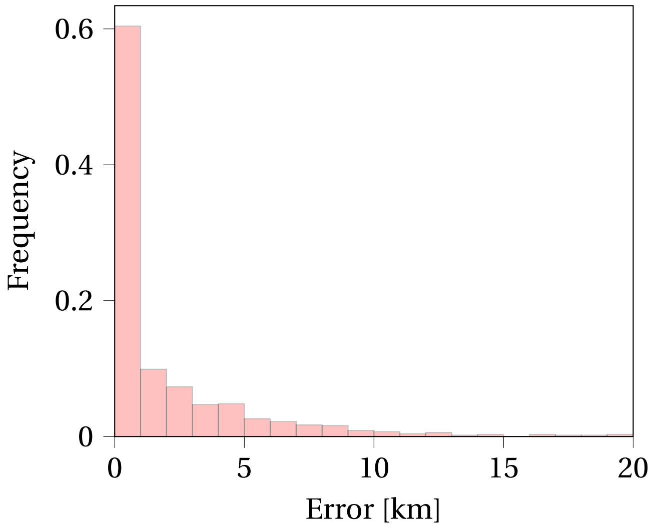


Figure 3.5: Distribution of the location privacy measurement

- *median-based*: $\hat{F}_{loc}$ is obtained by computing the median value of latitudes and longitudes of the N neighbors.

- *cluster-based*: the geographical area considered in our dataset is divided into a set of non-overlapping squares with sides equal to $\sim$ 155m. $\hat{F}_{loc}$ is the centroid of the square mostly visited by the neighbors.

For each baseline, the location of the target user is then estimated by computing the average between her/his most recent available location and $\hat{F}_{loc}$. In Tab. 3.3, we show the average error of the location inference obtained using the deep-learning approach and the three baselines. As expected, the deep-learning algorithm significantly outperforms all the considered alternatives, which confirms its ability to capture complex relationships among the input data.

To further elaborate on the results obtained with the deep-learning strategy, we show the distribution of the location privacy in Fig. 3.5. It can be noticed that 60% of the

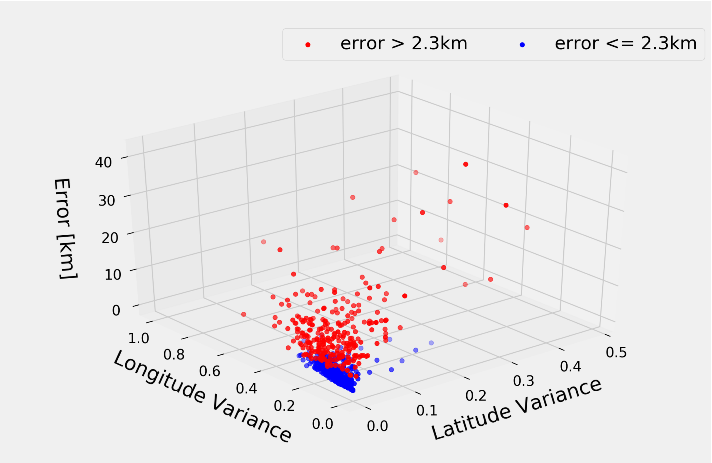


Figure 3.6: Geo-localization error vs variance of mobility

users have a level of privacy below 1km and almost 80% of the users achieve a privacy measure below 3km. Overall, the average privacy measure is of 2.3km and the median privacy value is of 0.4km. Such results corroborate our intuition that the secrecy of the location information can be violated by leveraging publicly-available information shared by other OSN users.

To better understand this result, in Figure 3.6, we depict the privacy measure as a function of variances of latitude and longitude visited by each user. Users whose privacy is lower (resp. higher) than the average (2.3km) are colored in blue (resp., red). The objective is to understand whether the variability of the visited locations can impact users' privacy. We expected that the variance of the locations visited by the user could be a strong indicator of user's privacy leakage. However, a subset of users with limited mobility achieves a privacy level above the average, suggesting that mobility variance is not the only factor determining the level of privacy risks. We further analyze other factors impacting users' privacy in Section 3.5.3.

3.5.2 Data Perturbation Strategy

The results related to the privacy assessment indicate a considerable peril for users' privacy in OSN. To respond to RQ 1.2, we now evaluate the effect of two possible countermeasures to increase the location error of attacker's estimates. Thereby, we examine the data perturbation strategies introduced in Section 3.3.1. In Fig. 3.7a, we depict the percentile of the geo-location error with varying data perturbation probability p related to the data obfuscation strategy. Notice that the geo-location error

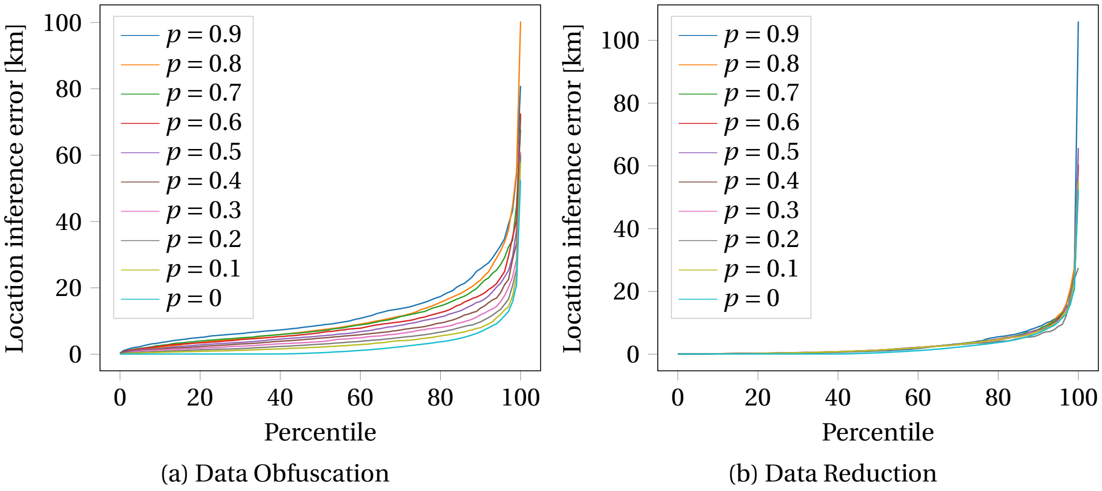


Figure 3.7: Percentile of the geo-localization error using data perturbation strategies

corresponds to the definition of privacy provided in Section 3.2.1. We first observe that the percentage of perturbed geo-tagged tweets significantly affects the ability of the attacker to correctly infer the geo-tags. In fact, as expected, the localization error increases with increasing p, i.e., users' privacy increases as the level of perturbation increases.

On the other hand, as shown in Fig. 3.7b, the data reduction strategy does not improve users' privacy as much as the data obfuscation strategy. The gap between the percentiles with varying p is very small, i.e., there is no significant privacy improvement with respect to the unperturbed scenario ($p = 0$). This result suggests that even a small percentage of uncorrupted information is sufficient to effectively estimate users' location. This is further confirmed in Fig. 3.8, which compares the average localization error of the perturbation strategies with varying p. Interestingly, in the data obfuscation strategy, users' privacy grows linearly with p, whereas the localization error is almost constant around 3.5km using the data reduction strategy. We observe that, using the data obfuscation strategy, privacy increases proportionally with the data perturbation level, which can in turn be tuned according to the desired level of privacy. This result suggests that data obfuscation is preferable with respect to data reduction.

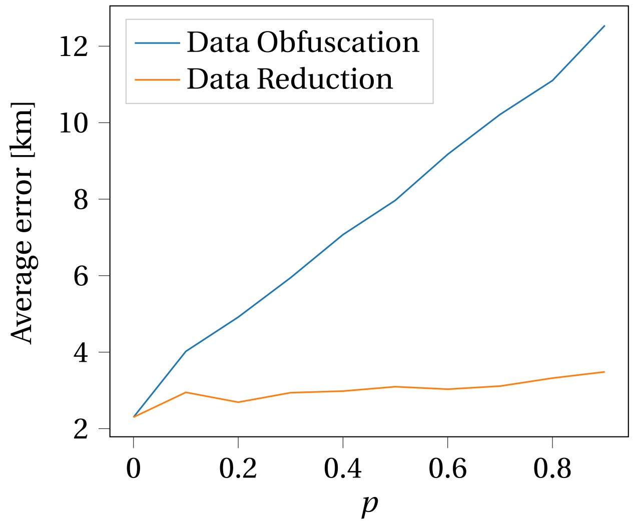


Figure 3.8: Comparison of data perturbation strategies considering the average geo-localization error with varying p

Table 3.4: Performance of the privacy model for several machine learning algorithms

Algorithm	Error [km]
Random Forest	1.4
Decision Tree	2.1
Gradient Boosting	3.0
k-nearest neighbors	3.7
Ridge Regression	3.8
Support Vector Machine	4.1

3.5.3 Validation of the Privacy Model

In this subsection, to answer RQ 1.3, we show the results on the privacy model described in Section 3.3.2. Specifically, we evaluate the ability of this model to estimate users' privacy and we examine the impact that users' behavior has on their level of privacy. To obtain an estimator of users' privacy, we train and compare several machine learning algorithms. In particular, we train and test every algorithm by following a 10-fold cross validation approach.

Table 3.4 shows the MAE for the different considered approaches. For further evaluations, we consider the model that yields the minimum MAE, i.e., the model based on a random forest. This machine learning algorithm also allows us to study the relevance of each feature in the assessment of users' privacy. In fact, the random forest algorithm computes and provides as an output the importance (or relevance) of each feature

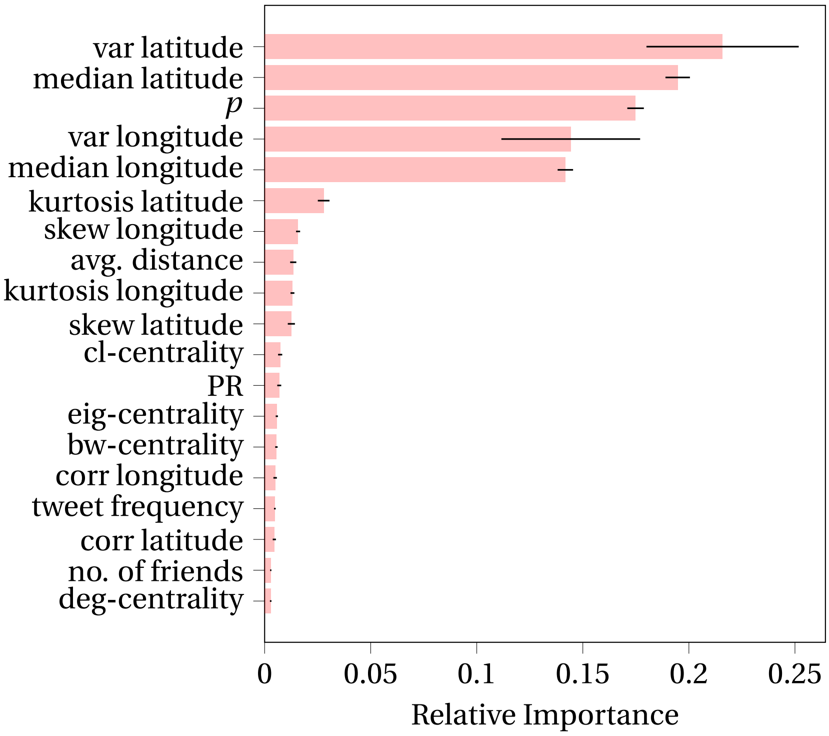


Figure 3.9: Features importance of the model based on random forest

in the estimation of the target variable. We present the feature importance of the random forest in Fig. 3.9. First of all, we observe that mobility-related features, e.g., variance and median value of the visited locations, are the most relevant features to assess users' level of location privacy. This can be explained considering that mobility patterns strongly affect the predictability of the visited locations. Perhaps surprisingly, the median of the visited locations highly influences the outcome of the model. This may suggest that some locations are strong indicators of the privacy of a user, which is in line with the findings of [29]. It is also noticeable that the variables (e.g., variance and median) related to the visited latitudes have a larger impact on the outcome with respect to the variables related to the visited longitudes. Arguably, this is due to the nature of users mobility within the specific urban area under analysis. Another very relevant feature is the level of data perturbation, i.e., p. This confirms the discussion done in Section 3.3.1, where the importance of tuning p to increase privacy has been highlighted. Finally, we notice that topology-related features have the least significant impact on the estimation of privacy, suggesting that the location privacy of a user is quite unrelated to her/his position in the social network.

Until now, we have considered the MAE between estimates of privacy (i.e., $\hat{P}_u$) and

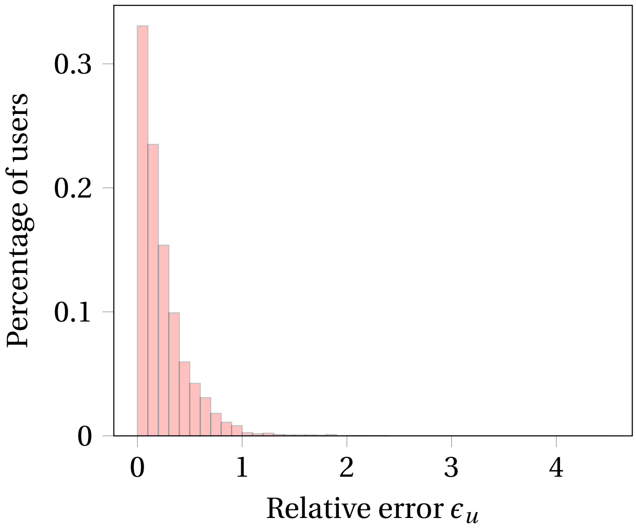


Figure 3.10: Distribution of the relative error ϵ_u between estimates and ground-truth values of privacy

corresponding reference privacy values (i.e., P_u) as the only metric to evaluate the quality of the privacy model. However, the MAE does not consider the impact of each estimate's error on its actual reference privacy value. For example, an absolute error of 1km is much more significant if $P_u = 3$km with respect to the case where $P_u = 50$km. To provide a more complete evaluation of the actual ability of the model to correctly estimate privacy, we propose two model validation strategies. The first one is the analysis of the *distribution of the relative errors* between estimates and target privacy values. The second one is the *Quantile-Quantile (Q-Q) plot*, which graphically shows the likelihood that two populations have been generated by the same model. The two validation strategies are explained in detail in the following.

3.5.3.1 Distribution of relative errors

We now present the distribution of relative errors between the target privacy values and the privacy estimates computed by the random forest model. The relative error of the privacy of user *u*, i.e., ϵ_u is defined as:

$$\epsilon_u = \frac{|P_u - \hat{P}_u|}{P_u}. \tag{3.4}$$

We depict the distribution of the relative error in Fig. 3.10. Notice that the relative

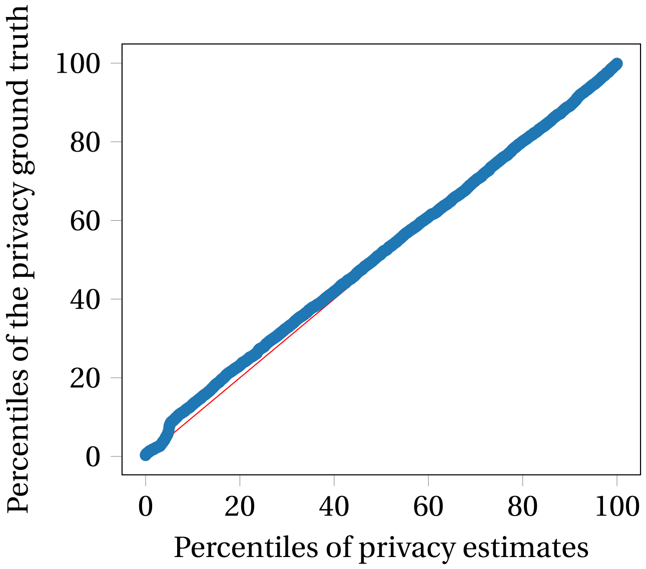


Figure 3.11: Q-Q Plot

error provides much more valuable information with respect to the MAE shown in Table 3.4 because it considers the error in relation to the reference privacy value (i.e., P_u). From Fig. 3.10, it can be noticed a low relative error for most of the users. More specifically, for around 33% of the users, the random forest estimates the privacy with a relative error below 10%. This percentage grows to 50% if a relative error below 20% is considered.

3.5.3.2 Q-Q plot

The *Q-Q plot* allows us to graphically assess the likelihood that two populations have been generated by the same model. In our case, the elements of the two populations correspond to the privacy values estimated by the random forest model and the target values, respectively. Each element of the population is represented in the Q-Q plot as a point whose coordinates are the corresponding percentiles of the two populations. If the data were generated from the same model, each point would lie on a line of slope equal to 1 passing through the origin (represented in red in Fig. 3.11). In Fig. 3.11, it is possible to observe that most of the points (in blue) lie in the proximity of this line (in red). The obtained results suggest the use of the privacy model as a reliable tool to estimate users' privacy based on their characteristics.

3.6 Conclusions

In this Chapter, we focus on the problem of geo-location privacy in OSNs, considering Twitter as a study case. We address this problem from two different angles: on one side, we develop methods to assess the ability of an attacker to correctly infer users' locations; on the other side, we propose effective strategies that users can adopt to measure and control their privacy.

To address RQ 1.1, we propose a deep learning model that can accurately infer users' locationS by only considering publicly-available geo-tags. From this estimate, we measure location privacy as the geographical distance between the inferred and the actual position. Our experiments confirm the concerns about privacy perils in OSNs. In fact, we showed that 60% of the users have a level of location privacy below 1km and almost 80% of the users achieve a privacy measure below 3km.

To respond to RQ 1.2, we propose data perturbation techniques that users can use to decrease the knowledge obtainable by analyzing their published geo-tags. We measure the effectiveness of such strategies considering their ability to increase users' privacy. In particular, the obfuscation of the actual location proved to be the most effective strategy. In fact, we observe that, using the data obfuscation strategy, users' privacy grows linearly with the perturbation probability p, whereas privacy does not improve significantly using the data reduction strategy. This result suggests that data obfuscation is preferable with respect to data reduction, as it allows to tune the desired level of privacy by varying the data perturbation level.

To further increase users' control and answer RQ 1.3, we model privacy considering features that capture users' behavior (e.g., characteristics related to the mobility of the user and to the enforced level of data perturbation). This model, based on the random forest algorithm, provides an accurate estimate of privacy (with an average error of about 1.4km) and enables a principled understanding of the features that mainly affect it. Features related to the mobility of the user and to the enforced level of data perturbation resulted to be the most relevant factor behind the infringement of users' location secrecy. Finally, to provide a more complete evaluation of the ability of the privacy model to correctly estimate privacy, we employ two model validation strategies (i.e., distribution of the relative errors and the Q-Q plot), whose results confirm the validity of the privacy model as a privacy estimation tool.

4
Social Influence Modeling

4.1 Introduction

Chapter 3 presented an example of privacy leakage in OSNs. In particular, we showed that users' geo-location can be accurately estimated by relying only on spatial and temporal dependencies among individuals. In this Chapter, we analyze the privacy issue in OSNs from a different point of view. We study the social influence phenomenon, a key factor that governs human behavior and drives individual decisions. The key idea behind social influence is that the interaction with other individuals (or a group) may affect subjects' behavior. In line with this concept, it might be possible to predict human behavior in new (and unseen) instances by modeling the flow of influence among users, i.e., the influence strength between pairs of subjects.

According to this idea, the goal of this Chapter is to understand whether the modeling of social influence can effectively provide a means to infer users' future behavior and, in turn, violate their privacy. As we described in Section 2.3, in this thesis, we focus on modeling social influence at the user-level as we are interested in measuring whether and to what extent an individual is influenced by other subjects. In Sections 1.3.2

and 2.3, we detailed the limitations of existing approaches and we highlighted how such solutions consider social relationships as the only factor impacting influence dynamics. To investigate such open points, in this Chapter, we seek to answer the following RQs:

RQ 2.1: *How is it possible to overcome the limitations of existing social influence models?*

RQ 2.2: *Can other factors (e.g., location) impact influence modeling other than social relationships?*

To address RQ 2.1, we introduce Social Influence Deep Learning (SIDL) [165], a framework that combines deep learning with network science [8, 49, 184] to model social influence and predict human behavior. SIDL only leverages dyadic social interactions between subjects to predict their behavior and provides a model that can be generalized to any kind of OSN and, less specifically, to any real-life domain. Moreover, we propose different approaches at varying social network granularity (i.e., the ego network of each user, the community within each user is embedded, and the totality of the social network) with the objective of facing two typical challenges of deep learning: interpretability (introduced in Section 2.6.1) and scalability (defined in Section 4.2.1).

To respond to RQ 2.2, we explore the role of factors such as geographical proximity, homophily (defined in Section 2.7), and social connections in the social influence phenomenon. In particular, we investigate the role of different communities related to such factors as sources of social influence. The rationale is that subjects may follow and be affected by the collective behavior of individuals (i) living in their physical area, (ii) with common interests, or (iii) in their social community.

This Chapter is organized as follows. In Section 4.2, we describe the SIDL framework, its three approaches, and corresponding results. Section 4.3 introduces a novel interpretation of physical, homophily, and social community as sources of social influence. Finally, Section 4.4 discusses the correlation between social influence and users' privacy presenting results and some notes of caution about the risks of sharing sensitive data.

4.2 Social Influence Deep Learning

In this Section, to respond to RQ 2.1, we present SIDL [164, 165], a deep learning framework that aims to model social relationships and, accordingly, infer human behavior, in terms of performed activities. Although SIDL can be generalized to every kind of human activity (both online and offline), we focus on real-world (offline) activities, such as attending an event or visiting a location. To the best of our knowledge, our work [165] is the first effort that seeks to provide a social influence model of real-life activities. For this purpose, we analyze data from Plancast [160], an Event-Based Social Network (EBSN), and Foursquare [27], a Location-Based Social Network (LBSN). In particular, Plancast allows users to create social events (e.g., concerts, conferences, etc.) and share their participation in these events, while in Foursquare users can register at named places (e.g., restaurants, theatre, etc.) and share their location with their friends. Such platforms provide remarkable opportunities to analyze users' real-world behavior and interactions from the lens of OSNs.

SIDL addresses the limitations of existing approaches (see Section 2.3) by learning the interplay among subjects by means of DNNs (described in Section 2.6.1), a class of machine learning algorithms inspired by biological nervous systems. The rationale of SIDL is to represent each user in the social network as an input node of the DNN, which in turn has the capability to automatically model relationships embedded in the input [125]. In SIDL, we consider different models by varying the network structure in which the user is embedded with the objective of addressing two typical challenges (detailed in the next subsection) of deep learning models: interpretability and scalability.

4.2.1 DNN Challenges

As mentioned in Section 2.6.1, the *outcome explanation problem* is the notion of interpretability that we study in this Section to model social influence among OSN users. Given the output of a black box predictor, the outcome explanation problem consists in reconstructing an explanation for it. One of the most common techniques to solve the outcome explanation problem is to focus on explaining what a neural network depends on *locally*, instead of trying to understand the full mapping learned by the model. This *local explanation* aims to predict the response of the predictor in a neighborhood of a given input. One of the techniques generally used to accomplish this purpose is Saliency Mask (SM). SM aims to explain the DNN outcome by identifying a subset of the input, which is mainly responsible for the prediction. As an example, in [91], SM is used to highlight the salient part of the images that causes a certain outcome.

While interpretability is a more rational problem, scalability represents a practical issue related to DNN implementation. The idea is that the social influence model should be able to adapt and scale to previously unseen users. In this sense, scalability issues may occur when new users register to a social network. A new user can perform activities, create social connections with other OSN users, and influence them. In such a case, the social influence model should be adjusted in order to consider new users in the social network. Our objective is to mitigate this issue in the most efficient way, in terms of time and resource consumption, while preserving performance.

Towards meeting these challenges, in this Section, we propose to combine network science (see Section 2.7) with deep learning. We consider different models by varying the network structure in which the user is embedded. We first describe the Global-SIDL approach, which considers the whole social network in a unique model. We then narrow our approach to the ego network of each individual and build a *local* model (Local-SIDL) for each user. Finally, we propose to decompose the social network in mesoscale level structures (introduced in Section 2.7), such as communities, so as to consider a larger social structure within each user is embedded.

4.2.2 Problem Definition

Let $G = (V, E)$ be an undirected graph representing the social network, where $V = \{u_1, u_2, \ldots, u_N\}$ is the set of users, and E is the set of edges that connect them. The edge $(u_i, u_j) \in E$ indicates a social tie between u_i and u_j, which in turn are referred to as *friends*. We denote with $\mathscr{A}_l$ the *action log*: a record of the actions performed by every user in the social network. Each entry of the action log $\mathscr{A}_l$ is a tuple (t_i, u_i, a) representing the action a performed by user $u_i \in V$ at time t_i. Let A be the set of the actions performed in $\mathscr{A}_l$, for each action $a \in A$, each user is either active (if she/he performed the action) or inactive (otherwise). In accordance with [106, 208], we say that an action $a \in A$ propagates from u_i to u_j if the two following conditions are met:

- $(u_i, u_j) \in E$,

- $(t_i, u_i, a), (t_j, u_j, a) \in \mathscr{A}_l$ with $t_i \leq t_j$.

We consider the scenario where $t_i = t_j$ to take into account the mutual influence between users in performing an activity at a fixed time $t = t_i = t_j$, e.g., attending an event. Finally, we indicate with A_{u_i} the set of actions performed by u_i and with S_{a,u_i} the set of active friends of u_i for the action a.

We keep track of users' activities over time to model complex social relationships among them. In particular, we focus on the comprehension of the social influence phenomenon, with the final objectives of forecasting influence propagation in real-world scenarios and, at the same time, understanding whether social influence modeling can expose OSN users to privacy risks. The idea behind our social influence modeling is to learn influence strengths among subjects by leveraging the actions performed by users in their history and how such actions propagated between each other. Once the model is trained, we target to exploit it to predict users' real-life behavior based on other users' actions. More specifically, our objective is to infer whether users will perform action a based on their active friends S_{a,u_i}, $\forall a \in A$. The rationale of this approach is based on the concept of social influence itself. A subject may perform an action, e.g., to buy a new product or to watch a TV show, when her/his friends have performed the same action.

4.2.3 Methodology

In this subsection, we first present the rationale of SIDL and the logic behind the proposed solution. We then present the SIDL approaches and corresponding implementations.

4.2.3.1 Social Influence Deep Learning (SIDL)

In [164, 165], we introduced SIDL, a DNN-based framework for both modeling social influence and predicting user behavior. To the best of our knowledge, SIDL is the first approach that attempts to model social relationships by using neural networks. The rationale of this solution is based on the capability of a DNN to extract relationships embedded in the input data. Thereby, if we represent each user in the social network as an input node of the DNN, we can model the interplay and dependencies among users by leveraging their activity history. In particular, SIDL aims to overcome the drawbacks of previous works described in Section 2.3. In particular, existing models [106, 208] have two main drawbacks: (i) they assume that the probability of friends influencing a subject are independent of each other, and (ii) they do not consider the actions not performed by the subject (but performed by her/his friends) to learn the influence probabilities.

According to the SIDL approaches introduced in the next subsections, we concurrently take into account the relationship between a given subject (from now on *target-user*) and her/his friends and the interactions among them. This allows us to overcome the first drawback of existing approaches. Also, for each target-user u_i, we consider both positive (i.e., u_i's performed actions) and negative (i.e., actions not performed by u_i but performed by her/his friends) instances to model social influence. In particular, we formulate the task of predicting whether u_i will perform action a (based on her/his active friends S_{a,u_i}) as a binary classification problem, where the DNN output $y_{u_i,a}$ is a Boolean variable that is equal to 1 if the target-user u_i performed a, and is 0 otherwise. We train the DNN by leveraging the action log $\mathscr{A}_l$. More specifically, to train the DNN and tune the model parameters Θ, we rely on the history of both (i) the actions propagated from u_i's friends to u_i (i.e., positive instances) and (ii) the actions not propagated from u_i's friends to u_i (i.e., negative instances). Notice that we denote an action as *propagated* according to the definition given in Section 4.2.2. This, in turn, addresses the second limitation of existing works (i.e., only positive instances were used to learn influence probabilities). Once the DNN is trained, we utilize the influence model to predict the target-user's new (not performed yet) activities according to her/his friends' activities.

4.2.3.2 Global-SIDL (G-SIDL)

The first SIDL approach we present is based on the idea of modeling the entire social network in a unique neural network, thus, we name this solution *Global-SIDL* (G-SIDL). The rationale of G-SIDL is to have a unique model that includes every user in the social network to learn the interplay among individuals and model their interdependencies. To accomplish this purpose, we employ a DNN structured as follows: The input layer is composed of two concatenated vectors referred to as *target-user ID vector* ($\mathbf{v}_{u_i}^{ID}$) and *social network vector* ($\mathbf{v}_{u_i}^{SN}$), respectively. Both of them have length $N = |V|$.

The former ($\mathbf{v}_{u_i}^{ID}$) is a one-hot vector that uniquely identifies each target-user $u_i \in V$.

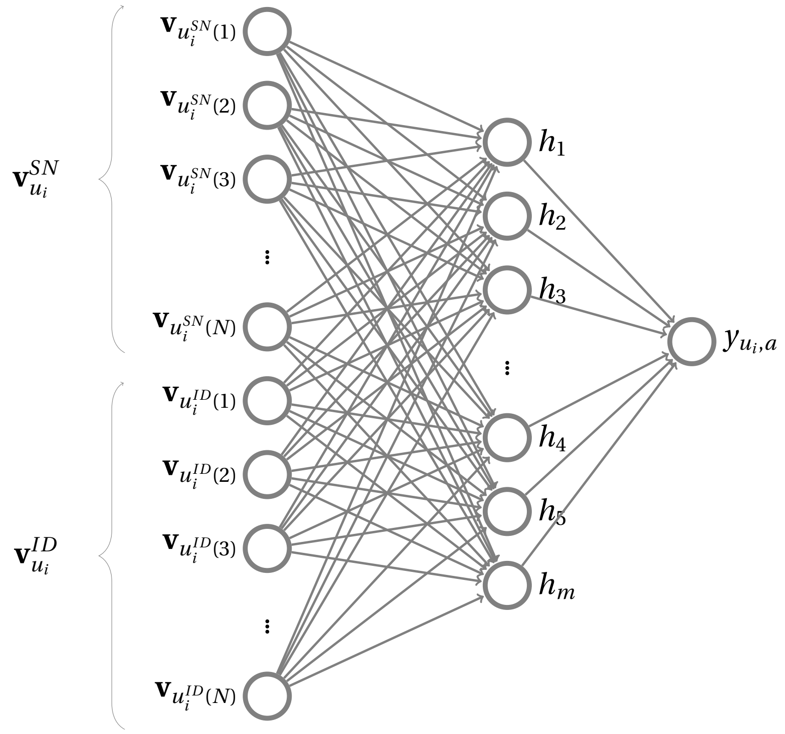

$\mathbf{v}_{u_i}^{SN}$
$\mathbf{v}_{u_i^{SN}(1)}$
$\mathbf{v}_{u_i^{SN}(2)}$
$\mathbf{v}_{u_i^{SN}(3)}$
$\mathbf{v}_{u_i^{SN}(N)}$
$\mathbf{v}_{u_i}^{ID}$
$\mathbf{v}_{u_i^{ID}(1)}$
$\mathbf{v}_{u_i^{ID}(2)}$
$\mathbf{v}_{u_i^{ID}(3)}$
$\mathbf{v}_{u_i^{ID}(N)}$
h_1
h_2
h_3
h_4
h_5
h_m
$y_{u_i,a}$

Figure 4.1: Global-SIDL (G-SIDL)

One-hot encoding is widely used in machine learning to distinguish the elements of a set. The target-user ID vector consists of all *zeros* with the exception of a single *one* that identifies the target-user, i.e., u_i is represented by the vector $\mathbf{v}_{u_i}^{ID}$, whose i^{th} element is the only element equal to one. The latter ($\mathbf{v}_{u_i}^{SN}$) gives a representation both of the social network connections and of the users' activity. In particular, each element represents a user in the social network and its value indicates the state (active/inactive) of the user for a given action a. Thereby, the social network vector $\mathbf{v}_{u_i}^{SN}$ of the target-user u_i represents the social network of u_i and the state of her/his active friends. The j-th element of $\mathbf{v}_{u_i}^{SN}$ corresponds to user u_j and the corresponding input value is computed as follows:

$$\mathbf{v}_{u_i}^{SN}(j) = \begin{cases} 1 & \text{if } (u_j, u_i) \in E \text{ and } u_j \text{ is active} \\ 0 & \text{otherwise} \end{cases}$$

These two vectors are first concatenated and then fed into a multi-layer architecture, as depicted in Fig. 4.1, where, for the sake of simplicity, a DNN with only one hidden layer is depicted. In our experiments, we design a network with a tower structure, where the bottom layer is the largest and the number of nodes of each successive layer is half of its precedent. In such a way, according to [124, 125], higher layers with few nodes can learn more abstractive features from the input data. Details about the implementation will be given in Section 4.2.6. The output of the DNN $y_{u_i,a}$ corresponds to the target-user u_i and is equal to 1 if she/he performed a, and is 0 otherwise. The predictive model of the G-SIDL approach can be formulated as $\hat{y}_{u_i,a} = f(\mathbf{v}^{ID}_{u_i}, \mathbf{v}^{SN}_{u_i}|\Theta)$, and the training is performed by minimizing the cost function $\mathscr{L} = -y_{u_i,a}\log(\hat{y}_{u_i,a}) - (1 - y_{u_i,a})\log(1 - \hat{y}_{u_i,a})$, which is referred to as binary cross-entropy loss.

Interpretability and Scalability Although the G-SIDL architecture provides a complete picture of the social network by embedding every user in a unique model, it presents two issues: interpretability and scalability. Note that, as we clarified in Section 4.2.1, the notion of interpretability used throughout this Section is related to a post-hoc explanation of the results. More specifically, we aim at understanding which subset of the input data is mainly responsible for the prediction outcome [116]. As each input of G-SIDL represents a social network user, the global model does not provide a comprehensible explanation of the results. In fact, G-SIDL maps the interplay between all the users and does not allow to identify the subset of input responsible for the prediction related to a given target-user, i.e., we cannot identify the subset of individuals that mainly influence the target-user.

Towards meeting this challenge, we exploit the notion of local explanations and, more specifically, the principle behind the Saliency Mask (SM) technique. SM is defined as a summarized explanation of where the classifier “looks” to make its prediction [66]. A suitable example is provided by the image classification scenario, where SM is used to find the part of an image most responsible for the classifier decision [91]. The idea behind SM is to understand what a neural network depends on *locally*, i.e., to identify a subset of the input used by the model to produce the output.

In this thesis, we reinterpret the usage of such a technique in the context of social networks. According to the concept of local explanation, here we narrow the input space by deleting regions of the social network that may not be *relevant* to infer the activity of a given target-user. We follow the Smallest Sufficient Region (SSR) approach [66, 91], which aims to identify the smallest set of the input that achieves a classification accuracy in line with the general (complete) model. We provide two

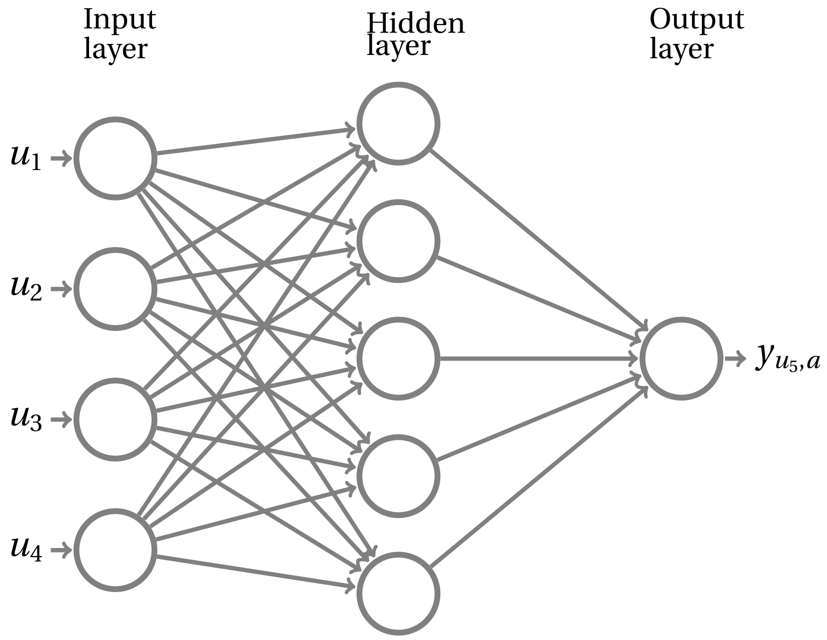


Figure 4.2: Local SIDL (L-SIDL) of user u_5

different solutions by varying the network connectivity of each user and adapting the DNN architecture accordingly. The two solutions, basically, represent two different sizes of the SSR.

On the other hand, scalability issues may occur when new users register to the social network. In such a case, G-SIDL requires to be retrained to include the new users in the model. This process is computationally expensive, both in time and resources. Our objective here is to mitigate this issue in the most efficient way while preserving the performance of the G-SIDL approach. We next present the solutions proposed to overcome both these issues.

4.2.3.3 Local-SIDL (L-SIDL)

Inspired by the SSR approach [66, 91], we consider the smallest social network within each individual is embedded into. Thereby, we take into account only the set of one-hop neighbors that each user is connected to. The resulting *ego network* (defined in Section 2.7) is used as the input of each user-based model. We refer to this solution as *Local-SIDL* (L-SIDL), as we create a *local* model for each target-user by employing a DNN for each one of them.

Such a solution can represent a more interpretable and scalable solution if compared to G-SIDL. While in the global approach the post-hoc explanation of the results was blurred by the huge number of inputs, we here restrict our analysis to the nodes directly connected to the target-user. In such a way, we can better understand whether and to what extent this subset of nodes influence the target-user. In terms of scalability, the user-based model offers a more agile solution in terms of time and resource consumption. In fact, a new user in the OSN requires only (i) to create a new instance of L-SIDL to model the new user, and (ii) to retrain only the L-SIDLs related to the new user's friends to update their model by including the new user, which both represent operations computationally less expensive (in terms of computational time and resource) than the retraining of the whole G-SIDL. This assessment will be detailed and discussed in Section 4.2.7.

As an example, Fig. 4.2 shows the L-SIDL related to the target-user u_5, whose ego network is shown in Fig. 2.1a. Also in Fig. 4.2, for the sake of simplicity, we depict a DNN with only one hidden layer $(L = 1)$, while in our implementation (Section 4.2.6) we employ multiple hidden layers. Differently from Fig. 4.1, and more in general from the G-SIDL approach, the number of input nodes of L-SIDL depends only on the number of target-user's friends. In fact, each input u_i of the DNN displayed in Fig. 4.2 represents a friend of target-user u_5. For each action a, input nodes can assume value 1 if the corresponding friend performed action a before u_5, and 0 otherwise. The output of the DNN $y_{u_5,a}$ corresponds to target-user u_5, and is equal to 1 if she/he performed a, and is 0 otherwise. The training is performed by minimizing the binary cross-entropy loss between $\hat{y}_{u_5,a}$ and $y_{u_5,a}$, where $\hat{y}_{u_5,a} = f((u_1, u_2, u_3, u_4))|\Theta)$ is the predicted output of the L-SIDL framework. Finally, we utilized the trained model to predict whether the target-user will perform new activities based only on her/his active friends S_{a,u_5} for those activities.

4.2.3.4 Community-SIDL (C-SIDL)

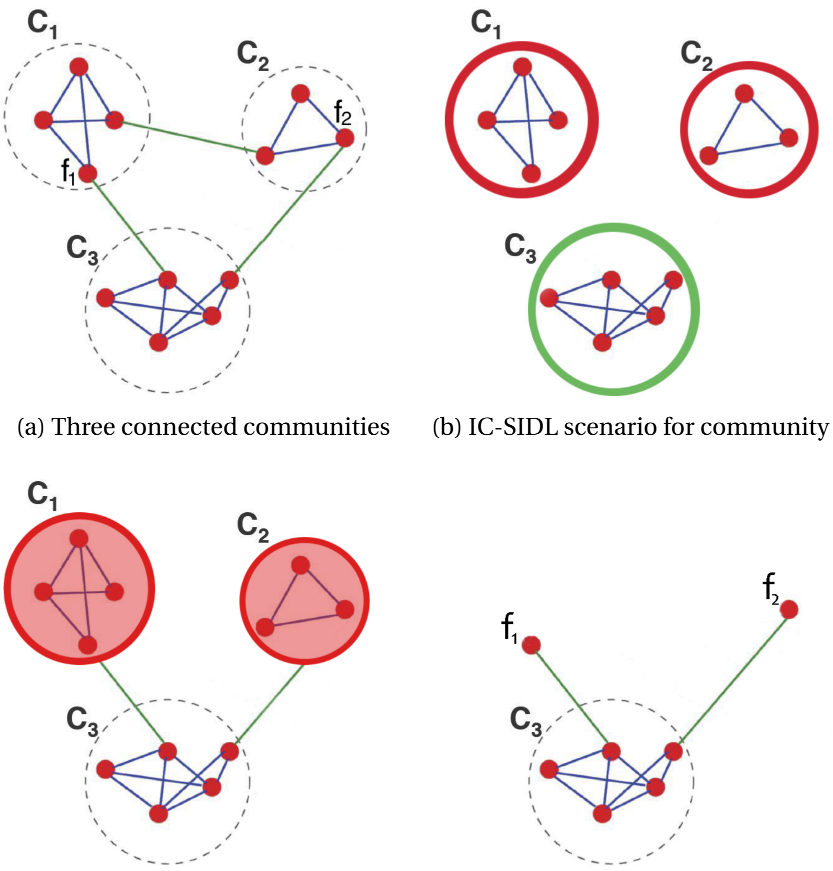
While L-SIDL can offer a more interpretable and scalable solution, the prediction performance may be affected by the reduced amount of information that each DNN utilizes. In fact, by splitting the social network into different and *isolated* ego networks, we break the G-SIDL into N L-SIDL models, and in turn, we do not exploit and model the interconnections between the ego-networks. Also, the L-SIDL approach does not take into account nodes distant more than one hop (e.g., friends of a friend) from the target-user. Thus, it assumes that the possible *influencer* nodes are only in the ego network of the target-user.

In this subsection, we propose *Community-SIDL* (C-SIDL), an approach that aims to solve the trade-off between performance, interpretability, and scalability by embedding users in mesoscale level structures, such as communities, and employing a DNN architecture for each community. Such an approach, which can be viewed as a combination of deep learning with network science, offers a solution in-between G-SIDL and L-SIDL. We move from the totality of the nodes (in G-SIDL) to a smaller subset (but larger than a ego-network) by splitting the social network into $1 \leq M \leq N$ communities and by employing a DNN for each of them. We refer to this solution as C-SIDL since each community is mapped into a distinct DNN. The rationale of this approach is to preserve the social network structure at a lower resolution, as a community can be considered as a partition of a graph [93], while enhancing the interpretability and scalability issues. In fact, by deploying M DNNs, instead of one (G-SIDL), we contain the inefficiency of the G-SIDL in the case of new users. In such a scenario, a new user in the OSN requires only to retrain the C-SIDL corresponding to the community she/he belongs to. We aim to partition the network by maximizing the modularity, i.e., the density of intra-community edges with respect to inter-community edges. As this is a NP-hard problem, we make use of a heuristic algorithm. We employ the Louvain method [37] for its computational efficiency.

The C-SIDL approach opens the door to different solutions by varying the inter-community connectivity (i.e., the way communities are connected among each other). For instance, each community can be considered as an independent component of the graph or as a unit connected with other (linked) communities. Figure 4.3a shows an example of three connected communities, referred to as C_1, C_2, and C_3. In Figure 4.3, we show the varying inter-community connectivity of C_3 according to the following approaches:

1. *Isolated Communities-SIDL (IC-SIDL)*: each community is analyzed separately. As shown in Fig. 4.3b, IC-SIDL does not consider inter-community edges of community C_3. The resulting IC-SIDL architecture is similar to G-SIDL, but the inputs of the DNN are the only users belonging to C_3.

2. *Collapsed Communities-SIDL (CC-SIDL)*: every connected community is considered as a super node linked to the community C_3, as it is depicted in Fig. 4.3c. In this solution, we collapse an entire community into a single input node



(a) Three connected communities (b) IC-SIDL scenario for community C_3

(c) CC-SIDL scenario for community C_3 (d) IFC-SIDL scenario for community C_3

Figure 4.3: Example of communities and inter-community connectivity

(*community node*) of the DNN. The resulting CC-SIDL follows the IC-SIDL implementation but includes also the community nodes as input. As an example, community node C_1 (same for C_2) acts as an input of the CC-SIDL of C_3.

3. *Inter-Friendship Communities-SIDL (IFC-SIDL)*: only users directly connected to C_3 are considered as input of the IFC-SIDL, other than the members within C_3 (as in IC-SIDL). Fig. 4.3d shows an example of two users (f_1, f_2) connected to C_3.

4.2.4 OSN Data

For validating and evaluating the SIDL framework, we consider different datasets from two OSNs. In particular, we focus on scenarios of real-life activities, such as attending an event or visiting a location. The idea is that a subject might participate in an event because she/he sees her/his friends taking that decision, or she/he may visit a location (e.g., a bar or a restaurant) because some friends have been there before and suggested her/him that venue. For this reason, we analyze data from Plancast[1], an Event-Based Social Network (EBSN) [160], and Foursquare[2], a Location-Based Social Network (LBSN) [27]. EBSN and LBSN provide remarkable opportunities to analyze users' real-world behavior through OSNs. In these scenarios, the set of actions A is defined by the event participation and location visits in the EBSN and LBSN, respectively. Thereby, $A_{u_i} \subseteq A$ represents the set of events (resp. locations) attended (resp. visited) by subject $u_i \in V$, while a subject is considered active for the action a if she/he decided to participate in (resp. visit) the event (resp. location) $a \in A$.

In this study, we analyze a dataset from Foursquare collected by Bao et al. [27], which gathers Foursquare check-ins from the cities of New York and Los Angeles. The first dataset (New York) was collected for 30 months from May 5, 2008, while the second dataset (Los Angeles) gathered check-ins for 36 months from February 5, 2009. In Foursquare, users can check-in their location through a mobile application, give recommendations (*tips*), connect with their friends, and share with them their experiences. In such a scenario, users have the chance to discover new venues, look for trend places, and read friends' reviews. This bundle of information can produce a social contagion effect, which may affect users' activity.

We also rely on the EBSN Plancast and, in particular, we use a dataset collected by Liu et al. [160] for three months (from September to November 2011). Plancast allows users to subscribe to each other providing direct connections among them. Subscription is similar to the concept of *following* on Twitter. Users can directly follow friends' event calendars: this mechanism allows a subject to be aware of friends' interests, event creation, and participation.

Table 4.1 and Figure 4.4 summarize properties and statistics of both the OSNs analyzed in this study. It should be noticed that we do not model users across the two OSNs as their IDs have been anonymized before the data release. Thus, it is not possible to match users from different datasets.

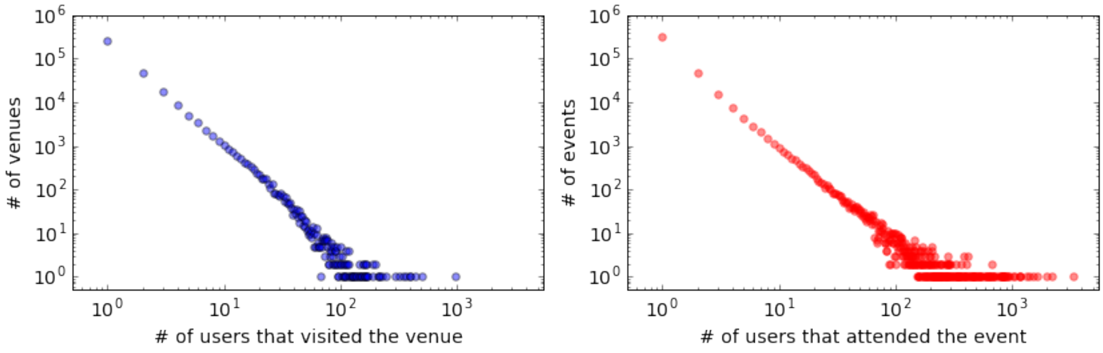
In Figure 4.5, we display the average number of friends that visited (resp. attended) a given venue (resp. event). As we expected, events involve more friends, and in turn represent a more direct and expansive form of social interaction, if compared to the

[1] www.plancast.com

[2] www.foursquare.com

Table 4.1: Basic statistics of the OSNs used to test SIDL approaches

	Foursquare		Plancast
	NYC	LA	
Users $\|V\|$	47240	30207	75598
Friendship Relationships $\|E\|$	596379	246560	1501618
Average degree $\langle k \rangle$	25.25	16.32	39.7
Diameter d_{max}	10	12	11
Average clustering coefficient $\langle C \rangle$	0.14	0.15	0.23
Density D	0.0005	0.0005	0.0005
Total number of actions $\|\mathscr{A}\|$	425692	268102	869200
Total number of venues v	206098	144348	401634



(a) Foursquare

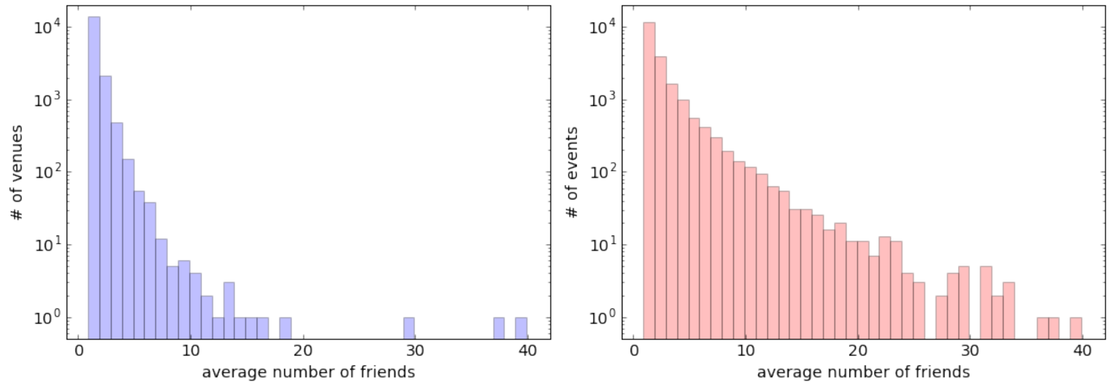
(b) Plancast

Figure 4.4: Distribution of the number of users that visited (resp. attended) a venue (resp. event)

visit of a certain location, which may represent mainly individual behaviors [160]. However, the number of friends that visited a given venue is not negligible and, thus, we consider this scenario in our study.

4.2.5 Experimental Settings

In this subsection, we describe the data processing and we discuss the training, validation, and testing phases of SIDL. Each dataset includes information about users' activity over time and social connections among subjects in the OSN, while no additional information or metadata are provided. These data allow us to build (as explained in Section 4.2.2) the action log $\mathscr{A}_l$ and the graph G, which represent the only two inputs required by our framework. In fact, the former is used to keep track of users' activity over time and, in combination with the latter, to understand how the



(a) Foursquare

(b) Plancast

Figure 4.5: Average number of friends that visited a given venue (a), and that attended a certain event (b).

actions propagated between the users. To reduce noise in the dataset and to build, for each user, a reasonable training and test set, we remove users with less than 10 actions. For Plancast, we restrict our analysis to the US as the majority of the users attended events organized in this country.

For each subject u_i, we consider the set of actions the user performed ($A_{u_i} \subseteq A$) and we randomly select n_{u_i} actions not performed in order to consider negative samples, where $n_{u_i} = |A_{u_i}|$. It should be noted that, for each user, we consider only the actions that have been performed by at least one of her/his friends, according to [106].

To limit overfitting and to reduce variability, we utilize a 10-fold cross validation. We built the folds to preserve the percentage of positive and negative samples for each subject in the dataset. Each sample (both for training and test) represents an action performed (or not) by a given target-user. For each action in the training set, we provide information about target-user's friends activity along with the ground truth related to the target-user activity. On the other hand, for the actions in the test set, we employ only the information related to target-user's friends activity with the purpose of inferring whether the target-user performed the action.

4.2.6 SIDL Implementation

This subsection details the implementation of every SIDL approach previously described. In each SIDL approach, every input node of the DNN represents a user in the OSN. The input node is a binary variable that indicates the activity of a user for a given action. Inputs are fed into the DNN architecture according to the different SIDL approach, as explained in Section 4.2.3.

While the input preparation for G-SIDL and L-SIDL is similar and has been already detailed (in Sections 4.2.3.2 and 4.2.3.3, respectively), some additional clarifications are needed for the C-SIDL approaches. In C-SIDL every community is mapped into a different DNN. The different C-SIDL approaches differ from each other in the way the inter-community links are considered. As an example, Figure 4.3 shows the three different scenarios for community C_3. IC-SIDL, which does not take into account the inter-community links, follows exactly the G-SIDL implementation but its inputs are the only nodes belonging to the community under exam (C_3 in the example in Fig. 4.3b). On the other side, CC-SIDL and IFC-SIDL consider the inter-community links in two different ways.

In CC-SIDL, each connected community to C_3 is collapsed into a single input node, named *community node*. The resulting DNN complies with the IC-SIDL implementation for the nodes belonging to C_3, but also includes the *community nodes* C_1 and C_2. Each community node is employed as an additional member of C_3 and is considered as a friend of the users directly connected (the top right users in the example in Figure 4.3c) to the community node itself. To represent the community node's activity for a given action we tested different strategies. We set the community node to (i) a binary value, which assumes value 1 if at least one user within the collapsed community performed the action, (ii) a binary value, which assumes value 1 if at least 50% of the users within the collapsed community performed the action, or to (iii) a real value representing the fraction of the users within the collapsed community that performed the action. Among the three strategies, we used the first approach as it achieved slightly better results (details on the performance in the next Section). Finally, IFC-SIDL considers the inter-community friends as additional members of the community under examination. In the example in Fig. 4.3d, users f_1 and f_2 are included in the DNN related to community C_3 and considered as friends of the connected nodes.

We implemented the DNNs by following a tower pattern composed of $L = 3$ layers with {128,64,32} nodes, respectively. For the Foursquare dataset, we used a fourth layer of 256 neurons (i.e., $L = 4$ layers with {256,128,64,32} nodes) as it improved the

Table 4.2: Grid search for hyperparameters optimization

Accuracy	Batch Size	Initialization	Epochs	Optimizer	ϕ_j	ϕ_o
88.7	20	Glorot	25	RMSProp	sigmoid	sigmoid
88.6	10	Glorot	10	RMSProp	sigmoid	sigmoid
88.3	20	normal	10	RMSProp	ReLU	sigmoid
87.8	10	Glorot	10	RMSProp	ReLU	sigmoid
87.7	10	Glorot	10	Adam	sigmoid	sigmoid
85.7	10	Glorot	10	Adam	ReLU	sigmoid

performance with respect to the three layers architecture used for the Plancast dataset. We tuned hyperparameters performing a grid search on a validation set (10% of the data). In particular, we examined the following hyperparameters:

- *Batch size* defines the number of samples used to update the model at each iteration;
- *Initialization* of the inter-layer weights;
- *Epochs* represent the number of times each sample in the dataset is considered during the training;
- *Optimizer* indicates the optimization algorithm used to update the network weights;
- *Activation function* at the hidden layers ϕ_j and at the output layer ϕ_o are the non-linear function used to *activate* the neurons over all the network.

The hyperparameter optimization consists of an exhaustive searching through the following hyperparameters space: batche size = {10,20}, initialization = {normal, uniform, Glorot}, Epochs = {10,25}, optimizer = {RMSProp, Adam}, activation functions ϕ_j = {sigmoid, ReLU}, and ϕ_o = {sigmoid, ReLU}. Table 4.2 depicts the six best results, in terms of accuracy, for the combinations of the above hyperparameters. Thereby, we employ a sigmoid as activation function (both at the hidden layers and at the output layer), and we use RMSProp [68] as optimization function. We train the network in data batches composed of 20 samples for 25 epochs. We further evaluated the impact of the number of epochs in the performance by testing the model for 50 and 100 epochs. Despite the gain of about 0.1% (in the case of 100 epochs), we decided to use 25 epochs for a time efficiency reason, as a larger number of epochs implies a longer training time. Finally, we apply a dropout technique [217], with a dropout equal to 0.1, to avoid overfitting.

4.2.7 Evaluation

In this subsection, we evaluate the results of the SIDL approaches and we compare them with two state of the art approaches: The Linear Threshold (LT) model proposed by Goyal et al. [106], and the Independent Cascade (IC) model of Saito et al. [208]. We use as baseline these two solutions as they offer general models to learn social influence between users by leveraging only the history of the actions performed by each subject. In fact, (i) they do not rely either on specific hand-crafted features or on topic affinity, which in turn may depend on the OSN analyzed and on the availability of metadata (e.g., personal attributes), and (ii) they both take as input only the action log $\mathscr{A}_l$ and the social graph G. In a similar way, we focus on a model that can be generalized to any kind of OSN and, less specifically, to any real-life domain. Both these approaches rely on two commonly used models in information diffusion, namely the LT and IC model, which we introduced in Section 2.3. In particular, they first aim at learning influence probability between users, and then they combine the social influence model with the diffusion model to predict users' activity. In [106], Goyal et al. introduced different metrics to estimate the pairwise influence between two individuals and proposed a static and dynamic (time-dependent) version of the LT model. We evaluated all the metrics and variations of the LT model proposed in [106] and we report the results related to the Discrete Time (DT)-Bernoulli approach as it achieved better performance if compared to the other metrics. We refer to this solution as LT-DT indicating the discrete time version of the LT model. On the other hand, Saito et al. [208] employed the IC model along with the Expectation-Maximization (EM) algorithm to estimate the influence probability associated with each edge. We developed this model, here referred to as IC-EM, by minutely following the 2-steps learning and the experimental setup suggested in their paper [208].

4.2.7.1 Performance Comparison

In Figure 4.6, we compare the performance of these solutions with our proposed approaches in terms of prediction *accuracy*. This metric stands for the number of correctly classified samples over all the samples classified. Results indicate that the

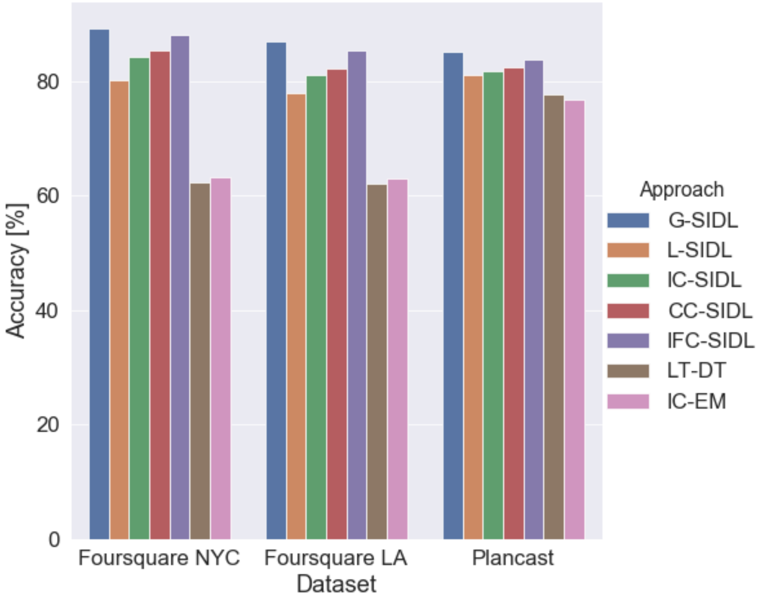


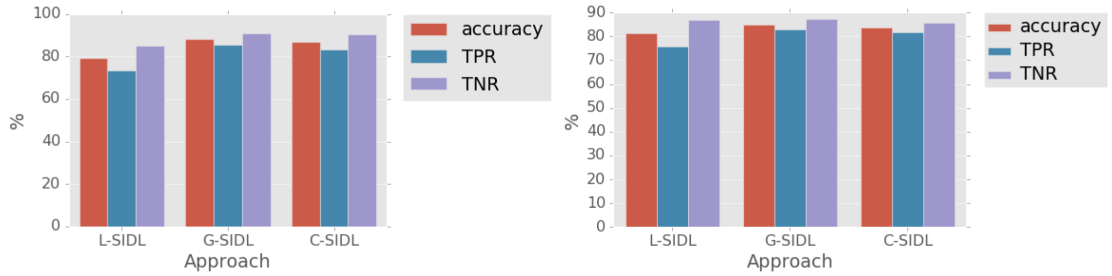
Figure 4.6: Models performance in terms of accuracy: G-SIDL vs. L-SIDL vs. C-SIDL (IC-, CC-, and IFC-SIDL) vs. baseline models (LT-DT [106] and IC-EM [208]).

proposed SIDL approaches, G-SIDL, L-SIDL, and C-SIDL, achieve an average accuracy of 86.6%, 80.1%, and 85.2%, respectively, and outperform baseline algorithms (LT-DT and IC-EM), which both achieve an average accuracy of 70.0%. Compared to the baseline classification accuracy, G-SIDL, L-SIDL, and C-SIDL reach an average gain of 23.7%, 14.4%, and 21.7%, respectively.

As we expected, G-SIDL outperforms the local approach, while C-SIDL, with its three variations (IC-, CC-, and IFC-SIDL), offers a valuable alternative to the global solution. Three aspects are worth noting:

- IC-SIDL performs better than L-SIDL as it considers the community within the user is socially embedded, and not only the direct social connections. However, this solution breaks the connectivity between linked communities, thus, performs worse if compared to CC-SIDL and IFC-SIDL.

- CC-SIDL slightly overcomes the IC-SIDL accuracy but has a small gain with respect to L-SIDL. Collapsing an entire community in a unique node oversimplifies the inter-communities social relationships, but provides additional information if compared to the IC-SIDL solution.



(a) Foursquare

(b) Plancast

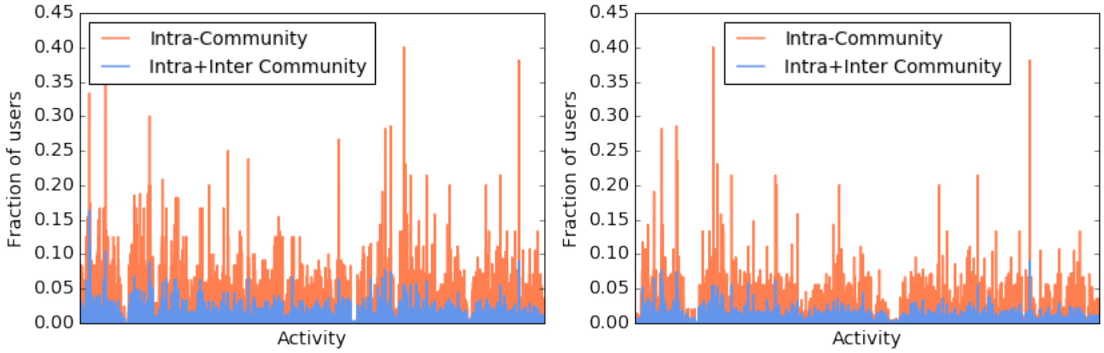
Figure 4.7: Accuracy, TPR, and TNR of SIDL approaches

- IFC-SIDL achieves the best performance among the C-SIDL approaches and its accuracy closely approaches G-SIDL, highlighting the importance of inter-community edges in modeling social influence.

To better investigate the performance of our approaches, we examine other binary classification metrics, such as True Positive Rate (TPR) and True Negative Rate (TNR). TPR measures the percentage of positive samples that are correctly identified as such, while TNR is the analog for negative samples, i.e., it measures the percentage of correctly classified negative instances. In our scenario, these two metrics represent the ability of the classifier to identify performed actions and not performed actions, respectively. In Figure 4.7, we compare our approaches using accuracy, TPR, and TNR. Please note that for the C-SIDL approach we consider IFC-SIDL as it outperforms the other two C-SIDL solutions (IC-SIDL and CC-SIDL). Overall, we observe that TNR is higher than TPR in every approach, meaning that our system (slightly) better classifies negative samples. The difference between TPR and TNR is more pronounced in the L-SIDL approach, while in C-SIDL and, especially, in G-SIDL the two metrics are more balanced. Interestingly, the difference between TPR and TNR is less noticeable in Plancast than in Foursquare, probably because the number of friends per visited location is significantly lower if compared to Plancast events, as we previously showed in Figure 4.5, and the model, in turn, is less accurate to classify this kind of activity.

4.2.7.2 Community Analysis

Further, we investigate whether the number of members per community impacts the prediction performance. For this purpose, we compute the Pearson correlation coefficient [31] between the accuracy and the number of members per community.



(a) Foursquare

(b) Plancast

Figure 4.8: Intra- and Inter-community activity

The result does not show a statistically significant correlation (ρ=0.3, p-value=0.17 $>$ α=0.05). We then explore whether and to what extent connected communities share their activities. To accomplish this purpose, for each community, we compare the actions performed by its members, namely *intra-community actions*, with those performed by the members of the inter-connected communities, referred to as *inter-community actions*. More specifically, for each community, we compute the fraction of users that performed a given activity considering (i) only the members of the community, and (ii) considering both the members of the community and of the inter-connected communities. Figure 4.8 depicts the results for both datasets. As expected, intra-community actions present higher fractions of users if compared to the combined (intra- with inter-community) scenario. However, the contribution provided by the inter-connected communities is not negligible and, according to the prediction performance, plays a significant role. Interestingly, in Plancast both the percentage of intra- and inter-community actions is higher with respect to Foursquare, further highlighting the differences we revealed before. In the Plancast dataset, we can also note that for some activities every member of the community performed the action. After further inspection, we recognize that these activities correspond to small clique events, which involved communities composed of a few members.

4.2.7.3 Scalability

In Table 4.3, we summarize the performance of the three SIDL approaches and compare them in terms of scalability (defined in Section 4.2.1). We measure scalability by considering the computational time required to train a single DNN in every SIDL

Table 4.3: Comparison among the three presented approaches in terms of performance and scalability

	L-SIDL	C-SIDL	G-SIDL
Accuracy	80.1%	85.2%	86.6%
TPR	74.4%	82.4%	84.3%
TNR	85.9%	88.1%	88.7%
Computational Time	t	7.8t	75.1t
# of DNNs	N	L	1

approach. We sort the table in increasing order of computational time or decreasing order of number of DNNs employed. The value t, in Table 4.3, is about 42 seconds, which has been computed by averaging the computational time required to train a single L-SIDL over the datasets. Although the G-SIDL approach achieves the best classification performance, it requires a (unique) DNN with a huge number of inputs (two times the number of nodes in the social network), which is not scalable for a huge graph. In fact, retraining G-SIDL every time a new user registers to the OSN is time and resource consuming. On the other side, L-SIDL offers more flexibility and efficiency as every user is modeled with a different DNN. Therefore, this approach may easily handle new users in the OSN by creating a DNN for the new user and updating only the DNNs of the new user's friends. The input size of each L-SIDL equals the number of friends of the target-user. Thus, the corresponding DNN is significantly smaller than the global neural network (G-SIDL). For this reason, the training phase of a single L-SIDL is, on average, 75 times faster than the G-SIDL. However, the computational efficiency of the L-SIDL approach is paid in terms of performance. The local solution breaks the whole social network structure in disconnected ego networks and, thus, performs poorly (average accuracy of 80.1%) if compared to G-SIDL (average accuracy of 86.6%).

The trade-off is solved by C-SIDL, which achieves performance close to the G-SIDL with limited issues in scalability: a new user in the OSN requires only to retrain one C-SIDL, whose computational time is about 10 times faster than G-SIDL and 8 times slower compared to L-SIDL. Note that the number M of communities, and in turn the number of DNNs in C-SIDL, depends on the connections between users in the social network and on the detection algorithm utilized to extract the communities. As a consequence, the input size of a C-SIDL depends on the number of members per each community. It should also be noticed that computational times are averaged over the different datasets, and that we run our experiment on a machine with a NVIDIA Tesla K20 (2496 CUDA cores - 5GB DDR5 RAM), a CPU Intel Xeon E5 2670 with a frequency of 2.3 GHz, and a 128 GB DDR3 RAM.

4.2.7.4 Interpretability

In Sections 2.6.1 and 4.2.1, we delineated the interpretability issue of DNNs and of the G-SIDL approach, respectively. In this subsection, we summarize our effort to provide a more interpretable model with respect to G-SIDL, while preserving its classification performance. In particular, we analyze whether C-SIDL and L-SIDL give us a better understanding in the *post-hoc explanation* of the results (which is the notion of interpretability used in this study) with respect to G-SIDL. Though the one-hop neighborhood (ego network) appeared as the straightforward solution to detect the set of input (nodes) that affected the prediction related to a given user, the performance related to the L-SIDL approach shows that the ego network alone is not enough to explain the social influence phenomenon. As we discuss in Section 4.3, mesolevel structures (defined in Section 2.7), such as communities, provide a significant contribution to the understanding of this phenomenon. Results show that C-SIDL closely approaches the performance of the general model. Therefore, the SSR identified by a community better explains the results achieved by G-SIDL if compared to the SSR built with the ego network only (L-SIDL). In these terms, C-SIDL provides better interpretability of the global model if compared to L-SIDL. Overall, we can further explain the gap between L-SIDL and C-SIDL in terms of influential nodes considered within each social structure. The rationale of L-SIDL is that the most influential nodes stand in the ego network of each user, while the idea behind C-SIDL is to consider a larger social structure (i.e., a community), which in turn includes a larger number of users (not necessarily connected to the target-user) and, therefore, (additional) potential influencers.

4.2.7.5 SIDL Augmentation

In this subsection, we discuss potential extensions of our proposed approach along with some preliminary results. In particular, we explore the usage of different deep learning architectures across our framework. In the current version, SIDL uses feed-forward neural networks, a class of DNNs in which the flow of information moves

Table 4.4: Architecture performance in terms of accuracy: Feed-forward vs. LSTM vs. GRU.

	Foursquare		Plancast
	NYC	LA	
Feed-forward	89.3%	87.0%	85.1%
LSTM	91.1%	89.2%	85.6%
GRU	86.7%	85.3%	83.3%

forward from the input to the output neurons through the hidden layers. As introduced in Section 2.6.1.1, feedback connections have been extensively used in RNN for modeling sequential data, such as human activities.

For this reason, we propose to use a RNN architecture in the SIDL approach. We expect that such a solution might be more beneficial in modeling users actions and dependencies over time with respect to a feed-forward architecture. In this subsection, we show some preliminary results obtained employing SIDL with two of the most commonly used RNNs: Long Short-Term Memory (LSTM) [126] and Gated Recurrent Unit (GRU) [64]. More specifically, in this test, we consider G-SIDL as it achieves the best prediction performance, but the same architecture can be extended to the other SIDL approaches. In Table 4.4, we compare the results of the three DNN architectures on the two datasets. We observe that LSTM outperforms the GRU solution and achieves a better accuracy of the feed-forward architecture. Interestingly, the gap is particularly noticeable in the Foursquare dataset. This may be due to the different nature of the activity in the two datasets. While an event is a one-shot activity held on a specific date, the visit of a certain location may occur in distinct days from one user to another. This hypothesis, with related analysis and a broader exploration of DNN architectures, will be expanded upon in future work.

4.2.8 Discussion

In this Section, we have shown how social influence modeling can be effectively used to infer users' future activities and, in turn, violate their privacy. In particular, to address RQ 2.1, we have introduced SIDL, a framework that combines deep learning with network science. SIDL approaches have proven to outperform existing social influence models by overcoming their limitations. Compared to the performance of such models, the SIDL approaches, G-SIDL, L-SIDL, and C-SIDL, reach an average gain of 23.7%, 14.4%, and 21.7%, respectively. Moreover, SIDL allowed us to face two typical challenges of deep neural networks: interpretability and scalability. We have shown that the approach based on mesoscale structures (C-SIDL) provides a more interpretable solution (in terms of post-hoc explanation of the results), while maintaining a good trade-off between scalability (in terms of computational time) and prediction accuracy.

4.3 Community Influence

In Section 4.2, we present SIDL, a framework for modeling social influence, which leverages dyadic social interactions between users to predict their behavior. In Section 4.3, to answer RQ 2.2, we still aim to learn social influence at the user-level, but we explore the collective influence that a group of people might have on an individual. Also, we consider different factors that can impact subjects' behavior, other than social relationships. Many works in the literature attempted to figure out the factors that affect people decisions and actions. We can classify them in three interlaced categories: physical location, homophily, and social ties.

In Section 4.3, we propose to examine the role that distinct communities, linked to these factors, play as sources of social influence. Although the ego network is typically used in the social influence modeling, our hypothesis is that individuals are embedded in communities not only related to their direct social relationships, but that involve different and complex forces. To this end, we analyze physical, homophily, and social communities to evaluate their relation with subjects' behavior.

To examine the role of these communities in the social influence modeling, we rely on the EBSN Plancast [160]. The motivation of this choice is related to the concept of event (defined in 2.8), which represents a collective and agglomerative circumstance and, thus, perfectly fits the purpose of this study. Differently, LBSNs describe mainly individual activities [160], thus, they are less relevant for studying group influence, as further confirmed in Section 4.2.7.2.

4.3.1 Problem Definition

In this subsection, we define the problem and the elements involved in this analysis, which aims to address RQ 2.2. We indicate with V the set of users and with A the set of events in our dataset. We denote A_u as the set of events attended by $u \in V$.

Further, we define the *centroid of interests* c_u, which represents the user's area of interests based on the locations of the attended events. Given that our dataset does not include any information related to the users' home location, we evaluate c_u to assign a representative geo-location to the user. This centroid is expressed as a pair of latitude and longitude coordinates. We compute c_u in two steps. First, we detect and remove the outliers from the list of coordinates in A_u, according to the Median Absolute Deviation (MAD) measure. Second, we calculate c_u by averaging the coordinates of the remaining events in A_u.

We then define three graphs related to three interlaced factors affecting human behavior, i.e., physical location, homophily, and social ties. The rationale is to characterize connections among OSN users by capturing their common (i) visited places, (ii) interests, and (iii) social relationships. Based on this idea, let SG be *Social Graph*, $SG = (V, E_s)$, where E_s is the set of edges connecting users in the social network. The generic pair of users (u, v) is referred to as friends if $(u, v) \in E_s$, and we denote with F_u the set of u's friends. We denote $PG = (V, E_p)$ as the complete *Physical Graph*, where each pair of users (i, j) is connected by an edge weighted by a function of the geographical distance between the users' centroid of interest. We utilize a Gaussian kernel as a function to transform the geo-distance into a measure of similarity: High similarity between a pair of users indicates a short distance between their centroids of interest. Finally, we define the complete *Homophily Graph* $HG = (V, E_h)$, whose edges E_h are weighted by an interest similarity measure defined as follows: $w_{u,v}^{HG} = (|A_u \cap A_v| / |A_u \cup A_v|)$. This metric indicates the number of common events attended by each pair of generic users (u, v), normalized by the joint set of attended events. It represents how similar two users are in terms of interests.

We partition these three graphs in order to create a set of communities within each graph: Social Communities (SC) for SG, Physical Communities (PC) for PG, and Homophily Communities (HC) for HG. The purpose is to group together users that have common social ties, which are in physical proximity, and which share interests, respectively. Similarly to Section 4.2.3.4, we extract communities so as to maximize the modularity and we utilize the Louvain method [37] for its computational efficiency.

Finally, for each user $u \in V$, we define the ego network ego_u, where the ego is u and the other nodes are the *alter*, i.e., u's friends. As every edge in the ego network represents a social tie between the ego and the alter, we extract ego_u from SG. Thus, ego_u is a user-centered subgraph of SG. Overall, we can see each user $u \in V$ as embedded in the ego network ego_u and in three different communities, each one representing a distinct characteristic of the individual in three dimensions: social (ties), space (physical location), and interests (homophily). We denote the three communities within each user u is embedded as SC_u, PC_u, HC_u. We point out that ego_u and SC_u capture two different social structures. The first includes only nodes directly connected to the user (e.g., friends), the second includes additional nodes (e.g., friends of a friend), according to the results of the Louvain partitioning method, and represents a larger social structure within the user is settled.

In this Section, we consider communities of different nature to evaluate their impact on human behavior and, in particular, on influence processes. More specifically, we aim to examine the role of physical, homophily, and social communities as sources of social influence and evaluate their relation with subjects' behavior. Differently from Section 4.2, here we explore the influence exerted by a group of people as a whole entity. Therefore, we analyze each community as a single source of social influence, instead of measuring influence probability among each pair of subjects. The rationale of this approach is based on the idea that subjects may follow the collective behavior of individuals (i) living in their physical area, (ii) with common interests, or (iii) in their social community.

4.3.2 Methodology

In this subsection, we present the methodology we employed to analyze the social influence phenomenon with respect to the communities presented above. From now on, we will use the term *group* to indicate both the three communities and the ego network. To analyze the influence exerted by each group on a given user, we rely on the activity performed by the user and the group members as well. In particular, for each event e attended by u, we compute the *group participation* feature $p_e^g(u)$ as follows:

$$p_e^g(u) = \frac{|\{i \in g | e \in A_i\}|}{|g|}\,, \tag{4.1}$$

where $g \in \{ego_u, SC_u, PC_u, HC_u\}$ and $i \in V$. This feature indicates the number of users in group g that attended event e, normalized by the dimension of the group. In such a way, we build, for each user, a dataset D_u, where each row represents an event attended by u and is a 4-tuple composed of the four features $\{p_e^{ego_u}(u), p_e^{SC_u}(u), p_e^{PC_u}(u), p_e^{HC_u}(u)\}$. Each feature represents a measure of group participation in an event. A high value indicates that most users in the group took part in the event. The idea is that when a subject sees that most of the group members have confirmed their participation in an event she/he may be more willing to participate.

To test this intuition, we try to predict user participation in social events based on group participation features. It should be noticed that our primary goal is not to find the best algorithm to perform the prediction, but to show that the communities we identified above influence human behavior and can be used to predict subjects' decisions. We rely upon supervised machine learning algorithms to infer whether a user participated in an event according to group participation. We frame this task as a classification problem, where the target T is a Boolean variable indicating user participation in the event and the attributes are the group participation features in the dataset D_u. However, this dataset includes only the events attended by u. To make the dataset balanced and, in turn, to infer also negative instances (i.e., the user does not participate in the event), we also consider n events not attended by u, where $n = |A_u|$. We select the n closest events to the centroid of interests c_u. For each event, we compute the four features $p_e^g(u)$, as in Eq. (4.1). Finally, the balanced dataset B_u includes $2n$ events, equally distributed between positive and negative instances, where each row is composed of four features $\{p_e^{ego}(u), p_e^{SC}(u), p_e^{PC}(u), p_e^{HC}(u)\}$ and a target T. Notice that, from now on, we will indicate the *features* as $\{ego, SC, PC, HC\}$, to simplify the reading.

4.3.3 Evaluation

We perform the classification on a per-user basis, i.e., by considering each user separately. In order to limit overfitting and to reduce variability, we utilize a 10-fold cross-validation to split the dataset B_u into training and test set. We build the folds so as to preserve the percentage of attended events in the dataset. To perform the classification task, we employ three machine learning algorithms: Decision Tree (DT), Support Vector Machine (SVM), and DNN.

In Table 4.5, we report the classification performance related to these three approaches, in terms of accuracy, precision, and recall. It is noticeable how SVM and DNN outperform DT, especially in terms of accuracy and precision. We decide to continue our analysis by utilizing SVM due to its computational efficiency.

We now investigate the relevance of each group (both communities and the ego

Table 4.5: Classifier performance

Classifier	DT	SVM	DNN
Accuracy	77%	81%	81%
Precision	74%	84%	85%
Recall	72%	76%	75%

Table 4.6: Performance utilizing one fixed feature

Feature	ego	SC	PC	HC
Accuracy	77%	77%	77%	76%
Precision	81%	81%	82%	82%
Recall	72%	71%	69%	67%

Table 4.7: Prediction performances comparison: all features vs. one fixed feature vs. feature selection

	all features	one fixed feature	feature selection
Accuracy	81%	77%	80%
Precision	84%	82%	84%
Recall	76%	70%	76%

network) in subjects' behavior. First, we try to evaluate our method utilizing only one feature among $\{ego, SC, PC, HC\}$. The goal is to understand whether one of the group features can be used to infer users' activities, without performance loss. In Table 4.6, we show the classification performance for each group participation feature taken singularly. It can be noticed that none of the results approaches the previous performance (SVM in Table 4.5), where all the features were considered. The ego network alone is not sufficient to achieve such results. This proves that also SC, PC, and HC are relevant in this analysis and corroborates our hypothesis.

Second, per each user, we select only one specific feature according to a feature selection algorithm. We utilize the concept of mutual information to select the feature that contributes most to the output variable. Table 4.7 compares the performance of the three described approaches: all the group features (SVM results in Table 4.5), one fixed group feature per every user (average results of Table 4.6), and one group feature per user based on the feature selection algorithm. Interestingly, feature selection closely approaches the performance obtained utilizing all the features. We notice that the selected group features are equally distributed among the users, i.e., each feature is selected for almost 25% of the users: Certain users are mainly influenced by their physical community, some others by their social community, some by their homophily community, and others by their direct social relationships, i.e., their ego network. This evidence confirms the importance of all the four groups, and it further validates our hypothesis. In fact, each community is differently significant for the users and each of them has a reference influence group, which is more correlated with her/his actions. In the next subsection, we explore the relation between each of these groups and their relevance in terms of influence strength.

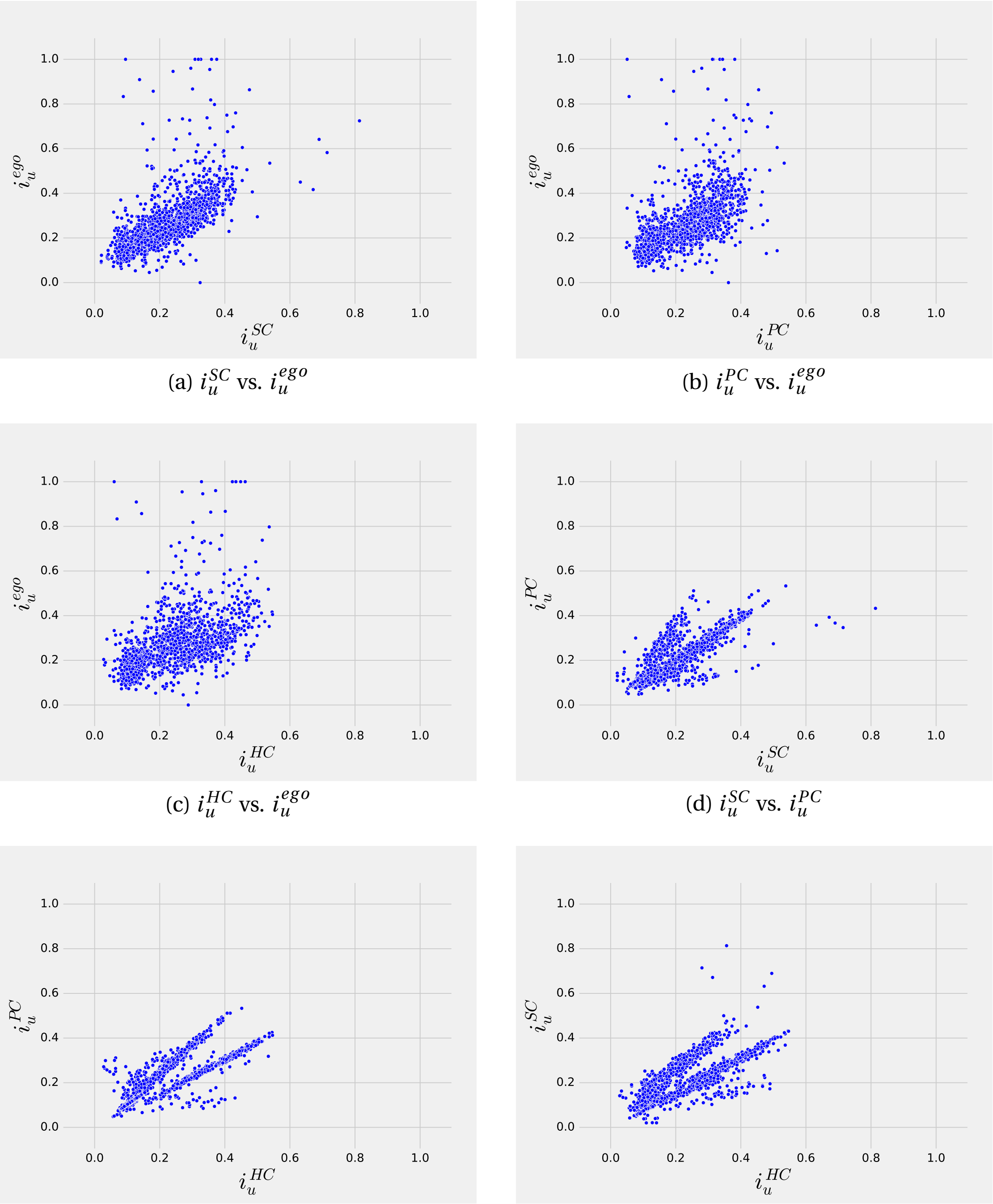
4.3.4 Behavioral Phenotypes

To investigate the role that groups have in influencing users' behavior, we characterize each user u with respect to the four group features $p_e^g(u)$. For this purpose, we utilize D_u (as described in Sect. 4.3.2), and we compute the average of the features $p_e^g(u)$ over all the events that u attended. We denote this value as *group influence* i_u^g, which is defined as:

$$i_u^g = \frac{1}{|A_u|} \sum_{e \in A_u} p_e^g(u), \tag{4.2}$$

where $g \in \{ego, SC, PC, HC\}$. Thereby, each user is identified by four features, each one of them representing the degree of influence the group has in the user participation history.

Our aim now is to understand if there exists any relation between these features over all the users under investigation. We start analyzing the relation of the ego network with the three communities. Figures 4.9a, 4.9b, and 4.9c relate i_u^{ego} with the group influence values of the other three groups. Each point in the figures represents a user. The y-axis indicates the group influence values related to the ego network parameter, while the x-axis changes with the group under inspection. We observe scattered correlation patterns between the ego network feature and the other three group influence features $(i_u^{SC}, i_u^{PC}, i_u^{HC})$. As expected, the strongest correlation can be observed between i_u^{ego} and i_u^{SC}, as the ego network is a subgraph of the SG. We then evaluate the combination of the three group influence features related to the communities, as shown in Figs. 4.9d, 4.9e, and 4.9f. Every Fig. presents similar patterns among each other, but different distributions and stronger correlations with



(a) i_u^{SC} vs. i_u^{ego}

(b) i_u^{PC} vs. i_u^{ego}

(c) i_u^{HC} vs. i_u^{ego}

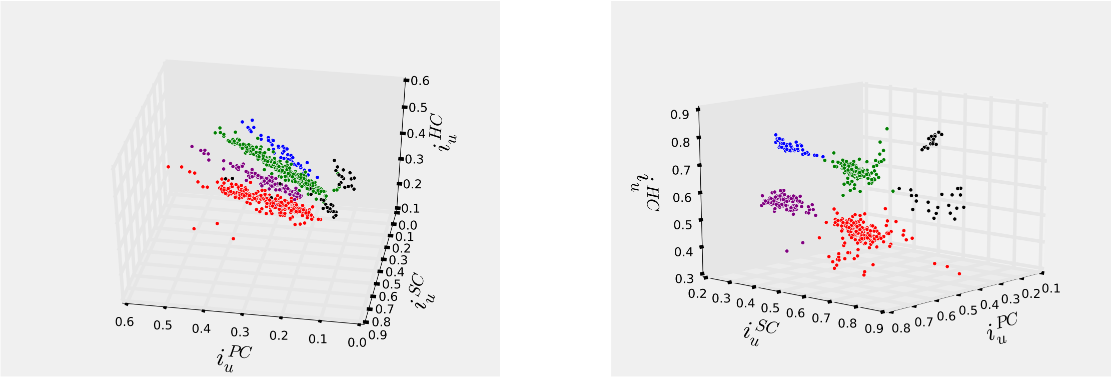
(d) i_u^{SC} vs. i_u^{PC}

(e) i_u^{HC} vs. i_u^{PC}

(f) i_u^{HC} vs. i_u^{SC}

Figure 4.9: Correlation between group influence features

respect to the figures above. From Figs. 4.9d, 4.9e, and 4.9f, it can also be noticed the presence of structures that indicate classes of correlation between every pair of features.



(a)

(b)

Figure 4.10: Group influence features in a 3-D space: Each color distinguishes a *finger*

To further investigate and to better understand these structures, we observe and evaluate the three group influence features in a three-dimensional space, as depicted in Fig. 4.10a. To minimize visual clutter in the 3-D visualization, we do not plot users with values close to zero. From Fig. 4.10a, we can clearly detect structures that look like five fingers. These fingers show a three-fold correlation between the group influence features. This correlation further confirms the interdependence among the three driving factors of human behavior considered in this study, i.e., social relationships, physical location, and homophily. Every user in a finger has the same type of behavior in terms of percentage of influence among the communities. In fact, the ratios between feature values are constant over the finger. This unexpected clustered pattern shows ratios between features that do not vary in $\mathbb{R}^3$, but that assume discrete values in the three-dimensional space. It should be noticed that these fingers overlap with each other when projected onto a 2-D plane, thus, it is possible to recognize only two fingers per plot in Figs. 4.9d, 4.9e, 4.9f.

We observe that each finger can reasonably be conceived as a line passing through the origin. To cluster together points on the same finger, we convert the Cartesian coordinates to spherical ones (r, θ, ϕ) and we project each point on the surface of a unit sphere. This procedure reveals five separate clusters, as depicted in Fig. 4.10b. In such a way, only θ and ϕ are needed to distinguish the points on the five fingers. We utilize the k-means algorithm to perform the partitioning. The number of clusters $k = 5$ has been chosen according to the number of visible fingers. Clustering results are shown in Figs. 4.10a and 4.10b. The colors in the plots distinguish each finger and reveal five classes of users, which from now on we will call *fingers*.

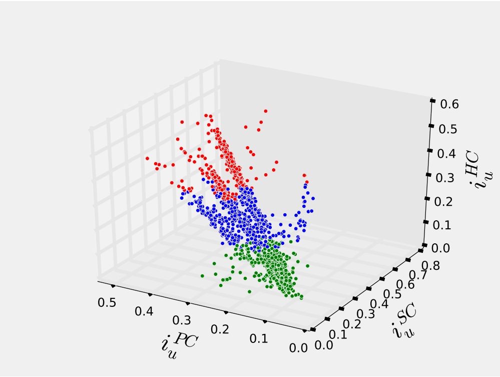


Figure 4.11: Group influence features in a 3-D space: Each color distinguishes a *class of influence*

Furthermore, we propose to cluster users according to the overall level of influence they experienced by jointly considering the group influence features. We aim to partition users in three clusters based on whether they were subject to “low”, “medium”, or “high” degree of social influence. We refer to these partitions as *influence classes*. We use the k-means algorithm ($k = 3$) to group the users into these classes based on the group influence features ($i_u^{SC}, i_u^{PC}, i_u^{HC}$). The results are depicted in Fig. 4.11, which includes also users with group influence feature values close to zero (differently from Fig. 4.10a). Red points represent users *highly* influenced by the groups, green points indicate the *low* influence class, while blue points represent the class of users with *medium* degree of social influence.

The two partitions (i.e., fingers and classes of influence) presented above reflect two different properties in the spherical coordinates (r, θ, ϕ). Each user is characterized by a specific combination of the three features, with the radial variable r characterizing the overall level of social influence, and the angular variables (θ, ϕ) describing specific ratios between pairs of community-features. The level of social influence is determined by many factors related to the users, e.g., involvement in the communities, personal preferences, social relationships, and geographic location. As a consequence, we observe a distribution of values between a minimum and a maximum with a broad dispersion around the mean value. We would have expected a similar distribution also for the angular variables. Instead, we observe a multimodal distribution composed of a small number of centroids, i.e., the five fingers, with a narrow distribution around each of them. These statistical patterns are probably a clue of some interesting sociological and psychological factors driving human behavior. This result

Table 4.8: Performances based on classes of influence

	average	low	medium	high
Accuracy	80%	72%	82%	87%
Precision	80%	71%	82%	88%
Recall	71%	57%	73%	82%

Table 4.9: Performances based on fingers

	average	f_1	f_2	f_3	f_4	f_5
Accuracy	78%	80%	78%	78%	77%	79%
Precision	79%	80%	75%	83%	74%	82%
Recall	67%	67%	69%	68%	65%	66%

confirms that subjects' behavior can be described by a limited number of behavioral phenotypes [195]. To our knowledge, this is the first study that proves the existence of behavioral phenotypes related to the social influence phenomenon.

4.3.5 Activity Prediction based on Behavioral Classes

Finally, we propose to utilize the classes described above (both fingers and influence classes) to predict users' behavior, in terms of event participation. More specifically, we aim to exploit the activity information of class members to infer the activity of other users belonging to the same class. Notice that with the term class we refer both to fingers and influence classes, which have been both defined in Section 4.3.4. We treat each class separately, building a unique model for each cluster of users. In particular, we utilize a subset of the class members in the learning process and the remaining members in the testing phase. We employ the same features used in 4.3.3 and, also in this scenario, we use a 10-fold cross-validation. Table 4.8 and Table 4.9 report the prediction results related to the influence classes (low, medium, high) and fingers (f_1, f_2, f_3, f_4, f_5), respectively. As a baseline, to be compared with these results, we evaluate the prediction performance considering all the users as belonging to the same class. This results in an accuracy of 72%, a precision of 75%, and a recall of 66%.

We can observe that:

- The highly influenced class achieves the best performance. This is reasonable

Table 4.10: Performance of the prediction based on behavioral classes

	average	low	medium	high
Accuracy	82%	74%	83%	89%
Precision	84%	75%	86%	92%
Recall	79%	73%	78%	85%

and it was also expected because every feature is a measure of social influence and, thus, they fit highly influenceable users. For the same reason, the low influence class performs poorly compared to the other two classes of influence.

- Fingers do not exhibit the same behavior of influence classes. There is no finger that outperforms the others. This is understandable because every finger represents a class of correlation, and as such, it includes also users belonging to the low influence class.

- The baseline result is close to the performance related to the fingers. It appears that predictions in finger classes get perturbed from users belonging to different influence classes.

These considerations gave us the idea to combine the 3 classes of influence and the 5 fingers in 3 × 5 sub-classes, named as *behavioral classes*. In such a way we expect to take benefits from both partitions. The results, reported in Table 4.10, confirm our intuition. The behavioral classes outperform all the previous results. The improvement is also reflected in the influence classes: all of them achieve better performance with respect to the outcomes in Table 4.8. This enhancement is due to the further partitioning introduced with the fingers, which allows similar users to be grouped together according to the two properties described in Section 4.3.4.

Overall, we showed how users' activity can be predicted based only on their behavioral class. In fact, in this approach, we do not count on the activity history of the user, while we rely only on the information related to the members of her/his behavioral class. This knowledge has also more predictive power if compared to users' personal data (Table 4.7) and raises significant privacy concerns, which will be discussed in Section 4.4.

4.3.6 Discussion

To answer RQ 2.2, in this Section, we have investigated the impact of different factors on social influence. In particular, we have evaluated the collective influence of physical, homophily, and social communities on users' behavior. Our results have demonstrated the ego network alone is not sufficient to model social influence as other factors play a relevant role in human behavior. We have shown that subjects are also affected by the collective behavior of individuals who (i) live in their geographical area, (ii) have similar interests, or (iii) are in their social community.

4.4 Social Influence and Privacy

The findings obtained throughout this Chapter opens the door to necessary consideration and discussion on users' privacy in OSNs and, in particular, on the relation between social influence and privacy. The results described in Sections 4.2 and 4.3 showed that social influence modeling can be conveniently used as a means to infer individuals' behavior and, in turn, violate their privacy. Users' activity and behavior, in turn, can represent sensible information, which a subject may not be willing to share in some (or future) instances. Although users may not disclose such information, we showed that their privacy is not only in their hands, as the availability of public information in OSN acts as a proxy to predict their behavior. As an example, we can consider the case of a generic user that, after sharing her/his data in the OSN for a certain period of time, decides to stop releasing information about her/his activities on the platform. In such a case, if her/his friends keep sharing their activity in the OSN, a (previously trained) social influence model can be used to infer the undisclosed activity of the user.

To further investigate this argument, we repeat our experiments in a different setting. We evaluate the prediction accuracy of SIDL by varying the probability p that each individual's friend shares the information about a given activity, i.e., if a friend performed a certain activity, we exploit that knowledge with probability p. In this analysis, we use the G-SIDL approach as we are mainly interested in exploring the privacy leakage in OSNs rather than focusing on the interpretability or scalability of the model. In Figure 4.12, we depict the accuracy of our model at varying p. Diverse aspects are worthy of consideration. There is a noticeable gap between the performance in Plancast and Foursquare. This is likely due to a higher involvement of friends in a

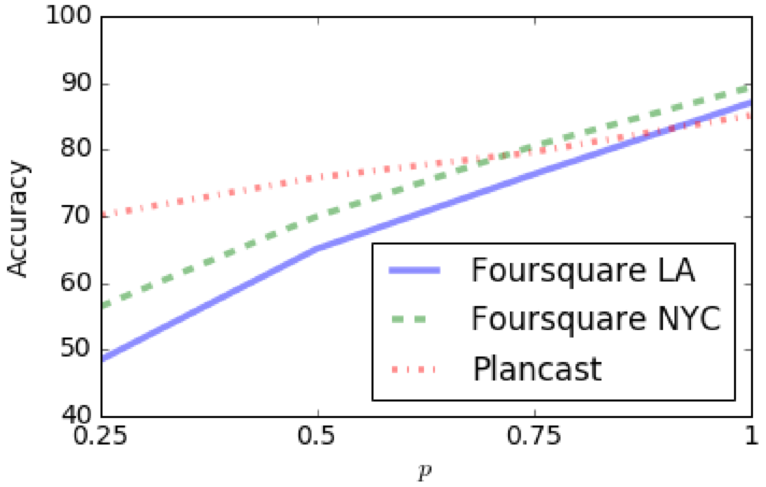


Figure 4.12: SIDL prediction accuracy at varying sharing probability p

social event compared to a visit in a certain location, as we showed in Section 4.2. Further, we can observe that our model, on average, is able to correctly classify about 70% of users' activities in case of $p = 0.5$. This result further highlights the weakness of data privacy in social platforms: with only 50% of available information from users' friends, we are able to classify around 70% of their activities. Such findings corroborate our hypothesis by suggesting that users' information domain is not only confined to what they deliberately share.

To further support our claim, we focus on the results in Section 4.3.5, which show an interesting, and possibly more alarming, scenario. The model presented in Section 4.3 allowed us to discover behavioral classes that group together users with similar behaviors. In Section 4.3.5, we employ such classes to train a machine learning model by using information from a subset of users belonging to a certain behavioral class and test it on another subset of users belonging to the same behavioral class. Basically, we use the activity information of the members of a class to predict the future activities of other members of the same class. Results show that, by following this approach, users' activities can be accurately predicted. In particular, the information provided by other users belonging to the same class has even more predictive power than the activity history of the users themselves. This finding further highlights the data privacy problem and indicates that the issue concerns the entire society. As Garcia [97] claims, we need to stop thinking that the decision to keep information private is under individual control and realize that information secrecy may be affected by the decisions of others.

4.5 Conclusion

In this Chapter, we explore the privacy leakage in OSNs in relation to the social influence phenomenon. To respond to RQ 2.1, we present SIDL, a framework that combines deep learning with network science for modeling social influence and forecasting human real-life activities. We validate and evaluate our approach using datasets from Plancast (an EBSN) and Foursquare (a LBSN). Our experiments reveal that SIDL approaches outperform state of the art baselines achieving an average prediction accuracy of 86.6% (G-SIDL), 80.1% (L-SIDL), and 85.2% (C-SIDL). We show that the opportune combination of network science with deep learning can address both the interpretability and scalability issues while maintaining accurate performance. In fact, C-SIDL provides a post-hoc explanation of the results by identifying a subset of users, which are mainly responsible for the prediction outcome, using social network mesolevel structures. Moreover, C-SIDL closely approaches the performance accuracy of the G-SIDL approach, also providing a more scalable model.

To address RQ 2.2, we show that the ego network alone is not sufficient to explain the social influence phenomenon as other groups, such as physical, homophily, and social communities, are also relevant sources of influence. In fact, our results demonstrate that users are differently affected by the collective behavior of subjects that live close to them, that share the same interests, and that are in their social proximity. According to these factors, we discover a finite number of behavioral classes, which group together users with similar characteristics. These classes can be powerfully exploited to infer users' activities using only the information of other members (not necessarily friends or acquaintances) that belong to the same behavioral class, further highlighting the privacy leakage in OSNs.

Overall, the results of this Chapter demonstrate how influence modeling can be used to infer individual undisclosed activities and corroborate the idea that users' privacy also depends on influence dynamics. This fact highlights the weakness of privacy on online platforms and raises privacy concerns that regard our society as a whole.

5

Detection of Manipulation Campaigns

5.1 Introduction

In Chapter 4, we analyzed social influence, a fundamental factor driving human behavior. We showed that an interested party can infer users' undisclosed actions and violate their privacy by modelling influence strength among them. This, beyond confirming privacy leakages in OSNs, highlights how social relationships and interactions can affect individuals' decisions. The concept that the interplay among users can shape peoples' behavior and belief has been used for other malicious purposes to a large extent. Various studies have discussed the role of social influence in the spreading of information raising awareness about the risk of mass manipulation of public opinion. Examples of OSN manipulation can be found in a variety of contexts, ranging from politics to public health. In the political context, manipulation campaigns have been creating significant concerns for democracy and fairness of political elections. In this context, the 2016 Brexit referendum and the 2016 US Presidential election represent recent remarkable instances of OSN manipulation [9, 21, 36, 71, 130, 131, 239].

Since then, OSN service providers have been increasing their efforts to suspend malicious actors, e.g., bots and trolls (both described in Section 2.4), and maintain a healthy conversation on their platforms. However, malicious activity in OSNs has not entirely stopped: Bots and trolls are still active online [19, 20, 134, 167] and play a pivotal role in manipulation and misinformation campaigns globally. In fact, the detection of such malicious actors in coordinated campaigns is still an open problem for the research community [57, 87, 232]. In particular, bots have been becoming increasingly sophisticated to resemble human appearance and, in turn, avoid detection, while the automated identification of troll accounts has not found any established solutions yet. To tackle these problems, in this Chapter, we aim to respond to the following RQs:

RQ 3.1: *How are social bots evolving to mimic human behavior and avoid detection?*

RQ 3.2: *What are the strategies implemented by bots to manipulate OSN users and are those strategies effective?*

RQ 3.3: *In a analogous way to the detection of bots, is it possible to implement an automated approach for the identification of troll accounts in OSNs?*

Along these research directions, the main objective of this Chapter is to examine the activity of malicious users with the final objective of improving their detection and adapting effective countermeasures. For this purpose, we examine the online behavior of bots and trolls during different political elections in the US. More specifically:

- To address RQ 3.1, we investigate the evolution of bots over the last two US election events, i.e., the 2016 Presidential election and the 2018 Midterms, with the objective of recognizing their scheme to mimic human users and escape detection [168].

- To respond to RQ 3.2, we aim to uncover the strategies implemented by bots for engaging with humans and measure their effectiveness in the latest US election, i.e., the 2018 Midterms [167].

- To tackle RQ 3.3, we analyze the activity of a set of (publicly identified) Russian trolls during the 2016 US Presidential election to characterize their behavior and propose an approach for their automated detection [169].

The contribution of these analyses is two-fold. First, the purpose of the study on bots is to keep the pace of such malicious accounts in order to provide insights that can inform actionable policies for adapting existing countermeasures for their detection. Second, the investigation related to troll accounts aims to build an automated tool for their identification, which nowadays is far from being effective. Notice that we focus on voting events in the US as the 2016 Presidential election represents the first widely recognized case of manipulation campaign in OSNs, where the activity of both trolls and bots have also been unveiled by government bodies (e.g., the US Congress). It is, therefore, of paramount importance to study the evolution and the strategies of malicious actors that interfere in such events to prevent and mitigate their activity in future occurrences.

This Chapter is organized as follows. In Section 5.2, we examine the evolution of bot accounts between the 2016 and 2018 US elections. Section 5.3 studies bots' strategy to interact with humans also focusing on their political leaning. In Section 5.4, we introduce and present an approach based on Inverse Reinforcement Learning (IRL) to detect trolls in OSNs.

5.2 Social Bots Evolution

Social bots, due to their scalable nature, represent a major concern in the fight against OSNs manipulation [38, 178, 213, 231], as further demonstrated by the recent suspension of millions of compromised accounts by Facebook[1] and Twitter[2]. Despite the attempts from OSN providers to suspend suspected, malicious accounts, the presence of social bots does not show any sign of decline [70, 242]. In such a scenario, detecting and keeping the pace of increasingly sophisticated malicious accounts is needed to build and adapt effective countermeasures. For this purpose and, more specifically, to address RQ 3.1, we aim to study the evolution of bots' strategy to mimic human users and avoid detection. Also, understanding how human users deal with the manipulation attempts of these automated accounts is extremely important to stop manipulation campaigns and raise awareness of such peril.

To explore the evolution of both bot and human users, in this Section, we monitor the activity of Twitter accounts engaged in the political discussion during the last two US voting events. We identify bots and characterize their activity in contrast with

[1] https://edition.cnn.com/2019/05/23/tech/facebook-transparency-report/index.html

[2] https://www.bbc.com/news/technology-44682354

humans. In the next subsection, we describe the methodology employed to collect the data, detect bots, and analyze the strategy and evolution of bot accounts over the two voting events.

5.2.1 Methodology

In this subsection, we detail the tools and methodologies used to detect and analyze bots' activities on Twitter during the last two US elections.

5.2.1.1 Data Collection and Processing

We capture the political discussion on Twitter by gathering election-related posts (i.e., tweets) during the two election periods through the Twitter API using a set of keywords as a filter. Different keywords have been selected per each election. For the 2016 US Presidential Election, we make use of the dataset collected in [36], which represents a benchmark in the research community. Tweets have been collected from September 16, 2016 to October 21, 2016 by using the following 23 keywords: #election2016, #elections2016, #tcot, #p2, #hillaryclinton, #donaldtrump, #presidentialdebate, #debates2016, #imwithher, #trump2016, #nevertrump, #neverhillary, #trumppence16, #hillary, #trumpwon, #debate, #trump, #garyjohnson, #jillstein, #jillnothill, #debatenight, #debates, #VPDebate. Overall, 42.1 million tweets generated from 5.9 million users have been gathered. For the 2018 US Midterms, tweets have been collected from October 6, 2018 to November 19, 2018 using the following keywords: #2018midtermelections, #2018midterms, #elections, #midterm, and #midtermelections. As a result, 2.6 million tweets from 997,406 users have been gathered.

In this study, we consider those users who are present in both the datasets to perform a comparative analysis between the two election periods. The rationale is to analyze the evolution of human and bot activity, behavior, and interplay. Thereby, we consider the 278,181 accounts that published tweets both in 2016 and 2018. This subset of users represents a continuum between the two election conversations, other than the core of the online discussion. In fact, these users were involved in 54% and 65% of the tweets collected in 2016 and 2018, respectively. To examine the same time window for the two voting events, data from the two datasets have been filtered considering only tweets ranging from the month before the election to the day following the election.

The 2016 US Presidential election occurred on November 8, 2016, while the 2018 Midterms occurred on November 6, 2018. The filtering results in 8,383,611 tweets from 2016 and 660,296 from 2018, originating in total from 244,699 users.

5.2.1.2 Bots Detection

One of the most important tasks for the uncovering and understanding of OSN manipulation is the identification of automated accounts, i.e., bots. While increasingly sophisticated techniques keep emerging [145], in this study, we rely upon Botometer (presented in Section 2.4), a publicly-available machine learning-based tool maintained by Indiana University [69, 231, 242] to detect automated accounts on Twitter.

For each analyzed user, Botometer outputs a score, namely *bot score*, which ranges from 0 to 1. The lower is the score, the higher is the probability that the user is human. Prior studies used 0.5 as a threshold to separate humans from bots. However, according to the re-calibration introduced in Botometer v3 [242], along with the emergence of increasingly more sophisticated bots, we here use a bot score threshold equal to 0.3 (i.e., a user is labeled as a bot if the score is above 0.3). This threshold corresponds to the same level of algorithmic sensitivity of a score equal to 0.5 in prior versions of Botometer (cf. Fig. 5 from [242]).

Botometer allows us to assign a score to the ~245K accounts. We performed such examination on January 2019. According to Botometer, 12.6% of the scored accounts were classified as bots, 86.1% as humans, while the remaining 1.3% of the accounts were not found on Twitter (indicating users that have deleted their account, have been suspended for violation of the Twitter rules, or have been quarantined by Twitter for further verification). The percentage of discovered bots (12.6%) represent a consistent result with respect to previous studies, e.g., the analysis of the 2016 US Presidential election [36].

5.2.1.3 Sentiment Analysis

To characterize the emotional content generated by both humans and bots, we rely upon sentiment analysis. More specifically, we employ SentiStrength [226] to map each tweet to the sentiment it expresses. SentiStrength is a lexicon-based approach that is conceived for OSNs text analysis. Lexicon-based algorithms are based on sentiment lexicons, dictionary of emotions where words are attributed to a given sentiment strength. SentiStrength attributes a numerical score of sentiment intensity. In particular, it returns two sentiment strengths for determining the level of positive and negative sentiment, which range from 1 to 5 (with 5 being the greatest strength). Overall, we are interested in the total sentiment, thus, we subtract the negative sentiment from the positive one. Thereby, the final score ranges from -4 (most negative) to 4 (most positive). As an example, the sentence "I am happy today even though it is raining" has a positive strength equal to 2 and a negative strength equal to -1. Therefore, the total sentiment is positive (+1).

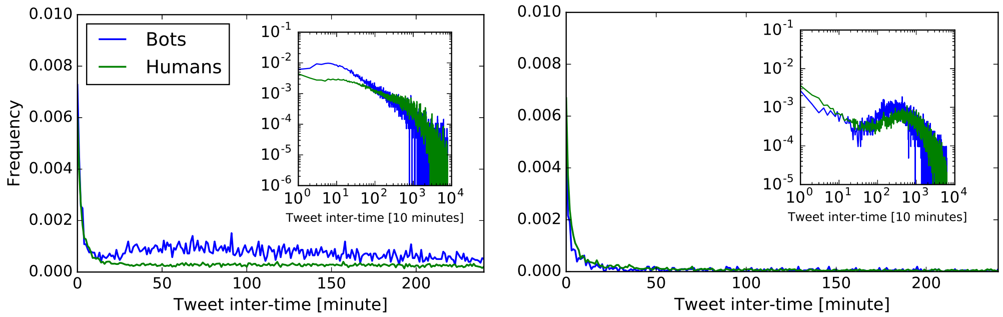
5.2.1.4 Granger Causality

To investigate the interplay between humans and bots, we evaluate whether human interaction with bots can be predicted by leveraging the volume of bots activity. For this purpose, we use the Granger causality test [110] on the time series representing the volume of shared content of these two classes of users.

More in general, Granger causality is used to determine whether time series X can be used to predict time series Y. Granger causality postulates that X "Granger-causes" Y if the predictions of future values of Y based on the combination of the past values of X with the past values of Y are better than the predictions of Y based only on the past values of Y. This holds true unless also the reverse (Y Granger-causes X) is verified. In such a case, no conclusion can be drawn. We further apply a differentiation to remove seasonal effects and, then, we tested the stationary time series. The autoregression of Y is augmented by lagged values of X and those individually significant (t-statistic) that increase the explanatory power of the regression (F-test).

5.2.2 Results of the Analysis

In this subsection, we discuss the results of our study. Based on the collected data, we perform an analysis of the evolution of bot and human accounts over the two election periods considered in this investigation. We first examine the temporal dynamics and the volume of the content shared by human and bot accounts. Then, we focus on bots' strategy to avoid detection and, finally, on their interaction with humans.



(a) 2016 Election

(b) 2018 Election

Figure 5.1: Posting inter-time for bots and humans

Table 5.1: Volume (in percentage) of sharing activities of humans and bots in the two election periods

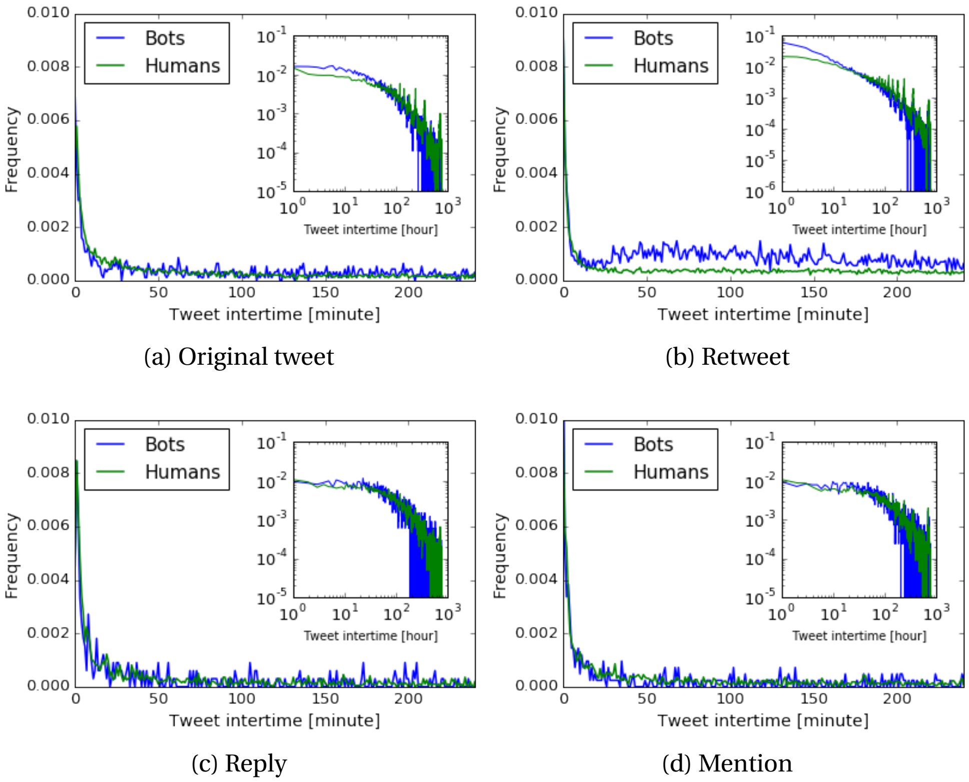
	Humans		Bots	
	2016	2018	2016	2018
Original Tweet	0.16	0.15	0.12	0.15
Retweet	0.76	0.72	0.83	0.79
Reply	0.04	0.09	0.02	0.03
Mention	0.04	0.04	0.02	0.03

5.2.2.1 Temporal dynamics

We explore bot and human dynamics by measuring the time lag between consecutive sharing activities (i.e., tweets). Figure 5.1 displays the inter-time tweet distribution comparing bot and human users in the 2016 and 2018 elections. Notably, in the 2016 election, the distribution of bots' tweet activity largely differs from the humans' distribution[3]. The discrepancy is particularly relevant in the time range between 10 minutes and 3 hours, consistent with other findings [197]: in 2016, bots shared content at a higher rate with respect to human users. On the other hand, in the 2018 Midterms, inter-time distributions are similar, suggesting that bots have been refined to emulate human timing.

To better understand the impact of each type of sharing activity on this result, we disentangle Twitter posts in *original tweets* (i.e., original content generated by users), *retweets* (i.e., re-share of the original content generated by other users), *replies* (i.e., response to a post), and *mentions* (i.e., involvement of other users in a post). Figure 5.2

[3] Note that, to validate our findings, we repeated this evaluation at varying bot score threshold (from 0.3 to 0.7) with no significant changes on the results.



(a) Original tweet

(b) Retweet

(c) Reply

(d) Mention

Figure 5.2: Inter-time distribution of the different sharing activities for bots and humans in 2016

depicts the inter-time distribution of the above sharing activities in the 2016 election, while no relevant discrepancy can be noticed in these sharing activities in the 2018 Midterms. From Figure 5.2, it is noticeable that the inter-time distributions of original tweets (Fig. 5.2a) and retweets (Fig. 5.2b) present the two principal gaps between humans and bots in 2016. This finding is in line with the established bots' strategy consisting of overwhelming online platforms with a high volume of original tweets and retweets, as shown in previous studies [36, 86]. The fact that such gaps are not present in the distributions related to the 2018 Midterms election further confirms that bots have been ameliorated to mimic human behavior and escape detection.

5.2.2.2 Activity Volume

To further investigate the nature of the difference in the temporal dynamics, we measure the volume of each sharing activity in the two election periods. In Table 5.1, we show the percentage of each activity over all the content shared by bots and humans.

It can be noticed that both humans and bots significantly diminished the amount of retweets in the 2018 Midterms (t-test results: $t(5,840,537) = 64.6$, $p < .001$ for humans and $t(3,083,630) = 42.4$, $p < .001$ for bots), while the percentage of mentions does not exhibit a noticeable variation in both groups of account. Additionally, humans have doubled the number of replies (t-test: $t(5,840,537) = 152.3$, $p < .001$) in 2018 and bots generated more original tweets (t-test: $t(3,083,630) = 29.8$, $p < .001$) in 2018 than in 2016.

The growing propensity of humans to discuss a post (either positively or negatively) instead of simply re-sharing the content generated by other accounts, represents an encouraging finding. This insight acquires even more significance considering the cost related to each form of interaction. While retweeting is a one-click operation, with a relatively small human cost in terms of time and effort, a reply requires a larger undertaking. From the perspective of bots, a retweet can be programmatically executed with one command line; however, programmatically composing a meaningful reply requires the use of sophisticated natural language models, such as those based on DNNs [198], which often require significant computing resources for training.

Interestingly, although bots increased the number of original tweets (with respect to the 2016 election) and, comparably to humans, reduced the number of retweets, the gap in the inter-time distribution in 2018 appears to be reduced, suggesting a more cautious broadcasting strategy adopted by bots. To shed light on this result and, more in general, to explore the evolution of the actions of both humans and bots on Twitter, we now analyze separately the three sharing activities that showed more variation between the two election periods, i.e., original tweet, retweet, and reply.

5.2.2.3 Original Tweet Activity

As far as original tweets are concerned, we found an anomalous pattern in the way bots published their content. Both in 2016 and 2018, bots published multiple times the content they generated. While in 2016 this repeated activity was mainly performed by each bot separately, i.e., each message was published multiple times by the same bot, in 2018 this repeated activity was also shared between bots, i.e., multiple bots shared the same message once. More specifically, during the 2016 (resp. 2018) election period, 5% (resp. 2.4%) of the tweets generated by bots were published more than once by the same author. The significant drop (t-test: $t(388,196) = 17.4$, $p < .001$) in

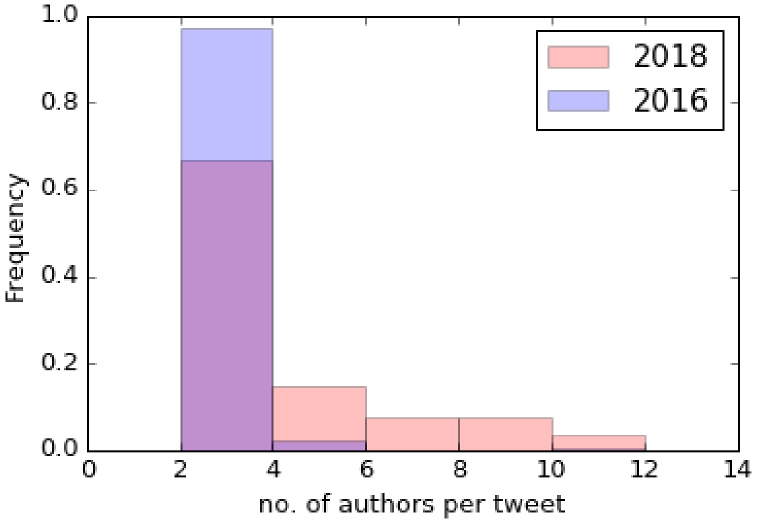
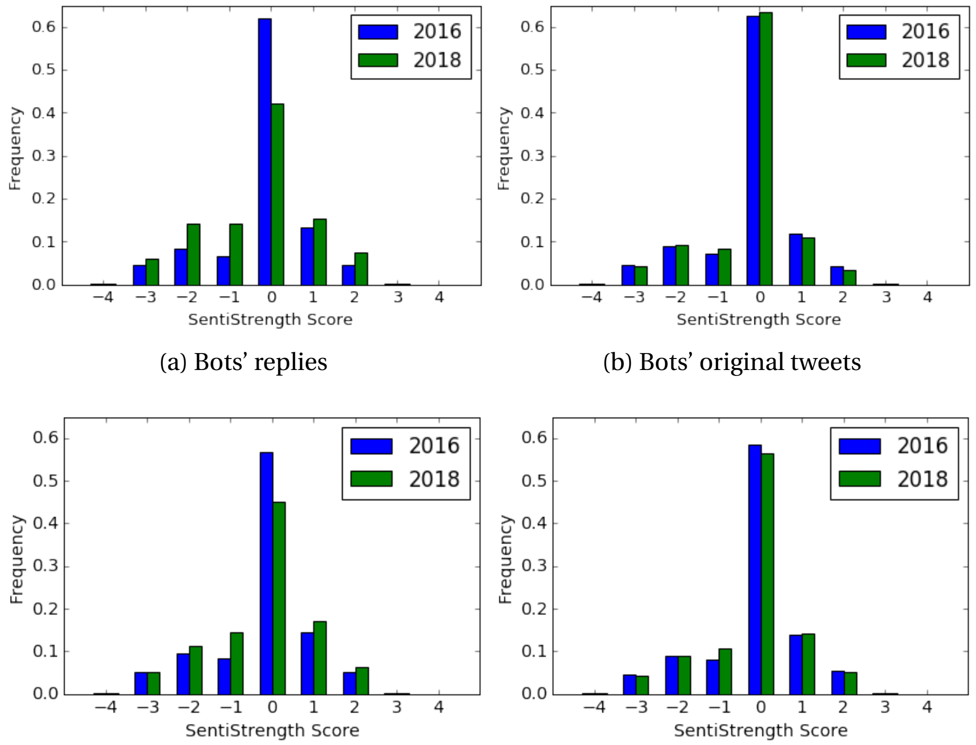


Figure 5.3: Probability distribution of multiple authors sharing the same content during the 2016 and 2018 election periods

this repeated activity during the 2018 election is, however, replaced by an increasing number of single sharing operations (of the same content) performed by multiple bots, possibly in the context of a coordinated effort.

In Figure 5.3, we investigate this scenario by showing the probability distribution related to the number of multiple authors sharing the same content in 2016 and 2018. Although the majority of content is shared by a few bots (from 2 to 3) in both periods under consideration, it is noticeable how in the 2018 election the number of authors sharing the same content increased. To evaluate the statistical difference between the sets of multiple authors in 2016 and 2018, we employ the Mann-Whitney rank test, which in turn corroborates our intuition (p-value < .001). The distributed sharing activity among bots can be conceived as a strategy to avoid detection. In fact, the replication of the same content represents one of the most common signals to identify automated accounts [231, 242]. Therefore, an account that repeatedly publishes the same content is simpler to classify as a bot with respect to accounts that share diverse content. Also, this multi-bot strategy can be used to promote ideas or information by creating the illusion of a consensus among the OSN population, as multiple accounts spreading the same information can elicit the idea of people sharing the same opinion [86].



(a) Bots' replies

(b) Bots' original tweets

(c) Humans' replies

(d) Humans' original tweets

Figure 5.4: Sentiment scores of replies and original tweets for bots and humans

5.2.2.4 Reply Activity

We investigate whether the same repeated activity is also recognizable in the replies provided by bots to other accounts' posts. We found some instances of multiple replies in both the periods under investigation, but in both cases with a limited extent (around 1.1% of replies were repeated).

To characterize the emotional content of the replies generated by bots and humans, we rely upon sentiment analysis and in particular on SentiStrength (introduced in Section 5.2.1.3). In Figure 5.4, we show the sentiment score distribution of bots' and humans' replies, in contrast with the sentiment score of their original tweets in the two election periods under analysis. Two facts are worth noting. First, from Figs. 5.4a and 5.4c can be noticed that both humans and bots shared less neutral (SentiStrength score = 0) replies in 2018. Second, the gap in the sentiment scores between the two election periods is more evident for bots (Figs. 5.4a and 5.4b), especially for negative replies, with the 2018 period exhibiting significantly more negative replies than 2016. With respect to original tweets, no remarkable difference exists between 2016 and 2018 regardless of the source (human or bot). Overall, the difference between the score sentiment of replies and original tweets is only noticeable in the 2018 election period, while the distributions are similar for the 2016 election. From these observations, we conclude that bots exhibited a more inflammatory and negative behavior in their conversation (i.e., replies) during the 2018 election with respect to the 2016 election, while they showed a similar attitude in the sharing of original content. This can be viewed as a strategy to engage with humans and gain their endorsement by strongly supporting a given faction in a conversation. In fact, participating in an online discussion allows a direct interaction among users that is different from sharing original content, which in turn can reach a limited number of accounts and does not necessarily entail interplay among users.

5.2.2.5 Retweet Activity

To examine the re-sharing activity (i.e., retweet) in the two voting periods, we initially consider the top retweeted posts created by bots and shared by humans. By analyzing the top 10 retweets in this subset, we notice an interesting difference between the two periods: On one hand, in the 2016 Presidential election, the majority of the retweeted posts were in support of candidate Trump, and in opposition to candidate Clinton [36]. On the other hand, in the Midterms, the most retweeted posts of bot-generated content were tweets aimed at surveying the Twitter population. From now on, we refer to these tweets as *poll-tweets*. In Table 5.2, we show the poll-tweets in the top 10 of human retweets of bot-generated content. Note that, for privacy reasons, we anonymized (using @???) the user names of the accounts associated with the originators of these examples, while we show the user names of the accounts that have been suspended for the violation of Twitter's rules. The objective of these tweets seems to have a quantitative understanding of voter turnout, political leaning, and preferences. This exploratory attitude of bots in the Midterm elections appears in contrast with the behavior towards the candidates shown in the 2016 Presidential election.

Although poll-tweets seem harmless and aimed only at surveying human opinion, their turnout might impact the human perception on the polled issues. To understand whether bots fostered the spread of poll-tweets, we analyzed the most retweeted content by bots. Results show that the most re-shared (i.e., retweeted) post is a poll-tweet, and 4 additional poll-tweets are in the top 10 retweeted content by bots. In Figure 5.5, we depict the timeline of the hourly volume of retweets shared by bots and

Table 5.2: Poll-tweets in the top 10 human retweets of bot-generated content

Poll-tweets	**F-test**	**steps**
@kwilli1046: Which Party Do You Plan To Vote For In The 2018 Midterm Election? Please vote and retweet for bigger sample size	22.8	1
@The_Trump_Train: RT if you agree: We need ICE agents at every polling station during elections.	15.2	1
@???: With all what's going on, if elections is today, would you vote for @realDonaldTrump? Please vote and Retweet.	15.1	3
@kwilli1046: Should voters in federal elections be required to show ID at the polls? Please vote and retweet for bigger sample size	2.9	6
@???: IF, AND ONLY IF, YOU ARE VOTING ON NOV 6, 2018, please answer this poll.	14.3	2
@Golfman072: As of Sept. 30th-Oct 5th how likely are you to vote in the MID-TERM elections? #Redwavepolls	20.7	1
@Golfman072: As of Oct 7th-13th how likely are you to vote in the MID-TERM elections? #Redwavepolls	4.5	1

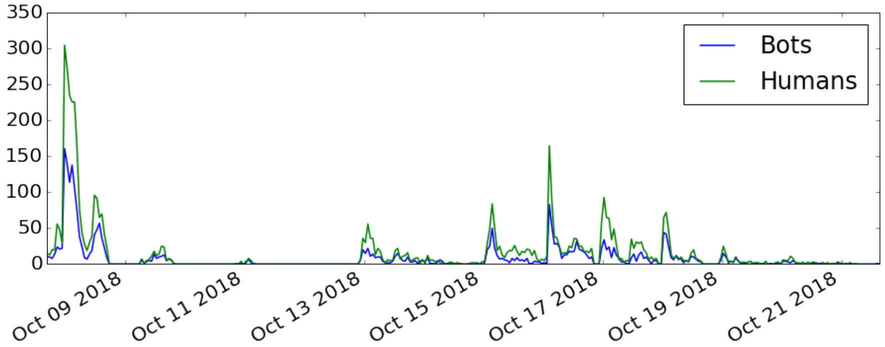


Figure 5.5: Timeline of the retweet volume of the poll-tweets shared by bots and humans

humans related to the poll-tweets listed in Table 5.2. Notice that we do not display the volume of retweets in the days after October 21 as the number of these tweets during that period is negligible. Interestingly, human users participated in these polls to a larger extent with respect to bots, which in turn were responsible for the creation and sharing of these posts.

Additionally, to investigate whether bots' retweet activity predicts humans participation in these polls, we evaluate the Granger causality (defined in Section 5.2.1.4) between the hourly volume of poll-tweets shared by bots and the hourly volume of poll-tweets shared by humans. Results of the Granger-causality test are displayed in Table 5.2, where the first column displays the content of the poll-tweet, the second column shows the value of F-test for significant causality, and the third column depicts the temporal lag (in hours) for which the causality has the highest F-test value. These results corroborate our hypothesis: For each poll-tweet under investigation, the volume of retweets from bots is Granger-cause of the volume of human retweet. This result suggests that, while bots acted as initial spreaders of poll-tweets, the human population was affected by the mass involvement in these polls and actively participated in and interacted with bots' poll-tweets. This finding evidences and further confirms the powerful influence of bots in online discussions.

Interestingly, three accounts that created the most successful polls (@kwilli1046, @The_Trump_Train, and @Golfman072) have been suspended by Twitter. Additionally, another suspended account classified as human (@MikeTokes) published a poll-tweet ("NATIONAL POLL: you are voting in the November 6, 2018 elections, what party are you voting for and why?") that received a large number of retweets from bots (top 3 of bots-retweet from human-generated content). This may indicate a combined approach leveraging both human (e.g., trolls) and bot activities.

5.2.2.6 Bots' Targeted Interaction

To further explore the retweet interactions from bots to humans (i.e., bots retweeting human content), we measure to what extent bots targeted humans within the social network. Prior work shows that bots target the most connected humans [220]. In particular, the number of incoming edges (e.g., followers) is often associated with the influence and the centrality that each user has in the social network. For this purpose, we compute the in-degree centrality (detailed in Section 2.7) of the retweet network. We here recall that the in-degree centrality is a network analysis measure that assigns a score to every node in a network based only on the number of inbound links held by each node. In our scenario, the rationale is to understand whether bots interacted mainly with the most retweeted (influential) humans, whose endorsement can be beneficial to spread information across the social network. This, in turn, might indicate a strategy for increasing the resonance of bots' content, as influential nodes can reach a large number of accounts and, thus, spread information to a wide audience. The nodes of the retweet network are the users, while every link of the network represents a retweet. In this context, the in-degree centrality measures the incoming interactions of each user over all the interactions. Hence, accounts with high in-degree centrality indicate users whose content has been largely re-shared and, thus, represent highly influential nodes in the social network.

In Figure 5.6, we display the violin plot distribution of the in-degree centrality related to the humans targeted by bots in the two voting periods. A violin plot is a method to visualize the distribution of numerical data and its probability density. While in the 2018 Midterms most of the probability mass is in the range between 0 and 0.01 (low centrality score), in the 2016 election bots also targeted a considerable set of humans with larger centrality scores. On average, we find that the humans targeted by bots in 2016 has in-degree centrality scores two times larger with respect to the humans targeted in 2018 ($1.7 \cdot 10^{-2}$ vs. $8 \cdot 10^{-3}$, p-value < .001). We conclude that in 2016, bots supporting the Presidential candidates were interested in attracting the attention of the most influential human users in the social network, while during the Midterms

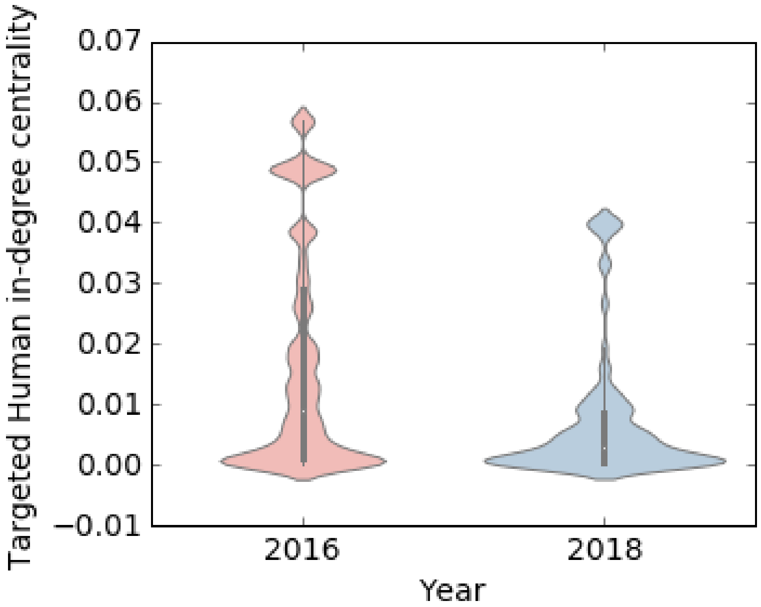


Figure 5.6: Distribution of in-degree centrality of humans targeted by bots

they interacted with any targets regardless of their position and relevance in the social network, further confirming their exploratory attitude during the 2018 Midterms and their mutable nature over different election periods.

5.2.2.7 Causal modeling of human-bot interactions

To further investigate the human-bot interplay, we evaluate the causality between bot and human interactions. We measure the Granger causality [110] of the daily volume of retweets between humans and bots. Our results show that in the 2018 Midterms retweets from bots to humans (i.e., bots retweeting human generated content) were Granger-cause (F-test = 3.51, p-value = 0.02, steps = 5 days) of the retweets from humans to bots, while there was no signal of causality revealed in the 2016 election.

In the 2016 Presidential election, users evidently re-shared content disregarding the authenticity of the information and its source [9, 36]. On the other hand, in 2018 humans likely engaged with bots as a consequence of bots prior interaction with them. Furthermore, in 2018, the volume of retweets from bots to humans was also Granger-cause (F-test = 4.46, p-value = 0.01, steps = 5 days) of the retweets from bots to bots, suggesting that bots strategically distribute and organize their interactions with other bots.

5.2.3 Discussion

In this Section, to address RQ 3.1, we have studied bots strategy to mimic human online behavior and avoid detection. Our results revealed bots evolution and mutable nature over the last two election events in the US. More specifically, we have shown how, in the 2018 Midterms, bots changed the volume and the temporal dynamics of their online activity to better mimic humans and avoid detection. For the same purpose, bots exhibited a more cautious broadcasting strategy, which also allows them to elicit the illusion of public consensus. We have also noticed a relevant reduction in the usage of retweets, both from human and bot accounts. Human users significantly increased the volume of replies, which denotes a growing propensity of humans in discussing (either positively and negatively) their ideas instead of simply re-sharing content generated by other users. This is a positive sign, since the spread of low-credibility content during the 2016 US Presidential election has been often associated with indiscriminate re-sharing [36, 213, 234].

While the increase in usage of replies, along with the reduction of retweets, may represent an encouraging step forward for the fight against misinformation, this intuition should be contextualized considering social media history and development over the last few years: Prior to the investigations into the 2016 US Presidential election, most OSN users may have not been aware of the existence of malicious and/or automated accounts. This may have changed over the course of the last few years. Although the observed change exhibited by human users in 2018 provides an optimistic perspective, there exists an inevitable interplay between the behavior of human users and bots. The mutable nature of bots, coupled with their continuous online presence, is a cause of concern when considering the integrity of the online information ecosystem, especially with respect to online political discussions concerning voting events all over the world.

This set of open problems poses numerous challenges in the fight against OSN abuse and motivates further research for a better understanding of bots' behavior, strategies, and effectiveness.

5.3 Social Bots Partisan Behavior

Section 5.2 highlighted the need for further research to keep the pace of bots strategy and mutable behavior. In particular, understanding and uncovering how bots manipulate human users is of paramount importance for fighting the manipulation of OSNs. In this Section, to respond to RQ 3.2, we face this problem from a different angle. We explore bots partisan behavior (i.e., their endorsement towards a political faction) by evaluating their political leaning and, accordingly, we investigate their strategy for engaging with humans and infiltrating political discussion. We focus on the online discourse during the latest US election, i.e., the 2018 Midterms, as we are interested in studying the activity, and corresponding impact, of the most recent and sophisticated bot accounts. With the final goal of measuring the effectiveness of bots manipulation attempts, we pursue the following steps.

First, we investigate whether social bots lean and behave according to a political ideology (i.e., liberal or conservative leaning). In line with this idea, we explore to what extent they act similarly to human users with the same political inclination (from now on simply *human counterpart*). Second, we explore bots' strategy to manipulate humans and we examine whether some differences can be observed between the strategies of liberal and conservative bots. For this purpose, we measure bots' activity in terms of volume and frequency of posts, interactions with humans, and embeddedness (described in Section 2.7) in the social network. Finally, we estimate the effectiveness of bots' strategies in involving humans in their conversation and evaluate the degree of human interplay with social bots.

In the next subsection, we present the methodology employed to carry out this analysis and we introduce the metrics used to measure the effectiveness of bots' strategy in engaging with humans.

5.3.1 Methodology and Metrics

In this subsection, we describe the tools and methodologies used to detect bots, analyze their political leaning, and measure the effectiveness of their strategies during the 2018 US Midterms.

5.3.1.1 Data and Bot Detection

In this analysis, we leverage the Twitter dataset described in Section 5.2.1.1 to study the online conversation during the 2018 US Midterms election and analyze social bot activity and interactions with humans. Differently from Section 5.2, where the analysis was focused on the set of users that shared content during both the 2016 and

Table 5.3: Dataset statistics

Statistic	Count
# of Original Tweets	452,288
# of Retweets	1,869,313
# of Replies	267,973
# of Users	997,406

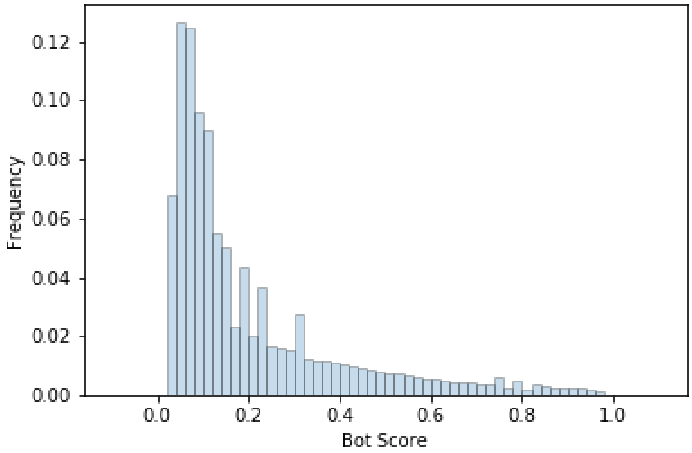


Figure 5.7: Bot score distribution

2018 elections, in this Section we consider the full set of users involved in the tweets collected during the 2018 Midterms. We report in Table 5.3 additional details of the 2.6 million tweets collected for 42 days around the election day from nearly 1 million users.

Similarly to Section 5.2, we rely upon Botometer to detect social bots. In Figure 5.7, we depict the bot score distribution of the 997,406 distinct users in our dataset. The distribution exhibits a right skew: most of the probability mass is in the range [0, 0.2] and some peaks can be noticed around 0.3. According to prior studies, and similarly to Section 5.2.1.2, we use the 0.3 threshold to separate humans from bots.

According to this choice, we classified 21.1% of the accounts as bots, which in turn generated 30.6% of the tweets in our data set. Overall, Botometer did not return a score for 35,029 users that corresponds to 3.5% of the accounts. We used the Twitter API to further inspect them. Interestingly, 99.4% of the 35,029 accounts were suspended by Twitter, whereas the remaining percentage of users protected their tweets turning on the privacy settings of their accounts.

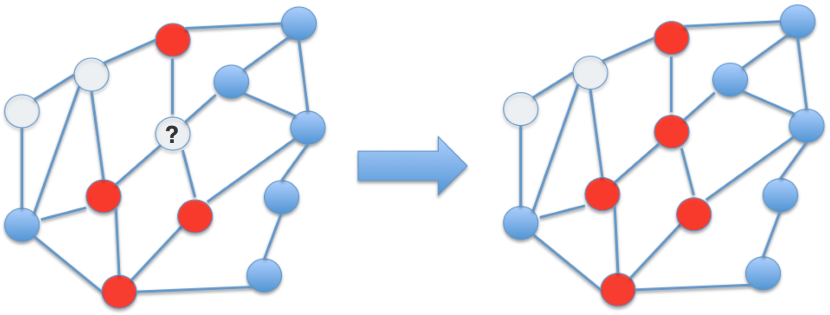


Figure 5.8: Label propagation example

5.3.1.2 Political Ideology Inference

In parallel to the bot detection analysis, we examine the political leaning of both bots and humans in our dataset. To classify users based on their political ideology, we capture and analyze the URLs they shared. Specifically, we rely on the political leaning of the media outlets (e.g., newspaper, TV station, etc.) related to the URLs published in users' tweets. We make use of a list of partisan media outlets released by third-party organizations, such as AllSides[4] and Media Bias/Fact Check[5]. We combine liberal and liberal-center media outlets into one list (composed of 641 outlets) and conservative and conservative-center into another (composed of 398 outlets). After cross-referencing the shared URLs with the media outlets URLs, we observe that 32,115 tweets in the dataset contain a URL that points to one of the liberal media outlets and 25,273 tweets with a URL pointing to one of the conservative media outlets.

To label Twitter accounts as liberal or conservative, we use a polarity rule based on the number of tweets they produce with links to liberal or conservative sources. Thereby, if an account has more tweets with URLs pointing to liberal sources, it is labeled as liberal and vice versa. Although the overwhelming majority of accounts include URLs that are either liberal or conservative, we remove any account that has an equal number of tweets from each side. Our final set of labeled accounts includes 38,920 users.

Finally, we use label propagation (introduced in Section 2.7) to classify the remaining accounts in a similar way to previous work [20]. Label propagation is a network-based algorithm, where each node is assigned a label, which is updated iteratively based on the labels of node's network neighbors. A node takes the most frequent label

[4] https://www.allsides.com/media-bias/media-bias-ratings

[5] https://mediabiasfactcheck.com/

of its neighbors as its own new label, as shown in the example in Fig. 5.8, where edges represent social connections, while red and blue nodes indicate conservative and liberal users, respectively. The algorithm proceeds updating labels iteratively and stops when the labels no longer change. For this purpose, we construct a social network based on the retweets exchanged between users. The nodes of the retweet network are the users, which are connected by a direct link if one user retweeted a post of another user. To validate the results of the label propagation algorithm, we apply a 5-fold stratified cross validation to a set composed of 38,920 seed accounts. We train the algorithm using 80% of the seeds and we evaluate the performance on the remaining 20%. Finally, we compute precision and recall by reiterating the validation of the 5-folds. Both precision and recall scores show value around 0.89 with bounds from 0.88 to 0.90. Both the scores for liberals are about 0.87 with 0.85-0.88 bounds, while for conservatives the scores are around 0.93 with 0.92-0.93 bounds. To further validate the proposed approach, we use as ground truth the political leaning of the media outlet that users shared in their profile, obtaining precision and recall scores in line with the previous approach.

5.3.1.3 Bots' Strategy Effectiveness

We introduce four metrics to estimate the effectiveness of bots' strategy in involving humans and, at the same time, measure to what extent humans rely upon, and interact with the content generated by social bots. Thereby, we propose the following metrics:

- *Retweet Pervasiveness* (RTP) measures the intrusiveness of bot-generated content in human-generated retweets:

$$RTP = \frac{\text{no. of human retweets from bot tweets}}{\text{no. of human retweets}} \tag{5.1}$$

- *Reply Rate* (RR) measures the percentage of replies given by humans to social bots:

$$RR = \frac{\text{no. of human replies to bot tweets}}{\text{no. of human replies}} \tag{5.2}$$

- *Human to Bot Rate* ($H2BR$) quantifies human interaction with bots over all the human activities in the social network:

$$H2BR = \frac{\text{no. of humans interaction with bots}}{\text{no. of humans activity}}, \tag{5.3}$$

where the numerator counts for human replies/retweets to/of bots generated content, while the denominator is the sum of the number of human tweets, retweets, and replies.

- *Tweet Success Rate* (TSR) is the percentage of tweets generated by bots that obtained at least one retweet by a human:

$$TSR = \frac{\text{no. of tweet retweeted at least once by a human}}{\text{no. of bots tweets}} \tag{5.4}$$

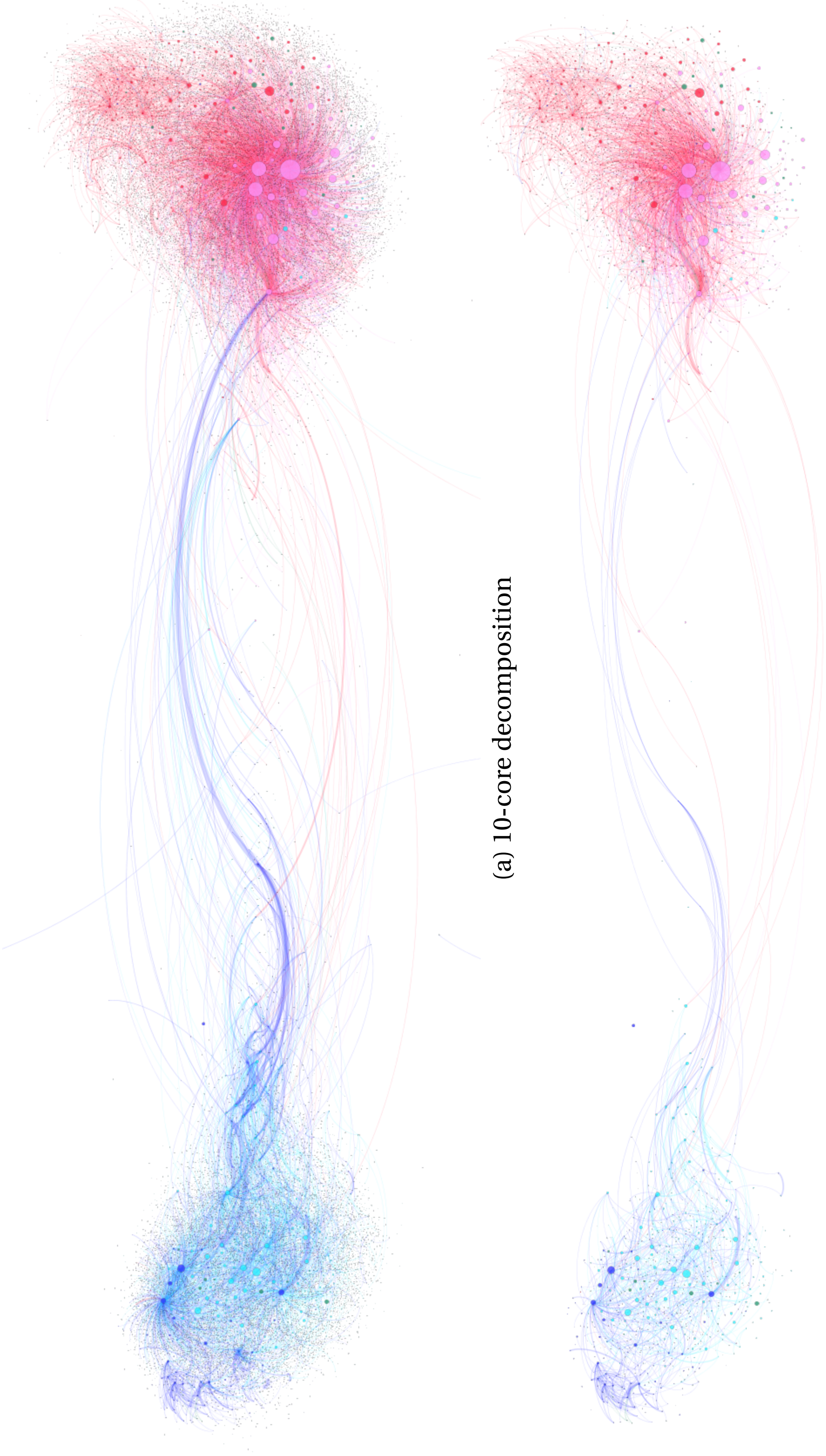
5.3.2 Results of the Analysis

In this subsection, we describe the results of our analysis. In particular, we examine bots' political leaning and, accordingly, we analyze their strategies and measure the effectiveness of their actions in terms of human engagement.

5.3.2.1 Bots' Political Leaning

The combination of the outcome from the bot detection algorithm (i.e., Botometer) and the political ideology inference allowed us to identify four groups of users, namely Liberal Humans, Conservative Humans, Liberal Bots, and Conservative Bots. In Table 5.4, we show the percentage of users per group. Note that percentages do not sum up to 100% as either the political ideology inference was not able to classify every user, or Botometer did not return a score, as we previously mentioned. In particular, we were able to assign a political leaning to 63% of bots and 67% of humans. We find that the liberal user population is almost three times larger than the conservative counterpart, similarly to other existing works [20, 21]. This discrepancy is also present, but less evident, for the bot accounts, which exhibit an unbalance in favor of liberal bots. Further, we investigate the suspended accounts to inspect the consistency of this result. The inference algorithm attributed a political ideology to 63% of these accounts, which shows once again the liberal advantage over the conservative faction (45% vs. 18%).

Figure 5.9 depicts the network of online interactions among Twitter users during the election period. In particular, the interaction network displayed in Fig. 5.9 captures the exchanged retweets among pairs of users. Thus, we refer to this interaction network



(a) 10-core decomposition

(b) 25-core decomposition

Figure 5.9: Political discussion over (a) the 10-core, and (b) the 25-core decomposition of the retweet network

Table 5.4: Number (percentage) of users per group

	Liberal	Conservative
Humans	386,391 (38.7%)	122,761 (12.3%)
Bots	82,118 (8.2%)	49,488 (4.9%)

as the *retweet network*. Specifically, Figure 5.9 shows two retweet networks. We first describe the elements in the retweet network and then we clarify the differences between the two displayed networks in Fig. 5.9.

In the retweet network, nodes represent Twitter users and links represent retweets among them. We indicate as *source* the user that retweeted the tweet of a *target* user. Links with weight (i.e., frequency of occurrence) smaller than 2 are hidden to minimize visual clutter. Blue nodes represent liberal accounts, while red nodes indicate conservative users. Darker tones (blue and red) depict bots, while lighter tones (cyan and pink) relate to humans, and the few green nodes represent unclassified accounts. The link takes the same color of the source node (author of the retweet), whereas node size is proportional to the in-degree (defined in Section 2.7) of the user. Nodes are spatially distributed according to a force-directed layout [135]. Such a scheme arranges nodes in a network by simulating a force that attracts (i.e., to move close to each other) nodes connected by an edge and repulse (i.e., to move away from each other) disconnected nodes. In our setting, this means that users are spatially distributed according to the number of retweets exchanged between each other. Therefore, users that retweet (resp. do not retweet) each other are displayed close (resp. distant) to each other. The output of the force-directed layout [135], in Fig. 5.9, results in a network naturally split into two communities, where each side is almost entirely populated by users with the same political ideology. As mentioned above, Fig. 5.9 shows two retweet networks. Specifically, Fig. 5.9 depicts two *k*-core decomposition (detailed in Section 2.7) graphs of the retweet network. In a *k*-core, each node is connected with at least *k* other nodes. Figures 5.9a and 5.9b capture the 10-core and 25-core decomposition, respectively. The purpose of showing the *k*-core decomposition is to understand the position of bots and humans within the retweet network for every political wing.

Overall, three facts are worth noting: *(i)* bots are embedded, with humans, in each political side, suggesting that bots support and behave according to a political ideology; *(ii)* as *k* increases, the left k-core appears to disrupt, while the right k-core remains

Table 5.5: Top 20 hashtags generated by liberal and conservative bots

Liberal Bots	Conservative Bots
#MAGA	#BrowardCounty
#NovemberisComing	#MAGA
#TheResistance	#StopTheSteal
#GOTV	#WalkAway
#Florida	#WednesdayWisdom
#ImpeachTrump	#PalmBeachCounty
#Russia	#Florida
#VoteThemOut	#QAnon
#unhackthevote	#KAG
#FlipTheHouse	#IranRegime
#RegisterToVote	#Tehran
#Resist	#WWG1WGA
#ImpeachKavanaugh	**#Louisiana**
#GOP	#BayCounty
#MeToo	#AmericaFirst
#AMJoy	#DemocratsAreDangerous
#txlege	#StopTheCaravan
#FlipTheSenate	**#Blexit**
#CultureOfCorruption	**#VoteDemsOut**
#TrumpTrain	#VoterFraud

well connected, suggesting that conservatives are more central users in the online discussion if compared to liberals; and, *(iii)* as k increases, bots appear to outnumber humans, suggesting that bots may populate areas of the retweet network that are more central and better connected.

Next, we examine the topics discussed by social bots and compare them with their human counterparts (i.e., human users with the same political inclination). Table 5.5 shows the top 20 hashtags utilized by liberal and conservative bots. We highlight (in bold) the hashtags that are not present in the top 50 hashtags used by the corresponding human group to point out the similarities and differences among the groups. In this table, we do not take into account hashtags related to the keywords used in the data collection (such as #elections, #midterms), and hashtags used to support the political group (such as #democrats, #liberals, #VoteRed(or Blue)ToSaveAmerica) as (i) the overlap between bot and human hashtags is noticeable when these terms are considered, and (ii) we aim to narrow the analysis to specific topics and inflammatory content, inspired by [220]. Moreover, we used an enlarged subset of hashtags for the human groups to further strengthen the differences and, at the same time, to better

Table 5.6: Number (percentage) of tweets per group

	Liberal	Conservative
Humans	957,726 (37.0%)	476,231 (18.4%)
Bots	288,659 (11.1%)	364,727 (14.1%)

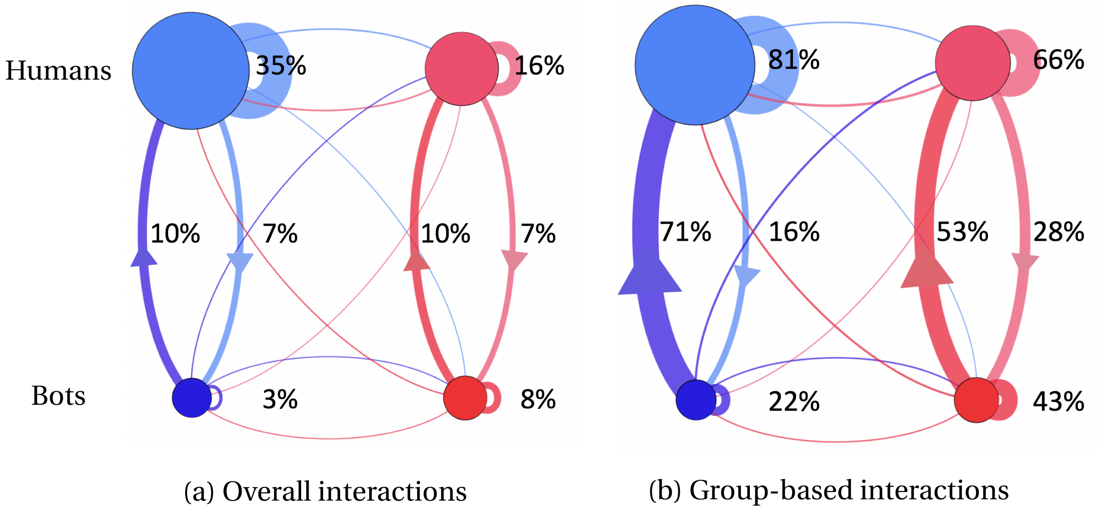
understand the objective of social bots. Although bots and humans share the majority of hashtags, two main differences can be noticed. First, conservative bots abide by the corresponding human counterpart more than the liberal bots. Second, liberal bots focus on more inflammatory and provocative content (e.g., #ImpeachTrump, #ImpeachKavanaugh, #FlipTheSenate) with respect to conservative bots.

5.3.2.2 Bots' Strategies

In this subsection, we investigate social bot activity based on their political leaning. We explore their strategies in interacting with humans and the degree of embeddedness in the social network.

Table 5.6 depicts the number (and percentage) of tweets generated by each group. Despite the group composed of conservative bots is the smallest in terms of number of accounts (see Table 5.4), it produced more tweets than liberal bots and closely approaches the number of tweets generated by the human counterpart. By computing the number of tweets per user, we note that conservative bots produce 7.4 tweets per account, which is more than twice the ratio related to the liberal bots (3.5), almost the double of the human counterpart (3.9), and almost three times the ratio of liberal humans (2.5). The fact that bots produce a larger number of tweets with respect to humans is not surprising, as confirms bots' attitude to infiltrate and flood online platforms with their content [86]. However, the larger volume of tweets shared by conservative bots, with respect to the volume of tweets shared by liberal bots, suggests a diverse broadcasting strategy adopted by the two groups (conservative vs. liberals) of automated accounts.

To further investigate the different strategies of liberal and conservative bots, we examine their interplay with humans by considering the previously described retweet network. Figure 5.10 shows the interaction among the four groups. We maintain the same color mapping described before, with a darker color (on the bottom) representing bots and a lighter color (on top) indicating humans. Node size is proportional



(a) Overall interactions

(b) Group-based interactions

Figure 5.10: Interactions according to political ideology

to the percentage of accounts in each group, while edge size is proportional to the percentage of interactions between each group. In Figure 5.10a, this percentage is computed considering all the interactions in the retweet network, while in Figure 5.10b we consider each group separately. Therefore, the edge size gives a measure of the group attitude to interact with the other groups. Consistently with Figure 5.9, we observe that there is a limited amount of interaction between the two political sides. The majority of interactions are either intra-group or between groups of the same political leaning. From Figure 5.10b, we can observe that the two bot factions (i.e., conservative vs. liberal leaning) adopted different strategies. Conservative bots balanced their interactions by retweeting group members 43% of the time, and the human counterpart 52% of the time. On the other hand, liberal bots mainly retweeted liberal humans (71% of the time) and limited the intra-group interactions to 22% of their retweet activity. Interestingly, conservative humans interacted with the conservative bots (28% of the time) much more than the liberal counterpart (16%) with the liberal bots. To better understand these results and to measure the extent of human engagement with bots, in the next subsection we evaluate the four metrics introduced in Section 5.3.1.3.

Finally, we examine the degree of embeddedness (defined in Section 2.7) of both humans and bots within the retweet network. For this purpose, we first compute different network centrality measures, and then we adopt the k-core decomposition technique to identify the most central nodes in the graph. In Tables 5.7 and 5.8, we show the average out- and in-degree centrality for each group of users. As we described

Table 5.7: Out-degree centrality

	Liberal	Conservative
Humans	$2.66 \cdot 10^{-6}$	$4.14 \cdot 10^{-6}$
Bots	$3.70 \cdot 10^{-6}$	$7.81 \cdot 10^{-6}$

Table 5.8: In-degree centrality

	Liberal	Conservative
Humans	$2.52 \cdot 10^{-6}$	$4.24 \cdot 10^{-6}$
Bots	$2.53 \cdot 10^{-6}$	$6.22 \cdot 10^{-6}$

in Section 2.7, the out-degree centrality measures the number of outgoing links, while the in-degree centrality considers the number of incoming links. Overall, conservative groups have higher centrality measures than the liberal ones. We can notice that conservative bots achieve the highest values both for the out- and in-degree centrality.

To further investigate bots' embeddedness in the social network, we use the k-core decomposition (described in Section 2.7). We recall that the objective of this technique is to determine the set of nodes deeply embedded in a graph. In fact, the k-core is a subgraph of the original graph in which every node has a degree equal to or greater than a given value k. We extracted the k-cores from the retweet network by varying k in the range between 0 and 30. Figure 5.11 depicts the percentage of liberal and conservative users as a function of k. We can notice that, as k grows, the fraction of conservative bots increases, while the percentage of liberal bots remains almost stationary. On the human side, the liberal fraction drops with k, whereas the conservative percentage remains approximately steady. Overall, conservative bots sit in a more central position in the social network and are more deeply connected if compared to the liberal counterpart.

5.3.2.3 Bots' Effectiveness in Human Engagement

In this subsection, we aim to estimate the effectiveness of bots' strategies in involving humans and measure to what extent humans rely upon, and interact with the content generated by social bots. We examine the effect of bots' activity by means of the four metrics described in Section 5.3.1.3. We evaluate each political side separately, thus, we compare the interaction between bots and humans with the same leaning. In

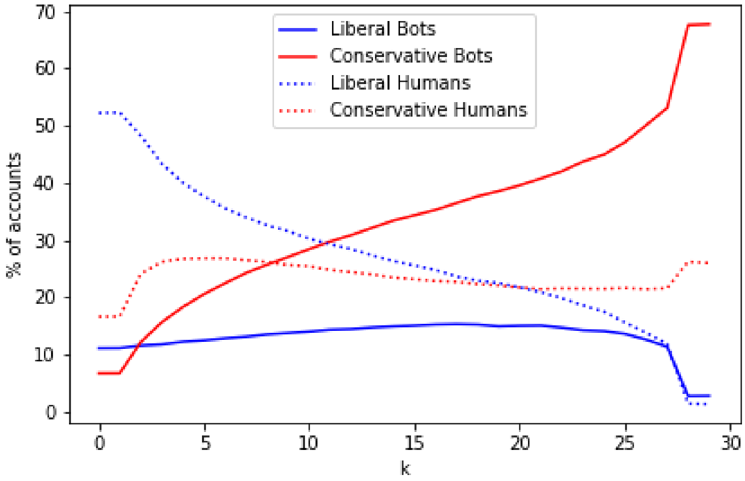


Figure 5.11: k-core decomposition

Table 5.9: Bots' Strategy Effectiveness

Metric	Liberal Bots	Conservative Bots
RTP	14.1%	25.6%
RR	4.5%	15.5%
$H2BR$	12.3%	23.2%
TSR	35.3%	35.0%

Table 5.9, we depict the results for each group of bots. Diverse aspects are worthy of consideration. We can observe that conservative bots are significantly more effective in involving humans in their conversations than the liberal counterpart. Although the TSRs of the conservative and liberal bots are comparable, the gap between the two groups, with respect to the other metrics, is significant. To carefully interpret this result, it should also be noticed that (i) the TSR is inversely proportional to the number of tweets generated by bots, and (ii) conservative bots tweeted more than the liberal counterpart, as depicted in Table 5.6 and explained in Section 5.3.2.2. Overall, conservative bots received a larger degree of interaction with (and likely trust from) human users. In fact, conservative humans interacted with the bot counterpart almost twice with retweets (RTP), and more than three times with replies (RR) if compared to the liberal group. Finally, the $H2BR$ highlights a remarkable amount of human activities that involve social bots: Almost one in four interactions performed by conservative humans goes towards conservative bots.

5.3.3 Discussion

The identification and exploration of bots' strategy is a fundamental asset to detect manipulation campaigns and fight OSN manipulation. Nowadays, this task has become particularly challenging as bots (based on AI) have been refined to infiltrate online discussion in a synergistic way in order to avoid detection (as described in Section 5.2). To tackle this problem, and respond to RQ 3.2, in this analysis, we have considered to analyze the collective behavior of bots based on their political leaning. In fact, we have shown that social bots are embedded in each political wing and behave in a similar way to their human counterpart (i.e., human users with the same political inclination). We have observed different strategies between conservative and liberal bots. Specifically, conservative bots stand in a more central position in the social network and abide by the topics discussed by the human counterpart more than the liberal bots, which in turn exhibit an inflammatory attitude. Further, conservative bots balanced their interaction with humans and bots of the conservative wing, whereas liberal bots focused mainly on the interplay with the human counterpart. Overall, the strategy of the conservative bots appears as the most effective in engaging humans.

5.4 Trolls Detection

Numerous studies showed that bots represent only one factor driving the manipulation issue in OSNs. For example, trolls are largely known for their purpose of sowing conflict and spreading misinformation. In the political context, a case in point is the rise of state-sponsored trolls, human operators tied and allegedly paid by information operation agencies to spread propaganda and politically biased information. The most remarkable example is represented by the "troll farms" associated with Russia's Internet Research Agency, which have been accused of interfering in the political discussion during the 2016 US Presidential election [48]. The list of Russian troll accounts involved in this operation have been released by the US Congress and have permitted to study the malicious activity of such operators [21, 83].

As we described in Section 2.4, it has been shown that trolls' strategies within a campaign change over time and, thus, automatically detecting their activity is not a simple task [244]. To address this challenge and respond to RQ 3.3, in this Section, we propose an approach based on Inverse Reinforcement Learning (IRL). IRL is a machine learning paradigm (detailed in Section 2.6.2.2) that has the goal of finding the rewards behind an agent's observed behavior. In our scenario, we employ IRL to infer the rewards that could have led trolls and non-troll accounts (from now on simply called *users*) to perform their online activity. We then consider to exploit the estimated rewards as features of a supervised learning algorithm aimed at classifying such accounts.

In the next subsections, we describe the data employed in this study and the rationale of our approach. Then, we detail the methodology employed to detect troll accounts and corresponding results.

5.4.1 Data

To attain a set of trolls that serves as ground truth, we rely on the list of 2,752 Twitter accounts identified as Russian trolls by the US Congress and publicly released[6] during the investigation of Russian interference in the 2016 Presidential election. To recover trolls' tweets, we leverage the dataset collected by the research community [5, 19]. The authors [5, 19] utilized Crimson Hexagon, a platform that provides paid datastream access. This allowed the authors to obtain tweets generated by trolls and subsequently deleted after their suspension from Twitter [5, 19]. The dataset presents 1,148 Russian trolls, which generated 1,226,155 tweets.

Our dataset also includes non-troll users' tweets, which have been collected by the researchers [5, 19] by utilizing a set of keywords, listed in Section 5.2.1.1, related to the 2016 US Presidential election event. Also, to capture users' baseline behavior (not strictly related to the political context), the authors gathered tweets from the same users that do not include the political-based keywords. This collection yielded 12,361,285 tweets produced by 1,166,760 users.

We pre-processed this dataset to filter out accounts engaged with just a few tweets. In particular, we consider only users and trolls that shared at least $k = 10$ posts and were involved in at least $k = 10$ other accounts' posts (retweet, reply, or mention). This allowed us to build, for each account, a time-ordered sequence of tweets (of at least 20 elements) in which the account is involved both actively and passively. Such filtering

[6] *Recode's Twitter's list of 2,752 Russian trolls*. See: https://www.recode.net/2017/11/2/16598312/russia-twitter-trump-twitter-deactivated-handle-list

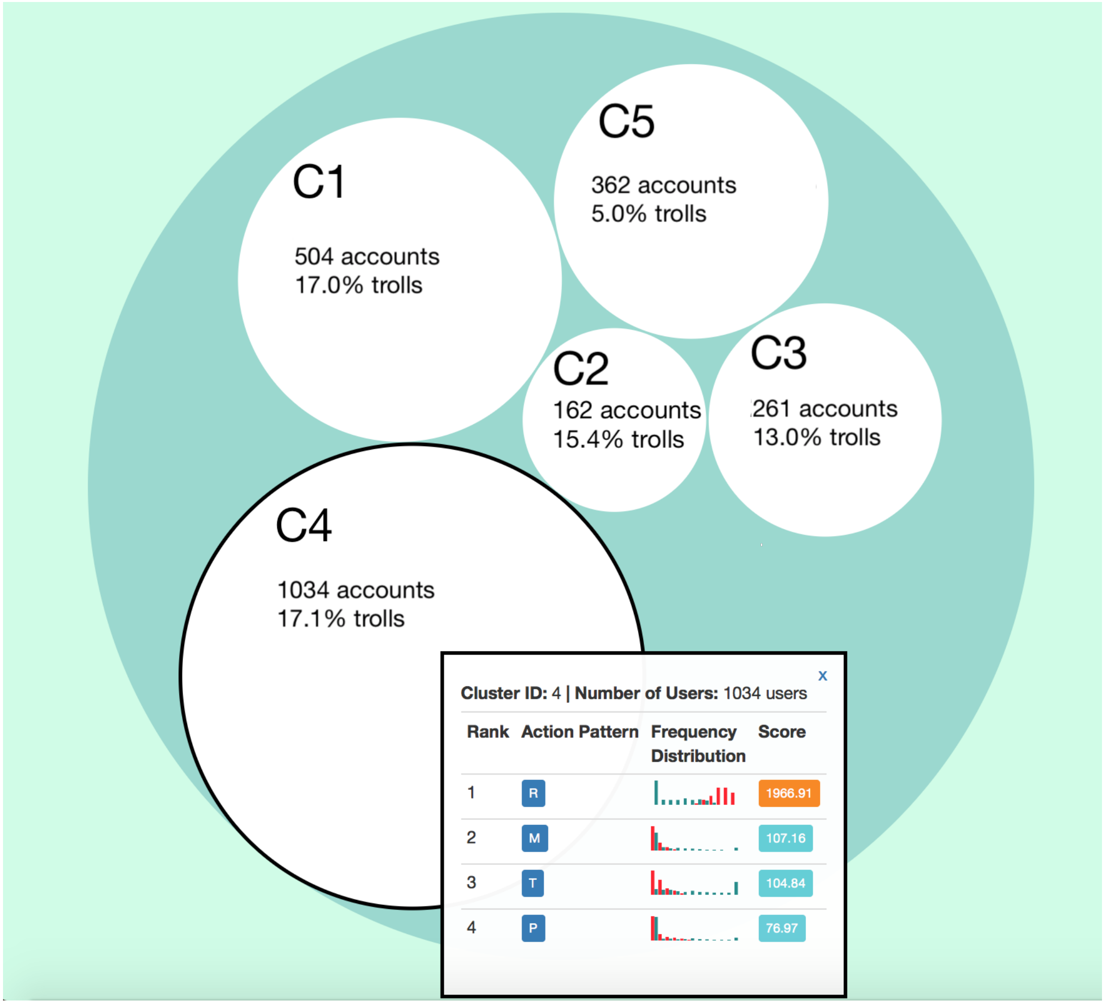
resulted in 342 trolls and 1,981 users. In the next subsections, we describe how we aim to analyze the sequence of users' online activity to characterize their behavior and recognize troll accounts.

5.4.2 User Behavior Analysis

Several studies try to provide an understanding of human behavior through their online activity [67, 151, 152, 235]. In [235], the analysis of users' clickstream on social media is used to identify common behaviors. The authors propose an unsupervised method to categorize behavioral patterns and cluster users accordingly. The *clickstream clustering* [235] is based on a hierarchical clustering approach and it has been proven to outperform existing clustering methods (e.g., K-means) in user behavior analysis. Also, it has shown that such a behavioral model can help service providers to identify unexpected user behaviors as malicious accounts and hostile chatters.

Therefore, with the objective of characterizing and detecting malicious trolls' activity, we attempt to model the online behavior of social media accounts by leveraging the clickstream clustering approach. The input of the clustering approach is represented by the sequence of online actions performed by every account. We determine these activities according to the sharing options available on Twitter, i.e., tweet, retweet, reply, and mention. The output of this approach is a tree hierarchy of behavioral clusters populated by the accounts under investigation.

In our scenario, the *clickstream clustering* [235] results in 5 disjointed clusters, which can be appreciated in Fig. 5.12. It should be noticed that such clusters do not present any hierarchical pattern among them, suggesting that such groups are strongly distinguished. Figure 5.12, which is generated with the visualization tool provided by Wang *et al.* [235], also depicts information about every single cluster. For example, information about cluster 4 (C4) is shown in the table superimposed in Fig. 5.12. In particular, the column *Action Pattern* characterizes the online behavior of the accounts in the cluster. Each row contains one Action Pattern. The Action Patterns are ranked by their discriminating power in classifying the accounts in the cluster under investigation. This means that an Action Pattern with a high rank is important in classifying accounts in this cluster. In the example in Fig. 5.12, retweet (R) is the most discriminating action, followed by mention (M), tweet (T), and reply (P). The column *Frequency Distribution* shows how frequently the given *Action Pattern* appears in the accounts in this cluster if compared to accounts outside the cluster. In other words, it



Cluster ID: 4 | **Number of Users:** 1034 users

Rank	Action Pattern	Frequency Distribution	Score
1	R		1966.91
2	M		107.16
3	T		104.84
4	P		76.97

Figure 5.12: Clickstream clustering [235] results

shows how accounts in the cluster are different from accounts outside of the cluster in a specific *Action Pattern*. The red bars show the pattern frequency distribution for accounts within the selected cluster, while the green bars show the pattern frequency outside the cluster. The more different the two distributions are, the more useful the Action Pattern is to characterize accounts in the cluster. Finally, the column *Score* shows the Chi-Square score of the Action Pattern. The clickstream model relies on this score to rank Action Patterns and perform the clustering.

By further inspecting each cluster, we aim to understand whether and how trolls can be distinguished from users. The 5 clusters (named C1, C2, C3, C4, and C5) and corresponding Action Pattern differ from each other as follows:

C1: composed of 504 accounts (17.0% trolls). Accounts in this cluster perform *reply* and *retweet* less frequently than the accounts outside the cluster.

C2: composed of 162 accounts (15.4% trolls). Accounts in this cluster *tweet* more frequently and *retweet* less frequently than the accounts outside the cluster.

C3: composed of 261 accounts (13.8% trolls). Accounts in this cluster perform *mention* more frequently than the accounts outside the cluster.

C4: composed of 1034 accounts (17.1% trolls). Accounts in this cluster *retweet* more frequently than the accounts outside the cluster.

C5: composed of 362 accounts (5% trolls). Accounts in this cluster perform *reply* more frequently than the accounts outside the cluster.

From the above clustering outcome, it appears that trolls behave similarly to other users. In fact, troll accounts are embedded in every of the above clusters with a comparable percentage (except from C5), which makes them indistinguishable from users.

This finding motivates us to explore other approaches for unveiling such malicious accounts. Although online activities represent users' behavior in the OSN, their analysis takes only into account individual behavior. In such a way, we do not consider whether and how users are influenced by the interaction with others [10, 165] and, more specifically, by the feedback they receive from their peer [65, 67]. The effects of social feedback on online activities have been studied in [67], where the authors argue how the potential endorsement from other users may trigger or retrain online activities. For example, a user might be more motivated to share new content if she/he receives positive feedback. In a similar way, we aim to understand the driving forces behind the online activities of trolls and users with the purpose of discovering whether behavioral differences can arise between these two classes of accounts. The objective of inferring intent and motivation form observed behavior has been extensively studied under the framework of Inverse Reinforcement Learning (IRL) [202]. Therefore, in this study, we rely on the IRL paradigm, whose formulation is detailed in the next subsection.

5.4.3 Methodology

In this subsection, we discuss how we leverage IRL to characterize the online activity of OSN users. We first define the environment of the Markov Decision Process (MDP) we aim to frame, i.e., Twitter. We then present the IRL formulation to estimate users' and trolls' motivation, which are subsequently used for the detection purpose.

5.4.3.1 Twitter MDP

We here propose to model the social media Twitter with a MDP. More specifically, we represent Twitter as an environment constituted of multiple agents (i.e., Twitter accounts) that interact with each other to achieve a certain goal (e.g., to spread content, increase their popularity, or influence other people). The interaction between every agent and the environment follows the RL schema displayed in Fig. 2.3 and described in Section 2.6.2.2: The agent performs a certain action (e.g., share a content) and receive a feedback from the environment (e.g., the content is re-shared by other accounts). In the Twitter environment, we consider that the following four actions can be performed by the agents:

- Generate original content. We refer to this action as *active tweet* (*tw*).
- Re-share content generated by others. We refer to this action as *active retweet* (*rt*).
- Interact with other users by means of reply or mention. We refer to this action as *active reply* (*rp*).
- Keep silent. We refer to this action as *active nothing* (*nt*).

The Twitter environment, in turn, responds to such actions and presents to the agent a new state. We represent the set of states with the three following feedback the environment can provide to the agent:

- Re-share agent's tweet. We refer to this state as *passive retweet* (*RT*).
- Interact with the agent by means of reply or mention. We refer to this state as *passive reply* (*RP*).
- Do not engage with the agent. We refer to this state as *passive nothing* (*NT*).

We leverage the history of the accounts on Twitter to analyze their interaction with the environment. To this aim, we sort in chronological order every online activity in which the account is involved, either actively (actions) and passively (states). Therefore, at each step (i.e., an element of the ordered sequence of the account's online activities) the agent can be in one of the above-mentioned states and perform only one action. We consider the agent to execute action *nt* (active nothing) if it does not react to the environment feedback, e.g., it does not perform *tw*, *rt*, and *rp*. Similarly, we represent with *NT* the case in which the environment does not react to the agent's action.

5.4.3.2 Users and Trolls IRL

As mentioned in Section 2.6.2.2, the state-action space is usually represented by a set of features f. Similarly to [204], we build f by concatenating variables related to the set of states and actions. Specifically, we define f as the possible combination of the following variables: (*RT*, *RP*, *tw*, *rt*, *rp*), where the first two features represent the state space, while the last three features indicate the action space. Each feature is a binary variable, which assumes value 1 based on the state of the agent and the action it performs (e.g., if the agent is in the state *RT* the first feature assumes value 1, while the second feature equals to 0). The state *NT* is represented by setting the first two features to zero, while the last three features equal to zero indicates the action *nt*. As an example, the tuple $(0,0,0,0,1)$ indicates that the agent performed an active reply (*rp*) while it was in state NT. Given that at each step the agent can be in only one state and performs only one action there exists the following 12 possible combinations of state-action pairs:

- (*NT*,*nt*): the environment does not provide any feedback, and the agent does not perform any action;

- (NT,tw): the environment does not provide any feedback, and the agent generates a tweet;
- (NT,rt): the environment does not provide any feedback, and the agent generates a retweet;
- (NT,rp): the environment does not provide any feedback, and the agent generates a reply (or mention);
- (RT,nt): the environment returns a retweet, and the agent does not perform any action;
- (RT,tw): the environment returns a retweet, and the agent generates a tweet;
- (RT,rt): the environment returns a retweet, and the agent generates a retweet;
- (RT,rp): the environment returns a retweet, and the agent generates a reply (or mention);
- (RP,nt): the environment returns a reply (or mention), and the agent does not perform any action;
- (RP,tw): the environment returns a reply (or mention), and the agent generates a tweet;
- (RP,rt): the environment returns a reply (or mention), and the agent generates a retweet;
- (RP,rp): the environment returns a reply (or mention), and the agent generates a reply (or mention).

Therefore, f is a 5×12 matrix, where 5 is the number of features and 12 is the number of state-action pair possible combinations. In turn, the reward function is a 1×12 vector, where each element represents the scalar reward of a certain state-action pair.

Based on the ordered list of states and actions, we build for each account a trajectory ζ composed of the feature representation of state-action pairs, as discussed above. Such a trajectory represents the observed behavior of the agent. We then utilize IRL approaches to estimate the reward function that drove the agent's behavior. The rationale is to investigate whether users and trolls share the same rewards or are guided by different incentives. For this purpose, we employ two IRL approaches introduced

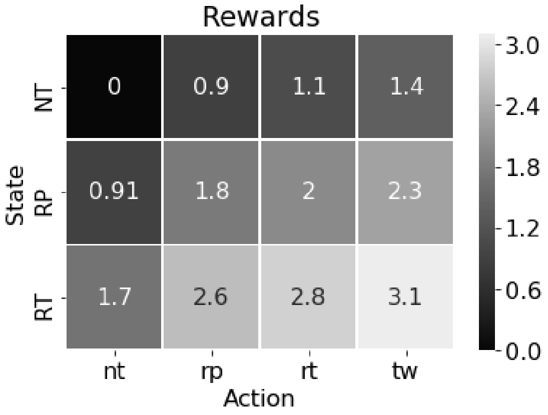


Figure 5.13: Example of estimated rewards for a given account

in Section 2.6.2.2. In particular, we employ the Maximum Entropy IRL [251] and its non-linear variation proposed in [240]. It should be noticed that every account is modeled independently from the others. This means that a unique IRL model has been developed for each account. Each model takes as inputs the feature matrix f, the trajectory ζ, and the transition probabilities T. As discussed in Section 2.6.2.1, the latter represents the probability of transition from state $s \in S$ to state $s' \in S$ after performing action $a \in A$. We compute T by observing the occurrences of (s, a, s') triplets in the account's trajectory, where the triplet represents the transition from s to s' when a is performed. Both the IRL approaches use these inputs to compute the reward function R, which determines the scalar reward for each state-action pair. In the next subsection, we explain how we employ the estimated rewards for detecting trolls activity.

5.4.3.3 Trolls Detection

For every account, IRL approaches output 12 scalar values related to the state-action combination described above. As an example, Fig. 5.13 depicts the estimated rewards related to the online activity of a generic user. Each value represents the reward that could have led the account to perform a given action in a certain state. A high reward in the generic state-action pair (s, a) indicates that the user is very motivated to perform action a in state s. The opposite holds for low reward values.

Our approach aims to understand whether some differences can be noticed in the motivation and incentives between trolls and users. We hypothesize that reward values might highlight distinctive behavioral characteristics between troll and user

Table 5.10: IRL approaches performance comparison in terms of AUC

	Max. Entropy IRL [251]	Max. Entropy Deep IRL [240]
K-Neighbors	83.2%	82.4%
SVC	74.2%	85.4 %
Gaussian Process	83.8%	**85.6**%
Decision Tree	82.7%	74.1%
MLP	84.4%	79.8%
AdaBoost	**89.1%**	83.3%
Random Forest	86.7%	81.3%
Naive Bayes	79.3%	78.7%

accounts. Therefore, we propose to utilize the 12 reward values to discriminate these two classes of accounts. More specifically, we utilize the estimated rewards as features of a supervised learning algorithm aimed at detecting troll accounts. We frame this problem as a classification task, where the two classes are *troll* (positive class) and *user* (negative class). We test several machine learning algorithms to perform such classification, whose results are discussed in the next subsection.

5.4.4 Evaluation

In this subsection, we first discuss the results of the IRL-based classification approach for detecting troll accounts on social media. Then, we analyze the incentives that users and trolls respond to and we highlight the behavioral differences between these two classes of accounts.

5.4.4.1 Account Classification

The dataset under investigation presents imbalanced classes, with the negative class (more than) 5 times larger than the positive one (1,981 vs. 342 accounts). To solve the data imbalanced issue, we employ the undersampling technique [74, 161], which uses only a random set of the larger class in the classification task. Therefore, we split the negative samples into five parts and we repeat our evaluation every time with a different subset of negative samples. Then, to train and test our model, we utilize a 10-fold cross validation preserving the percentage of samples for each class.

We evaluate the classification (troll vs. user) results obtained by using the two different IRL approaches to estimate the rewards. As mentioned in Section 5.4.3.2, we test the Maximum Entropy IRL [251] and the Maximum Entropy Deep IRL [240]. We utilize

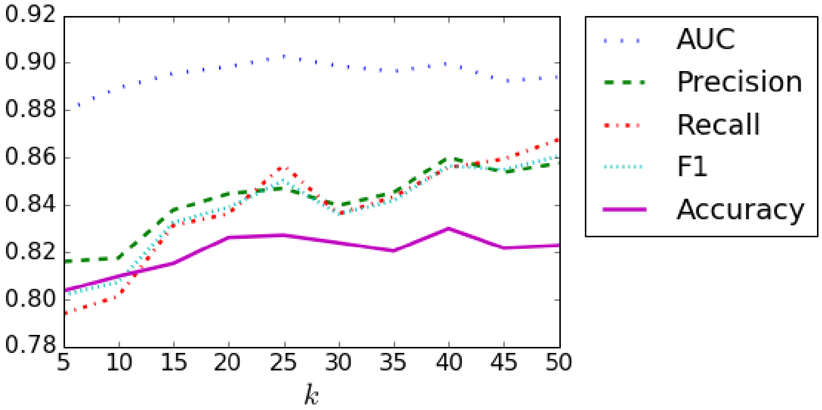


Figure 5.14: Classification performance at varying k

the rewards inferred by each of these approaches to feed a supervised learning classification algorithm. In particular, we compare several machine learning approaches. Performance, in terms of Area Under the Curve (AUC), of the different classifiers and IRL approaches are depicted in Table 5.10. In most of the cases, the Maximum Entropy IRL approach achieves better accuracy with respect to the Deep IRL solution. Thus, we continue our analysis by leveraging the rewards estimated with Maximum Entropy IRL. Also, we rely on AdaBoost as it outperforms the other supervised learning algorithms. We tune the AdaBoost classifier by performing a grid search of its hyper-parameters, whose best configuration involves 500 weak estimators and a learning rate of 0.05.

We further evaluate the performance of our approach by considering classification metrics, other than AUC, such as *accuracy*, *precision*, *recall*, and *F1*. In Figure 5.14, we display the results of the proposed solution at varying k. As mentioned in Section 5.4.1, we filtered out accounts that were involved in less than $k = 10$ active and passive online activities. Here, we aim to examine the impact of k on classification performance. As we could expect, performance improves as k increases as the Maximum Entropy IRL approach can rely on more information per each user. Overall, our detection approach, even with a small amount of information, achieves prominent performance in every classification metric. This validates our intuition of using IRL to analyze online behavior and to identify malicious troll accounts accordingly.

To investigate the most distinguishing characteristics between trolls' and users' behavior, we observe the feature importance of the AdaBoost algorithm. In Fig. 5.15, the

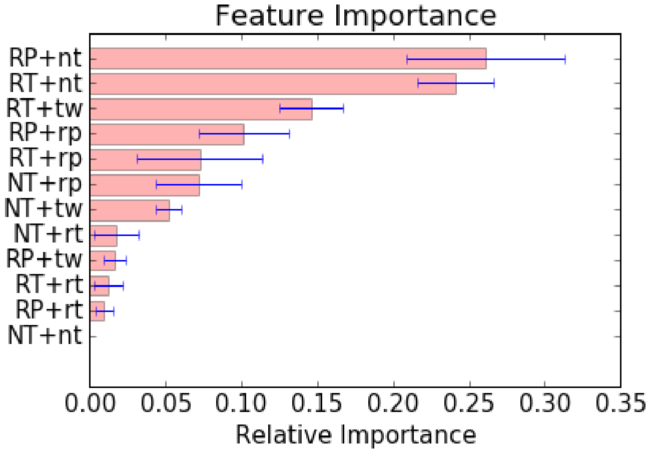


Figure 5.15: AdaBoost feature importance

relative importance of each feature (i.e., reward) is displayed. Interestingly, the most important features are related to the case when the account is in the state *RT* (i.e., its content has been re-shared by other accounts) or *RP* (i.e., it is engaged by other users).

5.4.5 Rewards Analysis

To better understand the difference in trolls' and users' intent, we compare their estimated rewards. Figure 5.16 displays the distribution of the rewards assessed with Maximum Entropy IRL and highlights some differences in the behavior of the two classes of accounts[7]. This discrepancy is also demonstrated by a two-sample Kolmogorov-Smirnov test, which in turn unveils significant differences ($p < 0.01$) between every pair of distributions, except for the state-action pair (*NT*,*nt*), (*NT*,*rt*), and (*RP*,*rt*). By looking at Fig. 5.15, we can notice how these three rewards are in the 5 least important features for the AdaBoost classifier. From Fig. 5.16, it is possible to notice a flat distribution for the pair (*NT*,*nt*). This is due to the fact that we do not observe any instance of such a scenario in users' trajectory, given that we can only leverage data that show users' involvement (either active or passive) in the tweets.

Furthermore, in Fig. 5.16, it can be appreciated how, on average, trolls are more motivated (i.e., higher reward) to share a new original tweet (action *tw*) with respect to users regardless of their state, which suggests that trolls merely focus on spreading

[7] It should be noticed that the displayed rewards have been normalized by using the StandardScaler technique (i.e., each feature is standardized by removing the mean and scaling to unit variance).

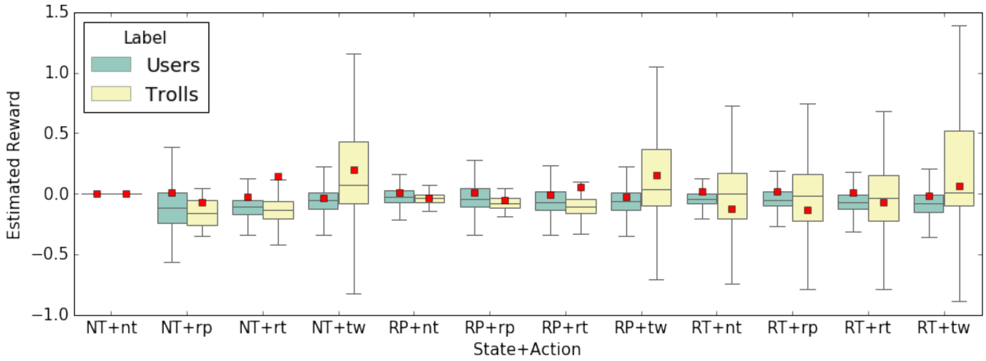


Figure 5.16: Distribution of the estimated rewards by Maximum Entropy IRL.

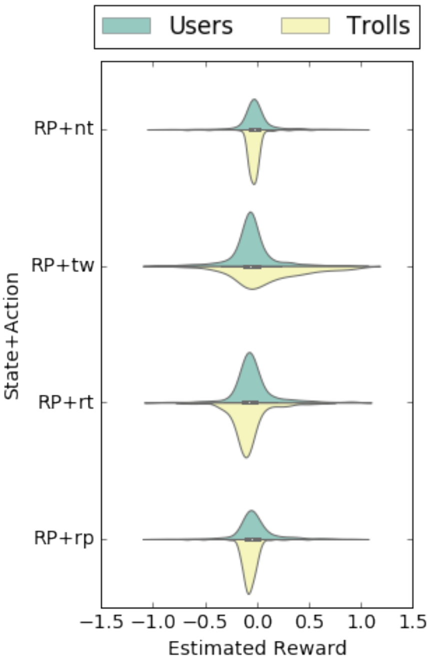


Figure 5.17: Violin plot of the estimated rewards in the *RP* state.

their content, independently from others' feedback. Also, the most noticeable discrepancy can be observed in the actions performed when the account is in the *RT* state, which is in line with the feature importance results displayed in Fig. 5.15.

The latter also showed that the *RP* state is relevant in the discrimination between trolls and users. To take a closer look at the features related to this state, in Fig. 5.17, we depict the violin plot distribution of the estimated rewards related to *RP* state. A violin plot is a method to visualize the distribution of numerical data and its probability

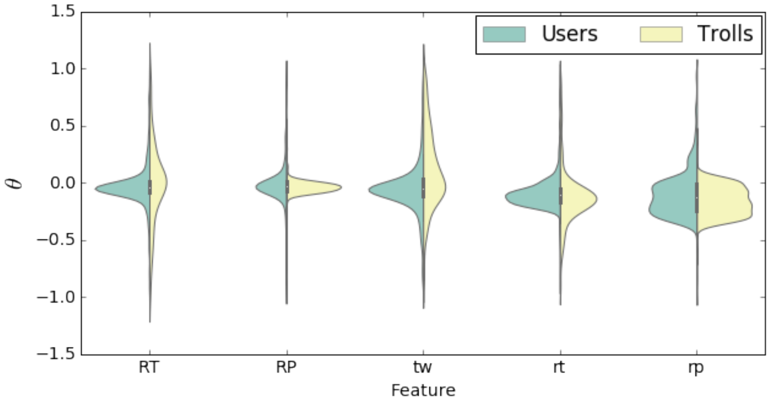


Figure 5.18: Distribution of weights θ for each feature in f

density. From Fig. 5.17, it can be observed how the reward distributions of troll and user accounts particularly differ in the case of action *nt* and *rp*, which are the first and forth most relevant features of the AdaBoost classifier.

Moreover, being a linear model, the Maximum entropy IRL approach allows to disentangle the reward of each state-action pair. This is done by solving Eq. 2.3, given that f is known and R has been estimated with Maximum entropy IRL. This operation allows us to recover the weights θ that represent a measure of importance for each feature in f. The weights distribution for each class of accounts is displayed in Fig. 5.18. Also in this scenario, we utilize a violin plot (defined in the description of Fig. 5.17), to visualize and compare the distribution of the weights for the two classes (i.e., trolls and users). The violin plot distribution further highlights behavioral differences between trolls and users, which are proved to be significant for every feature ($p < 0.01$) by means of a two-sample Kolmogorov-Smirnov test. Additionally, Fig. 5.18 confirms the differences of behavior between trolls and users when they are in state *RT* and *RP*, consistently with our previous findings. Also, the action that shows the most noticeable difference between the two groups of accounts is related to the active tweet (*tw*).

Finally, in Fig. 5.19, we depict the mean value of such distributions. The purpose is to compare the relevance of each feature between troll and user accounts. On average, trolls appear to be more motivated than users in generating new content (*tw*) and re-sharing others' post (*rt*). This finding is in line with trolls' purpose of spreading (false) pieces of information and harm the online conversation. On the other hand,

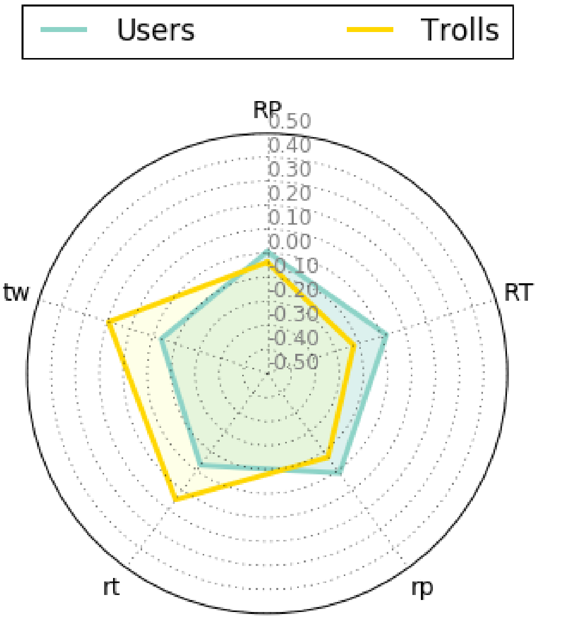


Figure 5.19: Comparison of θ average values of trolls and users

users appear to be more rewarded than trolls when their content is re-shared by other accounts (*RT*). This, representing a form of social endorsement [173, 220], might suggest that users are more concerned about others' esteem with respect to trolls.

5.5 Conclusion

In this Chapter, we examine the issue of OSN manipulation. We focus on the malicious actors, e.g., trolls and bots, that embed themselves in online social systems to interact with OSN users with the objective of deceiving them and manipulating the narratives they are exposed to. Although OSN service providers put increasing efforts to protect their platforms, malicious accounts keep acting in manipulation campaigns. While bots continuously adapt their strategy to escape detection, the automated identification of troll accounts has not found any established solution yet. In this Chapter, we tackle these challenges to empower and enable countermeasures for fighting the manipulation of OSNs.

To keep the pace of increasingly sophisticated bot accounts, we study how bots' online activity has changed over the last two US voting events, i.e., the 2016 Presidential election and the 2018 Midterms. To answer RQ 3.1, we unveil bots' strategy to avoid detection and mimic human accounts. We show how bots tuned their sharing activity to emulate human timing and adopted a (more cautious) synergistic approach to spread information, which allowed them to simultaneously avoid detection and create an illusion of public consensus.

To address RQ 3.2, we analyze the collective behavior of bots based on their political leaning. We observe that bots were embedded in each political wing and acted similarly to the human counterpart. We uncover different bots' strategies for engaging with humans and recognize the most effective. Overall, these findings (especially those concerning bots' coordinated strategy to avoid detection and manipulate humans) provide useful insights that can inform actionable policies to empower the detection of manipulation campaigns. However, our results show the multifaceted behavior of bots, their effectiveness, and mutable nature over time. These facts raise further concerns and pose novel challenges in the fight against OSN manipulation.

This fight, however, does not only regards automated accounts (i.e., bots) but also include human users (i.e., trolls) often associated with and paid by government agencies and/or foreign entities. As the detection of trolls is an open problem for the research community, in this Chapter, to respond to RQ 3.3, we propose an approach based on IRL that only relies on the flow of users' online activity. Our approach accurately (AUC=89.1%) separates trolls from other users by leveraging and exploiting the diverse motivations between the two classes of accounts. The IRL model also allows us to recognize and unveil the most distinctive behaviors that differentiate troll from non-troll accounts. For example, trolls and users differ in their behavior when they are engaged by other users or their content is re-shared. Also, troll accounts appear to perform their sharing activity irrespective of others' feedback and simply focus on the spread of the content they generate. This work represents a first step in the direction of understanding, characterizing, and detecting trolls, which is a critical challenge in the race towards healthy OSNs. In our future work, we aim to validate the proposed approach with data from other known state-sponsored trolls. Also, we aspire to extend the classification task to a diverse set of malicious accounts (e.g., bots).

6
Awareness of OSN Perils

6.1 Introduction

In the previous chapters, we explored the perils of OSNs along the lines of privacy and manipulation issues. We have demonstrated that such risks are concrete and take advantage of human vulnerabilities and OSN functionalities. However, although these problems can have significant consequences on people's lives, the majority of OSN users is still not conscious of (or underestimate) such pitfalls. Additionally, users have also difficulty to react to such issues, as they need timely, clear, and easy to understand information upon which to base their choices. Raising users' awareness of the perils of OSNs is, therefore, of pivotal importance. For this purpose, we appraise the idea of building a service to provide OSN users with feedback about their current risks in real-time. As smartphones represent the main means to utilize OSNs [128], we envision this service to be offered via a mobile application. However, as described in Chapters 1 and 2, guaranteeing this kind of service introduces additional challenges, such as privacy and security risks. To tackle these challenges, in this Chapter, we aim to answer the following RQ:

RQ 4: *Can we develop a secure and privacy-preserving service for assessing and communicating users' their privacy and manipulation risks in OSNs?*

To respond to RQ 4 and face the related challenges, we present a framework, called VIVO, which allows us to collect data from mobile devices in a secure and privacy-preserving way [166]. VIVO also enables real-time data gathering and direct communication with users via mobile applications. Although VIVO has been conceived to support a broad range of applications, ranging from crowd-sensing data collection to application development and testing, in this thesis, we exploit VIVO as it allows to address RQ 4 for three main reasons:

- It provides a privacy-by-design facility for application developers who are willing to collect data and/or test their applications. By using VIVO, developers do not have to deal directly with such privacy and security aspects as both of them are managed within the framework.
- It supplies an environment for developing an application that collects data from smartphones in real-time. This also allows to enlarge the spectrum of possible utilization, by considering not only OSNs as data source but any sensing application running on a smartphone. As a result, we can validate and extend the models based on information extracted from OSNs (e.g., the privacy model based on Twitter data) to data collected by mobile devices. For instance, it is possible to assess the accuracy of location inference attacks also when users do not share their position in OSNs.
- It allows to directly communicate with the users, thus, enabling a mechanism of direct feedback of the risks behind their online activity (e.g., privacy leakage in an OSN) in real-time.

Moreover, in terms of privacy, VIVO addresses issues at two levels: The first one, with VIVO architecture development, aims to build a privacy-preserving framework for application developers, as we mentioned above. The second one is privacy at the application level. We develop an application running onto VIVO that aims to collect users' data and return them feedback about their current risks. We envision this application to be used for two main purposes: (i) provide the users a measure of their level of privacy (e.g., geo-location privacy), and (ii) alert the users every time they interact with malicious actors in OSNs.

This chapter is organized as follows. We introduce VIVO in Section 6.2, while in Section 6.3 we provide an overview of its architecture. Section 6.4 details the building blocks of VIVO architecture. In Section 6.5, we describe the application that has been developed through VIVO to raise users' awareness of their OSN risks. Finally, Section 6.6 is devoted to an evaluation of the system performance in terms of functionality and battery consumption.

6.2 VIVO

Smartphones represent the main source of human digital traces. However, as described in Section 2.5.2, collecting crowd-sensed data is not a simple task as challenges related to device heterogeneity, security, and privacy need to be addressed. To tackle these challenges, we implemented VIVO, an open framework for crowd-sensed Big Data gathering, where security and privacy are managed within the framework at the client side. VIVO allows to test and validate any services that use social, physical, and environmental information. Unlike previous efforts [25, 81, 221], VIVO allows the deployment of multiple simultaneous experiments introducing an enrolled crowd-sensing model. In such a model, VIVO *experiment developers* (i.e., application developers who need to collect data) are not limited to a fixed set of experiments but they can build their own application without any constraint, in a more agnostic and generic way.

The collected data can be accessed both at the end of the experiment, as in traditional testbeds, and in real-time, as required by many big data applications. Yet, VIVO differs from traditional testbeds as testing experiments can be scheduled and run in *real-time* on the mobile phones of users, from now on referred to as *volunteers*. In fact, VIVO allows an easy development and deployment of experimental software on mobile devices. More precisely, experiment developers can dynamically deploy their own application on each VIVO volunteer device, without any extra-hardware requirements and pre-deployment testing as VIVO experimental applications run on standard Android versions (without any extra OS requirements).

One of the key features of VIVO concerns the security and privacy of volunteer data. As we leverage private smartphones, it becomes even more crucial to ensure that any deployed applications do not compromise the private data of the users and the regular behavior of their private applications. To address this challenge, VIVO manages

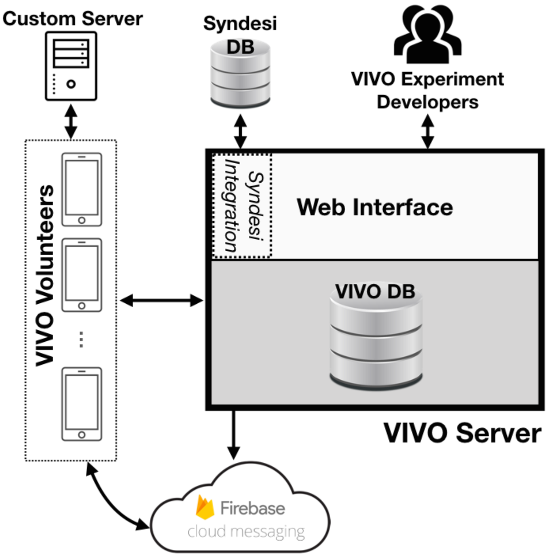


Figure 6.1: VIVO architecture

security and privacy within the framework itself, at the client side. VIVO provides an API with all the methods necessary to secure and privatize the collected data before they leave the smartphone. Clearly, we cannot prevent malicious behaviors, but these are legally prosecutable as a contract violation.

Finally, VIVO allows a paradigm shift from (i) taking care of the whole experiment cycle, i.e., from the experiment design up to the data provision, to (ii) managing only the experiment application, with built-in security and privacy capabilities. In fact, it provides to experiment developers a compact unified framework to collect data, from the architecture (e.g., server, data management, and security) to the mobile sensing nodes, i.e., volunteer smartphones.

6.3 Architecture

VIVO architecture is displayed in Fig. 6.1. At the top level, the VIVO experiment developers, i.e., individuals (e.g., researchers), employ VIVO to run an experiment for collecting a dataset or testing an application. VIVO experiment developers (from now on simply referred to as developers) constitute the target group for whom VIVO has been conceived. They exploit VIVO data storage and data collection capabilities as well as VIVO volunteers and their mobile devices to deploy their applications. Volunteers are people equipped with personal smartphones who accept to participate in VIVO experiments. For each experiment, volunteers can choose whether to participate or not using the VIVO Client application. By means of a *Web Interface*[1], developers have the possibility to define new experiments, upload the source code of the corresponding applications, and download the collected data. Experiments uploaded to the VIVO testbed are checked and validated with regard to respecting privacy and trust issues. Only accepted experiments can be deployed on volunteer devices. The alpha testing is performed during the pre-deployment phase and it checks the impact of the experiment on the overall system performance and on the user's privacy. We utilize Portable Opensource Energy Monitors (POEM) [88] to measure the energy overhead of the application and an extension of the Mockingbird platform [89] to monitor the information leakage. Mockingbird performs an on-device evaluation to retrieve the information accessible from the experiment application, e.g., when and how many times it access the file systems, the sensors, the contacts, etc. This platform produces an access-report that is compared with the experiment description in order to detect access patterns not compliant with the application task.

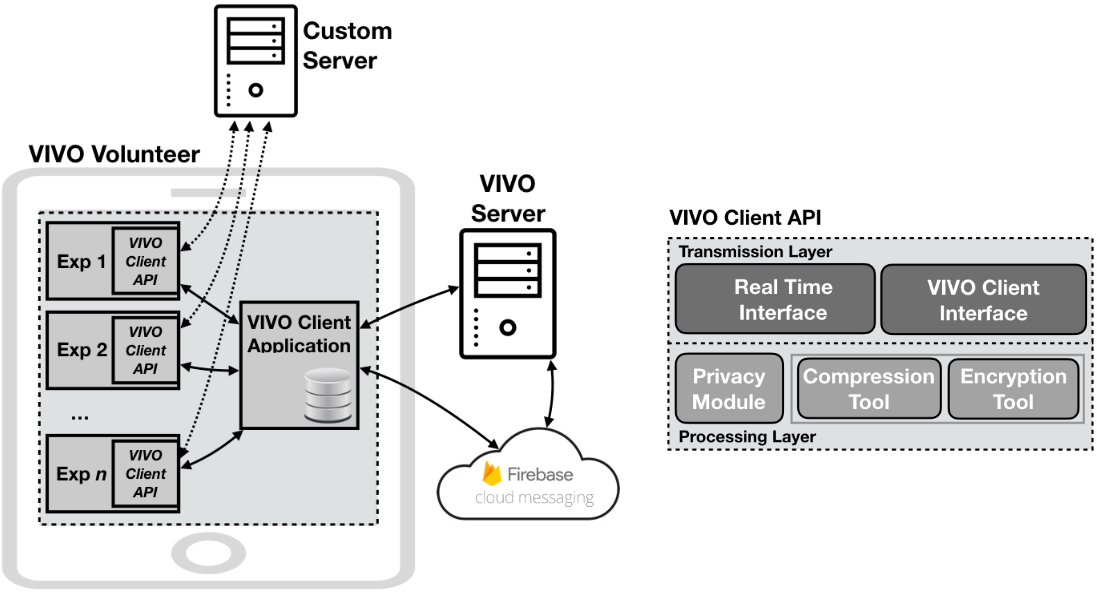
VIVO consists of three main components, which will be discussed in turn, namely VIVO Server, VIVO Client, and VIVO Client API.

1. The *VIVO Server* is the main back-end platform of the architecture. It controls the creation of new experiments, the notification to volunteers, and the experiments data upload. The VIVO Server uses Google Firebase[2] to push notifications to the volunteer devices to notify the availability of new experiments. The data collected from the volunteers are periodically sent to the VIVO Server, which handles the data upload from the devices and their storage on the *VIVO Database (VIVO DB)*. Once an experiment is terminated, the VIVO Server allows the developer to download the collected data through the web page. As mentioned in Section 2.5.2, VIVO Server is enhanced by the functionalities of the *Syndesi* IoT testbed. The integration between VIVO and Syndesi is performed at the web service level. In particular, after terminating an experiment, the developer can download the data collected by the experiment as well as the data collected by Syndesi environmental sensors, within the experiment time window.

2. The *VIVO Client* enables the communication between volunteers and the VIVO Server. Volunteers contributing to the VIVO testbed are required to install

[1] The VIVO Web Interface is reachable at http://vivo.dti.supsi.ch:3000/

[2] https://firebase.google.com/



(a) (b)

Figure 6.2: VIVO Client (a) and VIVO Client API (b)

and run the VIVO Client application on their devices. In particular, by means of the VIVO Client application, each volunteer can register in VIVO and run several experiments on their smartphones. The Client application displays the list of available experiments updated in real-time and allows volunteers to manage the experiment life-cycle in a very straightforward and user-friendly manner (one-click operation). As depicted in Fig. 6.2a, the VIVO Client acts as a middle layer between VIVO experiments and the VIVO Server. It gathers data collected by all the experiments running on the volunteer devices and manages their forwarding to the VIVO Server. All the data handled by the VIVO Client application are compressed and encrypted, as explained in Section 6.4.

3. The *VIVO Client API* is a fundamental component of VIVO and enables the key features of the proposed architecture, such as security, privacy, and real-time data collection. Developers are requested to use the VIVO Client API in their application as a requirement to use VIVO and its features. The API is represented in Fig. 6.2b. In the Processing Layer, it provides all the tools necessary to compress (*Compression Tool*), encrypt (*Encryption Tool*), and privatize (*Privacy Module*) the collected data before the transmission from the volunteer device. As depicted in Fig. 6.2a, each experiment exploits the Transmission Layer of the VIVO Client API to forward the collected data to (i) the VIVO Server via the *VIVO*

Client Interface, or to (ii) a Custom Server (configured by the developer) through the *Real-Time Interface*. The VIVO Client Interface is destined for offline data collection, while the Real-Time Interface, and in turn the Custom Server, enables real-time applications. In both cases, the VIVO API provides the underlying tools to encrypt, compress, and privatize the data.

6.4 Implementation

This Section provides a detailed description of each VIVO component, providing details on the functionalities, implementation choices, and technologies utilized in the system design. Section 6.4.1 describes the *VIVO Client* component, while Section 6.4.2 depicts the features of the *VIVO Client API*. Finally, in Section 6.4.3, we detail the *VIVO Server*.

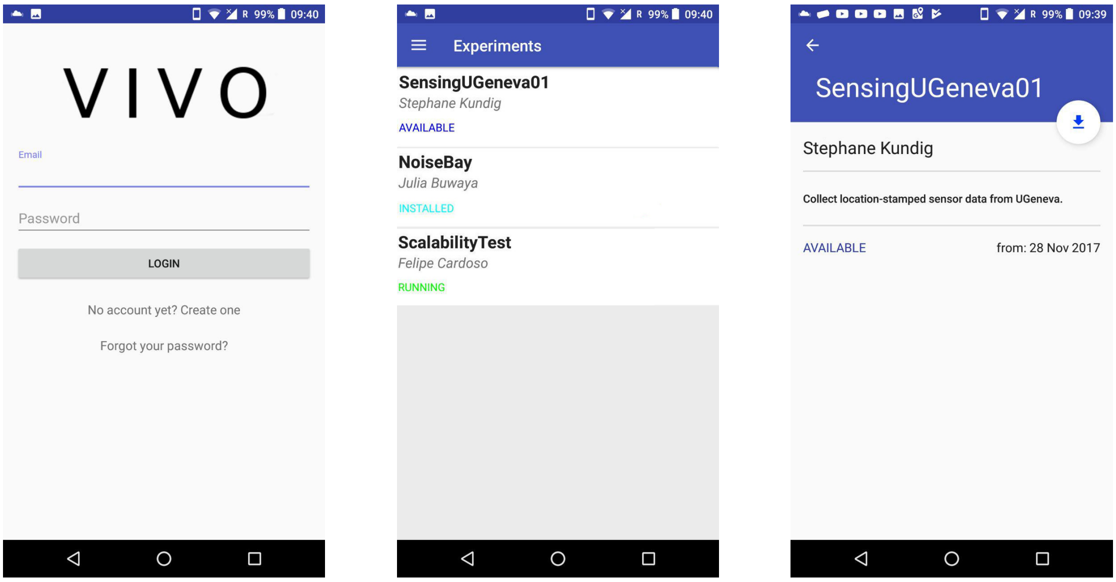
6.4.1 VIVO Client

The *VIVO Client* is an Android application (compatible with OS version 4.2 or above) for volunteers to interact with the VIVO platform. The VIVO Client also provides synchronization of the collected data with the VIVO Server.

6.4.1.1 User Interface (UI)

The VIVO Client UI is the interaction point between the VIVO Server and the volunteers. The UI allows them to monitor the experiments available and running on VIVO, as well as managing their participation in each experiment.

Fig. 6.3 shows three screen shots related to the main foreground components of the UI. After installing the VIVO Client application, volunteers are asked to register or log-in through the Login page (Fig. 6.3a). Once logged in, volunteers can explore all the available experiments from the Experiment List page (Fig. 6.3b). Each entry in this list includes the identification parameters of the corresponding experiment (e.g., name and author), and the experiment status on the volunteer smartphone. An experiment can be: *(i) available* for download and installation on the volunteer devices; *(ii) installed* and ready for launch; *(iii) running* on the volunteer smartphone. Additional information along with a detailed description of each experiment are available on the Experiment Details page (Fig. 6.3c). Through this page volunteers



Collect location-stamped sensor data from UGeneva.

(a) (b) (c)

Figure 6.3: VIVO Client activities: (a) login, (b) experiment list, (c) experiment details

can manage the experiment life-cycle, starting, stopping, and un-installing it any time they want to. The VIVO Client continuously monitors the status of the experiments and displays notifications every time an experiment is created or terminated.

6.4.1.2 VIVO Client Data Collection

The UI has a key function in the interaction between volunteers and VIVO, nonetheless the VIVO Client performs several fundamental tasks in the background. These tasks are related to the *data collection process*. The VIVO Client is in charge of all the processing necessary for sending the experiment data from volunteer smartphone to the VIVO Server. Experiment data are sent to the VIVO Client through the VIVO Client API in the format of *data blocks*. A *data block* is a structure of data containing three fields: *data*, *timestamp*, and *type*. The field *data* contains an encrypted and compressed version of the data (details on data preprocessing will be given in Section 6.4.2.3). The *timestamp* is the time at which the data was collected, while the *type* field is used for differentiating the types of data and is defined by the developer during the experiment development phase. There are no restrictions on the type of data that can be collected (i.e., string, number, custom structure). To receive data from the running experiments, the VIVO Client instantiates a *Service* component named *Data Receiver*. This represents the VIVO Client connection with the VIVO Client API. Every time this component receives a data block from the VIVO Client API, it (i) verifies whether the data sender is an authorized application by checking its package name, (ii) extends the data block by adding a field named *experiment ID*, which identifies the experiment that generated the data, and (iii) temporarily stores the data block into the local database until the synchronization with the VIVO Server is performed.

To perform the synchronization VIVO utilizeS the Android *Sync Adapter* component, which provides a smart way to manage data synchronization and battery consumption. Each time a synchronization is performed, a batch of data is sent to the VIVO Server. In this phase, a field named *device ID* is added to the data block to identify the device that collected the data. Every time the VIVO Server receives the data, it returns an acknowledgement and the data is deleted from the local database of the VIVO Client.

6.4.2 VIVO Client API

The *VIVO Client API* is a software component needed for a VIVO experiment in order to interact with the VIVO Client. An *Experiment Development Tutorial* is available on the VIVO website: it guides developers throughout the integration of existing or new Android applications with the VIVO Client API. The API is the core enabler of the VIVO architecture. Besides the interaction with the VIVO Client for forwarding the collected data, the API provides these additional features: *(i)* an interface for storing data on the VIVO Server; *(ii)* an interface for sending data to a custom server in *real-time*; *(iii)* tools for data *compression and encryption*; *(iv)* a *privacy module* to privatize the data.

The VIVO Client API is an Android Library and has the same minimum OS requirements as the VIVO Client. The API does not interact with mobile sensors and does not require any permission, thus, device heterogeneity does not affect its functionality. Thereby, developers should handle experiment dependencies by ensuring in their code whether volunteer devices meet the given requirements.

6.4.2.1 VIVO API Data Collection

The VIVO API Data Collection is the main function of the API, providing an interface to the VIVO Client for secure data transactions. In fact, this feature benefits from one of the main tools embedded into the API: a module for data compression and encryption. In our context, encryption is necessary because the collected data is exposed to security risks during both the API-Client and the Client-Server transmission. We

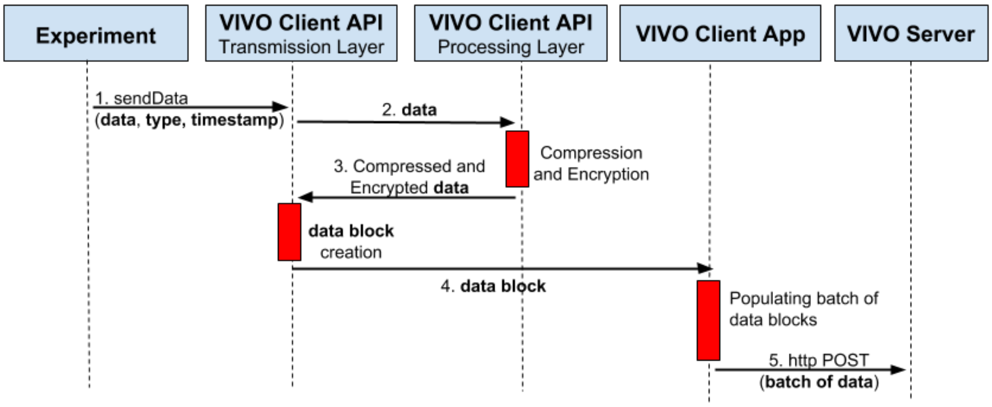

Experiment
VIVO Client API Transmission Layer
VIVO Client API Processing Layer
VIVO Client App
VIVO Server
1. sendData (data, type, timestamp)
2. data
Compression and Encryption
3. Compressed and Encrypted data
data block creation
4. data block
Populating batch of data blocks
5. http POST (batch of data)

Figure 6.4: Sequence diagram of the VIVO API (offline) data collection

address this issue by encrypting the data block locally (within the application), before sending it to the VIVO Client. To accomplish this task, the API makes use of asymmetric cryptography. As this technique can encrypt a limited block of data each time, the API compresses the data before encryption. In such a way, a larger amount of data can be encrypted in a single block. Experiment developers have to create a public-private key pair and configure the API for the usage of the public key.

Fig. 6.4 shows a diagram representing the sequence of actions performed by each component of the system in a data collection scenario, where *(i)* the collected data is compressed and encrypted by means of the Processing Layer of the VIVO Client API, *(ii)* the data is encapsulated in a data block in the Transmission Layer and then forwarded to the VIVO Client application, which *(iii)* accumulates data blocks in a batch of data until the synchronization with the VIVO Server is accomplished.

6.4.2.2 Real-Time Data Collection

Data collected in the VIVO DB is suitable for *offline* data post-processing and analysis. However, this solution does not fit real-time data processing and applications with low latency requirements. As the Sync Adapter framework of the VIVO Client does not assure real-time synchronization, it is not guaranteed that the collected data reaches the VIVO Server with a short delay. To enable real-time data collection and low-latency applications, we propose a simple interface, named Real-Time Interface, for integrating a Custom Server endpoint into the architecture. The developer should only set the Custom Server address in the VIVO API settings to perform HTTP requests with the Real-Time Interface, without configuring any server-side API. Different levels

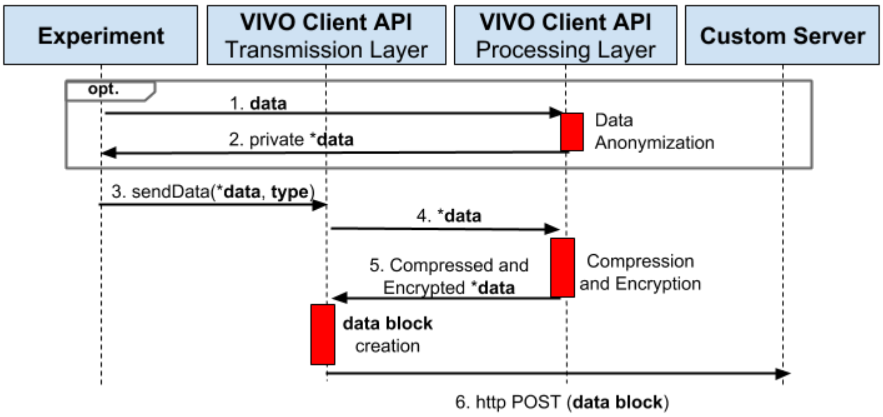

Experiment
VIVO Client API Transmission Layer
VIVO Client API Processing Layer
Custom Server
opt.
1. data
2. private *data
Data Anonymization
3. sendData(*data, type)
4. *data
5. Compressed and Encrypted *data
Compression and Encryption
data block creation
6. http POST (data block)

Figure 6.5: Sequence diagram of the VIVO API (real-time) data collection

of request customization are available, ranging from a simple request with only raw data to a highly customized HTTP request. Parameters available for customization are: request body, headers, path, and callback on response.

Fig. 6.5 shows a sequence diagram representing a real-time data collection scenario. In this figure, we depict the optional sequence of actions required to perform data anonymization, which can be implemented also in the offline scenario. The Processing Layer of the VIVO Client API is in charge of anonymizing, compressing, and encrypting data before the transmission to the Custom Server. Differently from the offline scenario - where a data block is buffered in a batch of data and then transmitted according to the Sync Adapter policy - in the real-time scenario each data block is sent to the Custom Server without any buffering and additional delay.

6.4.2.3 Compression and Encryption Tools

Compression and encryption are fundamental tools in the proposed architecture. These features are bundled in a utility Class and use only Java standard libraries. To compress data we use the Deflate algorithm, while for the encryption we adopt asymmetric encryption implemented by the RSA algorithm. Developers can freely decide the key size for the RSA algorithm[3]. This encryption technique is highly secure if a large key size is selected but it supports a limited data block size at each encryption, which in turn depends on the key size. Thus, developers should take into account the

[3] We strongly suggest to use a key size of at least a 3072-bit as recommended by NSA: https://cryptome.org/2016/01/CNSA-Suite-and-Quantum-Computing-FAQ.pdf

size of the data before using the API. The maximum block size b can be computed as: $b = k - p$, where k is the key size (in byte) and p is the padding size (currently fixed at 11 byte).

6.4.2.4 Privacy Module

One of the key points of the proposed architecture is to ensure security and privacy at the client side. The privacy module provides routines to anonymize data by removing any personal information from identifiers collected during an experiment. First, we created a simple tool, which consists of an Anonymous ID Generator. Given a set of IDs that can potentially be used to identify a user, it produces a new set of IDs using the SHA-256 algorithm [183]. This simple anonymization is, however, generally insufficient to preserve privacy, as any personal information can be used to identify a user. For this reason, in collaboration with our partners of the SwissSenseSinergy Project[4], we have implemented an interface to support differential privacy.

Differential privacy [76] is a property that provides an upper bound on the information a third party can obtain from the data after the release. Consider the experiment example of a simple survey composed of some yes-or-no questions. If we use a differentially private method to collect and analyze volunteer answers, then any third person that sees a statistical result over the answers (e.g., the proportion of participants saying yes) would not be able to identify the answers of individuals up to a certain specified privacy parameter called the *privacy loss*. The simplest method available is the Laplace Mechanism [77], which can be used when the statistic of interest is a real number. Within VIVO, it has also been implemented the Hybrid Mechanism [54], which can be seen as an extension of the Laplace Mechanism for streaming computations, and the Randomized Response [77] mechanism, which is applicable to any type of multiple-choice question.

The differential private methods available in the API are usable - both for offline and real-time settings - for experiments that can be *reduced to a survey*. Any experiment can be *reduced to a survey*, when the collected data belongs to a bounded set of numbers that correspond to possible answers. As an example, we consider a crowdsensing application that collects the heart rate of volunteers. This experiment can be viewed as a multiple choice survey where the sensors act as participants and the sensor measurements represent votes.

[4] http://www.swiss-sense-synergy.ch

6.4.3 VIVO Server

The *VIVO Server* is a Node.js web application based on the *KeystoneJS* Framework. It is backed by the VIVO DB, which is a MongoDB database. This architecture provides a versatile and flexible *No-SQL* backend solution suitable for the scale of this project. The VIVO Server supports the overall architecture handling the experiments, the notification mechanism, the data collection engine, and the integration with the Syndesi IoT testbed.

6.4.3.1 Experiment Management

A VIVO *experiment* is an instance of an Android application that is built considering the guidelines in the development tutorial. In particular, the developer needs only to (i) integrate and configure the VIVO Client API as a library in the application project, and (ii) name the application package with a fixed string. An experiment can be deployed only once. If developers want to run an experiment multiple times, they have to create new experiments based on the same application. Once a developer has been approved by the administrator, she/he can upload applications and manage experiments from the experiment page. Every time the developer instantiates an experiment, a dedicated page is created on the web interface for experiment management. From this page the developer can request the administrator approval, start, and stop the experiment. Once the approval is granted, the experiment will be made available to all the volunteers through the VIVO Client application. Finally, when developers stop the experiment they can download (from the experiment page) all the data collected from the experiment instances installed on volunteer devices.

6.4.3.2 Notifications

The VIVO Server makes use of Firebase Cloud Messaging (FCM)[5] to manage notifications. FCM is a cross-platform messaging solution for delivering notification messages at a very reduced cost to drive user re-engagement. By means of FCM, the VIVO Server sends notifications to the VIVO Client applications about new and finished experiments. FCM is also integrated with the VIVO Client. Each time an FCM message arrives at the VIVO Client, a system notification is issued and displayed on the notification bar.

[5] https://firebase.google.com/docs/cloud-messaging/

6.4.3.3 VIVO Server Data Collection

The *data collection system* manages the collected data in the VIVO Server, which periodically receives data blocks from volunteer devices. As described in Section 6.4.1, a single data block is composed of: *data*, *type*, *timestamp*, *experiment ID*, and *device ID*. Every time a device is synchronized with the VIVO Server, a batch of data blocks are sent in a JSON structure. A batch may contain data from different experiments. The VIVO Server dispatches each block to the correct database based on the *experiment ID*. Every time an experiment finishes, the VIVO Server creates a JSON structure that combines the experiment data, which the developers can download from the experiment page.

6.5 VIVO Human Behavior Application

Through the VIVO testbed, some real-world applications have been successfully implemented (e.g., indoor localization [248] and *NoiseBay*, an app to collect noise levels [45, 46]). In this Section, we present VIVO Human Behavior (VHB), an application we launched, through the VIVO framework, with the objective of raising users' awareness of OSN perils. We aim to achieve this objective by analyzing users' activities in OSNs and communicating them with their current risks in real-time. For this purpose, we rely on VIVO and its features. In particular, we develop VHB as an Android application that exploits VIVO functionalities through the VIVO Client API. The latter, which is embedded in the application as an external library, handles the anonymization and encryption procedures of the collected data. This level of protection allows us to preserve privacy and manage security issues before the data reach the VIVO Client and, in turn, the VIVO Server. Therefore, by running this application through the VIVO framework we can benefit from VIVO provisioning towards privacy and security, other than its real-time data collection feature.

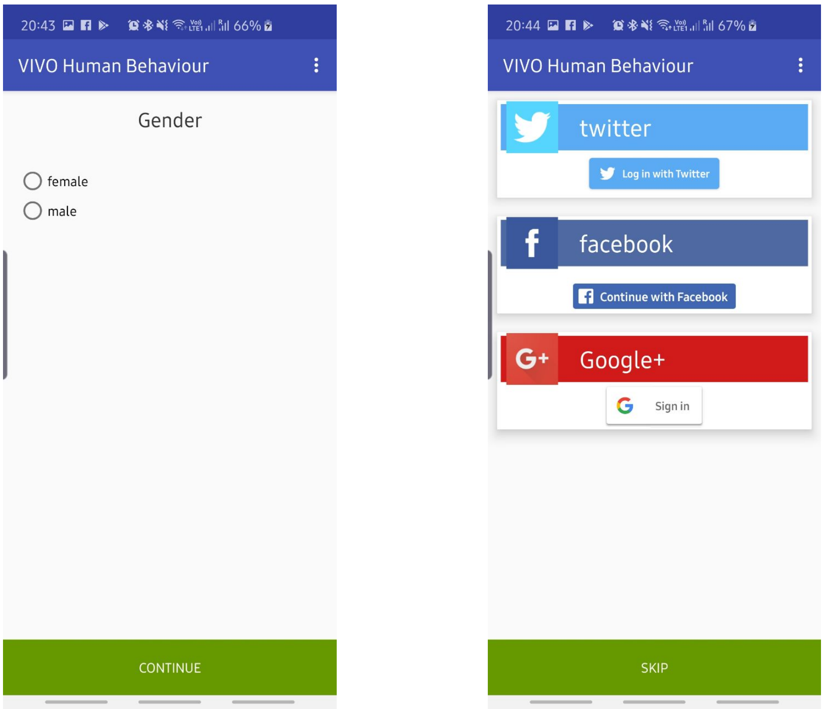
To increase users' awareness of OSN perils, in particular of the privacy and manipulation issues, the VHB experiment needs to collect users' data. Thereby, we envision this application as a service to first collect data in a privacy-preserving way, and to successively communicate to OSN users: (i) an assessment of their geo-location privacy (see Chapter 3), (ii) a measure of the predictability of their actions through the lens of our social influence model (see Chapter 4), and (iii) an alert every time they interact with malicious actors in OSNs (see Chapter 5). It should be noticed that VHB relies on the analysis of OSN risks presented in the previous chapters of this thesis (Chapters 3, 4, and 5) to evaluate users' perils, while exploits VIVO functionalities for a real-time, secure, and privacy-preserving data collection and communication with users. As explained in detail in Sections 6.5.1 and 6.5.2, users are only required to provide a small set of information at the beginning of the experiment as most of the data collection is performed in the background.

VHB also aims to demonstrate the flexibility of VIVO in collecting heterogeneous (type of) data. In fact, for VHB, we propose to develop an experiment that collects from volunteer smartphones physical (e.g., user's location and activity), social (e.g., OSN connections, call logs, and contacts), and environmental (e.g., weather) information. The rationale is to extend the analysis presented in previous chapters by leveraging different sources of information. For instance, we can rely on weather or activity information to enhance the inference of users' location, or we can use smartphone contacts and call logs to extract social relationships and enrich our social influence model with new social connections. Overall, we categorize the collected data into *static* and *dynamic* information according to their sampling frequency. In the next subsections, we detail the information collected in both categories.

6.5.1 Static Data Collection

Static data are collected once at the beginning of the experiment. In this phase, volunteers are requested to participate in a survey and log-in to their OSN accounts. The survey aims to collect volunteers' personal information, which can be used for further studies. Each volunteer is asked to provide information about her/his age (range), gender, nationality, job field (academy, company, or unemployed), phone number, home, and work location. As an example, Fig. 6.6a shows an instance of the survey. Volunteers can deliberately decide to provide the information in the survey (only age and gender are mandatory requested to provide some statistics about the sampled population). However, as mentioned in Section 6.4.2.4, we stress that VIVO privacy module allows us to both anonymize and utilize differential privacy methods for preserving volunteers privacy.

Volunteers are then requested to log-in to OSN platforms, such as Facebook, Twitter, Google+. This phase is not mandatory and volunteers can log-in or log-out whenever they prefer. Each OSN requires a specific access implementation and allows to collect a given subset of data. For Google+ and Facebook, we gather only the friends list of



(a)

(b)

Figure 6.6: VIVO Human Behavior application: (a) shows an instance of the survey, while (b) shows the log-in page to OSN platforms.

the volunteer. For Twitter, we collect only the user identifier, which we successively use to gather publicly-available information (tweets, list of followers/followee, etc.) via the Twitter API. Figure 6.6b display the OSNs log-in page of the VHB application.

6.5.2 Dynamic Data Collection

Dynamic data are continuously collected by the VHB application since the first boot. Through a persistent background service, VHB gathers information about volunteers':

- activity (walking, biking, running, etc.)
- contacts
- call logs
- location and corresponding weather
- phone state (i.e., connected cell ID information)

Activity is detected and collected every minute, along with the location information. Contacts (resp. call logs) are collected at the log-in time and then updated at each change (resp. call), whereas weather conditions are gathered every 30 minutes. Finally, phone state information is collected every time the cell status changes. The background collector service manages and launches specific sub-services for gathering these data. The only sub-service that does not rely on Android integrated services is related to the collection of the weather condition, for which we leverage the web service provided by OpenWeather[6].

6.5.3 VHB: Reliability and Challenges

To test the VHB application, we distributed the experiment to the volunteers through the VIVO Web Interface, as described in Section 6.3. Initially, we relied only on a small set of three volunteers to assess the functionality and potential bugs in the implementation of the application. This phase proved both (i) VIVO flexibility to collect heterogeneous data, and (ii) the integrity and correctness of the collected data during the whole experiment. Although the data collection did not present any issue, two challenges arose in the successive phase.

The first challenge involves the recruitment of a larger basis of volunteers, which is a typical issue in crowd-sensing platforms [56, 123, 203]. To motivate volunteers in the participation of the awareness campaign, we plan to develop a reward-based mechanism that will allow the participants to receive an economical reward according to their involvement in the campaign. The second challenge concerns the communication with volunteers about their privacy and manipulation risks (notice that the assessment of such issues follows the analysis detailed in the previous chapters of this thesis). While VIVO allows direct and real-time communication with volunteers, the presentation of the risks to the users still needs further investigation and is an open research problem [156]. The feedback should be easy to understand and provide useful recommendations to change users' behavior and avoid further misuse [18]. In [72], Dolan et al. delineated the psychological factors that can impact humans in changing their behavior. These mechanisms can affect users' motivation to actually adopt the knowledge offered by our awareness service and, therefore, we plan to consider these guidelines in our future works.

[6] openweathermap.org

6.6 System Performance

To validate the functionality and to evaluate the performance of the VIVO architecture, we developed different test-applications. In Section 6.6.1, we examine the scalability of VIVO by distributing an experiment to a group of volunteers scattered over the whole Switzerland. In Section 6.6.2, we present a comparison of the performance of the VIVO Client API with legacy solutions through a large suite of benchmarks. Finally, in Section 6.6.3 we compare the battery consumption of real-time upload with offline data collection.

6.6.1 Scalability Test

In this test, we examine the functionality and the scalability of VIVO by distributing an experiment to a group of forty volunteers scattered over the whole Switzerland. Further, this test allowed us to evaluate the robustness of VIVO by analyzing the integrity and the correctness of the collected data during the whole life-cycle, and the presence of anomalies or bugs in the implementation.

In the experiment, we gather accelerometer measurements from volunteer smartphones every minute in both offline and real-time settings. Volunteers installed the experiment from the VIVO Client, which worked without any issue. The VIVO Server handled well both the experiments and the volunteers, without any loss of data and any performance degradation. In the current version of the architecture, the VIVO Server is designed to run on a single node as a monolithic web application and, thus, it does not scale automatically on a cluster of multiple nodes. The VIVO Server instance runs on a machine with a CPU Intel(R) Pentium(R) D, dual core at 3.00GHz, 8GB DDR3 RAM, and 200GB HDD disk. To properly evaluate VIVO scalability, we should consider that the VIVO Server is a Node.js web application. Node.js operates on a single thread using non-blocking I/O calls. Thereby, it supports much more concurrent connections with respect to traditional web-serving techniques. Node architecture works well for tasks with non-intensive CPU computation, as for the VIVO Server, which performs light tasks at each synchronization. Concurrent connections capability can be computed taking into account the amount of RAM [2]. As an example, a traditional web server with 8GB of RAM can support at most few thousands of concurrent connections, while Node architecture can handle tens of thousands of simultaneous connections with the same amount of memory.

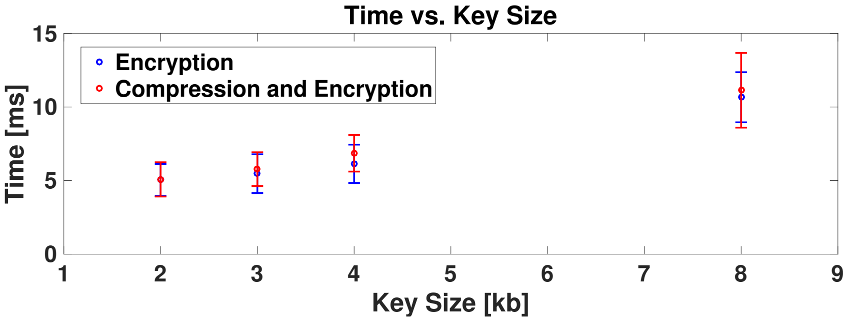


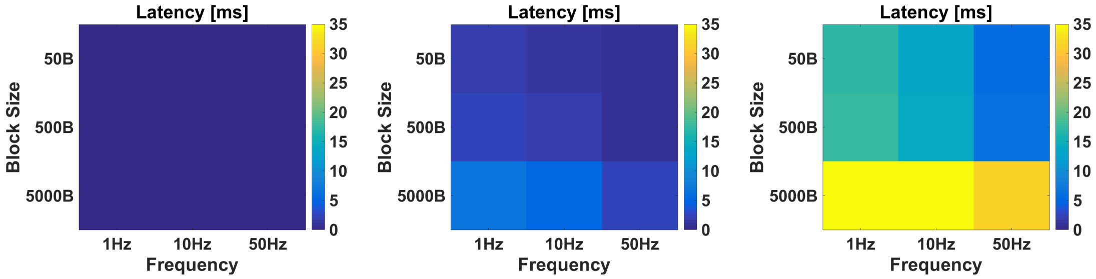
Figure 6.7: Data processing time vs. key size

6.6.2 VIVO Client API Performance

To guarantee security in the data transaction, the VIVO Client API performs data compression and encryption. It is crucial to ensure that this data processing does not affect the performance and the proper operation of VIVO, both in the real-time and in the offline settings. Low latency in the data processing is mandatory for real-time applications, while a moderate battery consumption is fundamental to support volunteer involvement.

To this end, we developed two classes of experiments. First, we evaluate the delay introduced by compression and encryption at varying key size. As VIVO developers decide the size of the RSA key during the API configuration, we aim to quantify the impact of this choice in terms of additional delay. To perform these measurements, we used a *Nexus 5X* running OS version 8.1.0. We compare our proposed solution, i.e., compression and encryption, with a standard security solution based only on encryption. Fig. 6.7 represents the time measured on the VIVO Client API to perform *(i)* only encryption and *(ii)* both compression and encryption on packages of 100 byte at varying encryption key size (2048, 3072, 4096, and 8192 bits). Our proposed solution closely approaches the processing time required by the standard security solution. Data compression allows encoding information using fewer bits than the original representation, while requiring, on average, only 5% of additional time if compared to the standard security solution.

Second, we created a suite of benchmarks to measure both latency and battery consumption, comparing three processing scenarios: (i) raw data (does not perform any data processing), (ii) compression, and (iii) compression and encryption (VIVO Client API). We performed a set of 27 benchmarks varying the three processing scenarios, the block size (50, 500, and 5000 byte), and the frequency (1, 10, and 50 Hz).



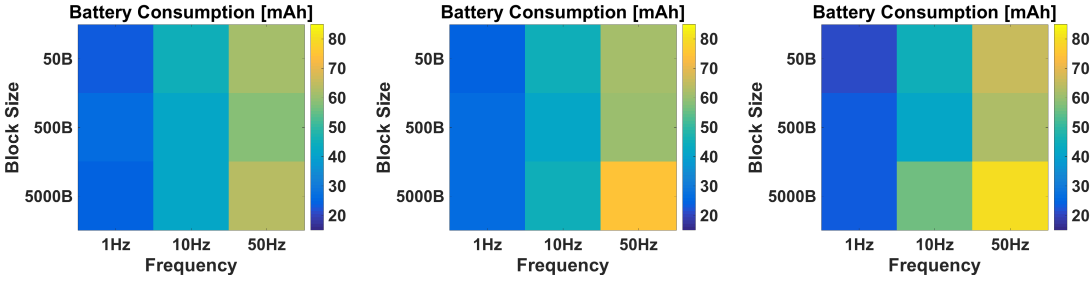
(a) Raw Data (b) Compression (c) VIVO Client API

Figure 6.8: Latency of the three processing scenarios

Each benchmark performs, for a given amount of time (fixed to 30 minutes), multiple data processing operations based on the frequency, which in turn determines the number of data processing operation performed in a second. We empirically choose the values of the parameters to exploit as much as possible the hardware resources at our disposal.

Fig. 6.8 compares the average latency over the processing instances of the three scenarios as a function of block size and frequency. As it can be observed in these figures, and contrarily from what we expected, for every data processing scenario, the higher the frequency the lower is the latency. Our hypothesis is that an optimization system dynamically adapts the resource allocation according to the throughput of the benchmark. We strongly believe that this optimization is performed by a Kernel component of the OS: the CPU Governor, which controls the CPU frequency in response to the demands of the running processes. Thereby, when the data processing frequency is low, the Governor maintains a lower CPU frequency - taking more time to process the data - with respect to the case of a higher data processing frequency. While the block size does not affect the raw data processing, in the other two scenarios we observe that the larger the block size the higher is the latency. An interesting behavior can be noticed in the VIVO Client API scenario, where encryption limits the data block size. In this experiment, we used a key size of 8192 bits, which allows to encrypt up to 1013 byte of data. Thereby, in case of larger blocks (e.g., 5000B), the API splits the data in smaller blocks introducing a computational overhead, as it can be appreciated in Fig. 6.8c. Finally, we observe that every parameter combination produces an acceptable delay in every processing scenario.

The battery consumption as a function of the benchmark parameters can be seen in Fig. 6.9. As we expected, for every data processing scenario, the battery consumption



(a) Raw Data (b) Compression (c) VIVO Client API

Figure 6.9: Battery consumption of the three processing scenarios

increases with the benchmark frequency. Note in particular that in a real experiment the frequency is set by the developer and depends on the purpose and on the requirements of the experiment itself. Further, block size does not significantly affect battery consumption in every scenario.

Overall, through these two classes of experiments, we proved that the VIVO API introduces a large suite of tools at the cost of a slightly larger latency and a moderate battery consumption if compared to legacy solutions.

6.6.3 Battery Consumption in Data Synchronization

In this subsection, we compare the offline data collection with the real-time upload in terms of battery consumption. In the former scenario, a batch of data is sent at irregular intervals based on the Sync Adapter policy, whereas in the latter, data is forwarded without any buffering. The Sync Adapter aims to transfer data while limiting the battery consumption according to the current network usage and the device sleep state. As the synchronization mechanism is strongly affected by the usage of the device, which in turn is a stochastic process, we forced the transmission every time a batch of fixed size, referred to as *queue*, is filled. In such a way the resulting consumption will be an upper bound of the real battery consumption, as the Sync Adapter manages the transmission more efficiently. In this test, we evaluate queue sizes ranging from 1 (real-time scenario) to 1000 elements. For each queue size, we performed a benchmark of 90 minutes sending data block of 50 byte at a frequency of 1, 10, and 50Hz.

Fig. 6.10 shows the battery consumption as a function of queue size and frequency. As it was expected, short queues consume more energy than longer ones as the system

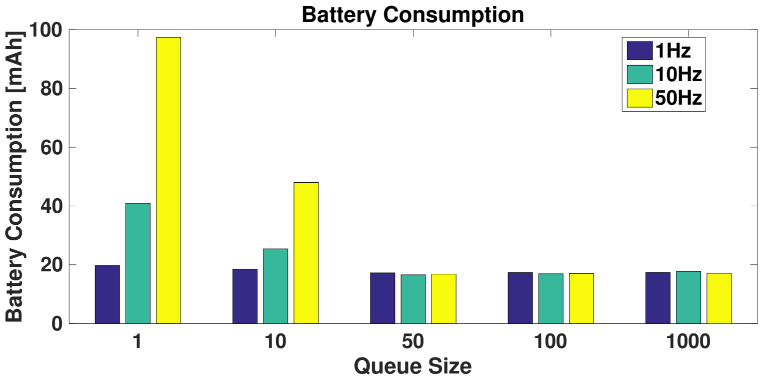


Figure 6.10: Battery consumption as a function of queue size and frequency

requires more frequently network infrastructure and communication. In particular, the battery consumption in the case of long queues achieves almost the same value. Our hypothesis is that, in such scenarios, the battery consumption converges to a lower bound, which does not depend on the frequency and on the queue size of the data upload.

6.7 Conclusions

In this Chapter, we aim to propose a service to raise users' awareness of OSN perils. To tackle the challenges of guaranteeing such a service and address RQ 4, we presented VIVO, a framework for collecting human behavioral data in a privacy-preserving way. Although VIVO has a broader objective of collecting crowd-sensed data and supporting application development, its provisioning of trust and privacy makes VIVO suitable for the objectives of this thesis. On one side, VIVO architecture provides experiment developers a secure and privacy-by-design facility for collecting data. On the other side, it allows us to develop a mobile application to collect data in real-time and directly communicate with the users through mobile applications. Towards the objective of increasing users' awareness of OSN perils, we develop a mobile application for collecting (in a privacy-preserving way) human behavioral data from volunteers' smartphone and to successively provide them feedback about their privacy and manipulation risks. We tested the functionality of the application and the integrity of the collected data by distributing the application to a small set of volunteers. In future works, we aim to develop a mechanism to reward and enlarge the group of volunteers. Also, we plan to research the most efficient and comprehensible way to provide users useful recommendations to restrain and mitigate their OSN risks.

7 Conclusions

The advent of OSNs have drastically changed our culture, society, and everyday life. The combination of OSN technological and social features, however, hides pitfalls still largely unknown to their users. Among all the perils presented in the literature, personal data privacy and the manipulation of public opinion have recently gathered concern for the individual users and the entire society, respectively.

For this reason, in this thesis, we have explored the nature of the privacy and manipulation perils with the final objective of raising users' awareness of these risks. More specifically, we have investigated how the networks structure of OSNs enables these perils, showing how social relationships and interactions play a significant role in fostering and exacerbating both issues. Interestingly, although the two perils appear disjointed and affect users at different granularity, we have shown that both of them are not under the single individual control and the whole society is involved.

In the next Sections, we detail our contributions and propose future directions.

7.1 Main Contributions

In this thesis, we have explored to what extent a generic user's privacy can be violated by leveraging information provided by other users in the OSN. In particular, we have examined the privacy issue by analyzing the problem of geo-location privacy on Twitter. In Chapter 3, we have provided users with means to measure and control the public exposure of their personal information. The first contribution of this thesis is related to the methodologies proposed to accomplish such purposes. In particular, to answer RQ 1.1, we have introduced a deep learning approach to assess the ability of an attacker to infer the location of a user given the locations of other OSN users. The results confirmed the concern about privacy leakage in OSNs and motivated us to investigate the usage of countermeasures to improve users' privacy. For this purpose, and to face RQ 1.2, we have explored the usage of two data perturbation strategies that users can apply to tune their desired level of privacy. Finally, to address RQ 1.3, we have proposed an interpretable model that provides users with a measure of their privacy and an understanding of the factors that impact it.

To further explore the idea of inferring personal information by leveraging other users' publicly available data, we have explored the social influence phenomenon. More specifically, we have investigated whether social influence modeling (i.e., to learn influence strength among users) can be used to predict and, in turn, violate users' personal information. In Chapter 4, we have examined how to model social influence among subjects to measure to what extent they are affected by others and, accordingly, infer their future real-life activity. Therefore, the second contribution of this thesis consists in the proposed approaches to model social influence and predict users' behavior. To overcome the limitations of existing solutions in the literature, and to address RQ 2.1, we have presented a novel approach based on deep learning, which outperforms existing solutions. Moreover, to respond to RQ 2.2, we have examined other factors that might impact social influence, other than direct social connections (e.g., friendships). For this purpose, we have provided an approach based on the collective influence of communities based on geographical location, homophily, and social ties. Our results validate our intuition and present further privacy concerns that involve our society as a whole. In fact, once again, we have shown that users' information domain is not only confined to what they deliberately share and, thus, the secrecy of their data depends on and is affected by others' decision to share information in OSNs.

Social influence has also been proven as an effective tool to affect and manipulate individuals' opinions in OSNs. In the political context, since the 2016 US Presidential election, there has been a big spotlight on the risks of mass manipulation of public opinion in OSNs. Although the attempt of OSN providers to purge their platforms, malicious actors (e.g., bots and trolls) persist in OSNs and still play a pivotal role in these manipulation campaigns. The detection of coordinated campaigns remains an open challenge for the research community: While bots have become increasingly sophisticated to mimic human behavior in order to escape detection, the automated detection of trolls has not found any established solution yet. In Chapter 5, to tackle these challenges, we have provided the following contributions. To answer RQ 3.1, we have performed an in-depth analysis of the evolution of bots during the last two US voting events. We have shown how such fake accounts have evolved to emulate human behavior and avoid detection. To respond to RQ 3.2, we have uncovered the strategy employed by bot accounts to manipulate human users and involve them in their conversation. Our findings revealed the mutable nature of bots, their effectiveness in engaging with humans, and posed new challenges in the detection of coordinated campaigns. However, our insights can inform actionable policies to empower bots detection and fight online abuse. Finally, to address RQ 3.3, we have proposed an approach to automatically detect trolls' activity in OSNs. Such an approach, based on IRL, accurately identifies such malicious actors and permits to unveil their distinctive behavior with respect to non-troll accounts.

In the final contribution of this thesis, we aim to raise users' awareness of OSN perils. For this purpose, we have proposed to implement a mobile application for communicating OSN users their current risks in real-time. However, this kind of service introduces further privacy and security risks. To face these challenges and address RQ 4, we have developed a framework, called VIVO, that allows to collect data in a secure and privacy-preserving way. In Chapter 6, we have presented VIVO and we have shown how this framework is beneficial for the purpose of this thesis. In fact, VIVO provides a privacy-by-design facility that allows us to deploy an application, which securely collects data in real-time and permits to directly communicate with the users. This, in turn, enables a mechanism of direct feedback through which it is possible to increase users' awareness of the risks behind their activity in OSNs.

7.2 Future Directions

In this thesis, we have examined the perils related to data privacy and manipulation of the public opinion in OSNs. Our investigation is not exhaustive and presents several open points and future directions.

As far as privacy is concerned, a possible extension of our work can be directed towards a more extended campaign to study users' awareness of, and reaction to, privacy risks when they utilize smartphones and OSNs. The rationale would be to investigate users' behavior before and after they become aware of privacy risks (e.g., if they reduce the amount of shared information). However, as we have described in Chapter 6, an open problem is to identify an understandable and effective way to communicate information about their current risks to the users. This, and a reward-based mechanism to recruit volunteers, represent the main challenges to propose an awareness service.

The major challenge for the manipulation risk is represented by the mutable nature of social bots. Their continuous presence in OSNs, coupled with their increasingly sophisticated strategy to escape detection and interact with humans, is cause of concern for the integrity of the online discussion and of global voting events. This set of open problems poses numerous challenges in the fight against OSN abuse and motivates further research for keeping the pace of malicious automated accounts. Along this research direction, a relevant peril is represented by the recent rise of a new form of spam, referred to as *spamming with AI* [85], which relies on AI to conduct spam operations of different sorts. Examples of spamming with AI are related to the alteration of video content by superimposing others' face and/or voice [224], which result indistinguishable to the human eye. These fake videos have been also diffused in the political context raising further concerns on the diffusion of misinformation [140].

While this thesis explores the privacy and manipulation risks in OSNs, other perils need to be investigated to prevent the abuse of OSNs (as we have detailed in Chapter 1). For example, cognitive absorption, addiction, and cyberbullying are relevant alarming problems related to the misuse of OSNs. Numerous studies analyzed cases of harassment in online platforms [129, 137] showing that an analysis of social interactions can lead to an automated detection of bullying episodes. However, the detection task is a difficult problem as cyberbullying manifests in different ways and, thus, remains an open challenge. Additionally, mechanisms to avoid health issues (e.g., addiction, depression, etc.) still need to be investigated and developed.

Finally, as we have shown throughout this thesis, each of these potential and effective perils exploit users' vulnerability. Human effort is a necessary ingredient to fight the harmful attempts to undermine the single individual and our entire society. Therefore, our future undertaking will be focused on proposing users a set of instruments to defend themselves from OSN perils.

Bibliography

[1] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning.* ACM, 2004, p. 1.

[2] M. Abernethy, "Just what is node. js," in *IBM Developer Works*, 2011.

[3] A. Acquisti and R. Gross, "Imagined communities: Awareness, information sharing, and privacy on the facebook," in *International workshop on privacy enhancing technologies.* Springer, 2006, pp. 36–58.

[4] L. Adamic, O. Buyukkokten, and E. Adar, "A social network caught in the web," *First monday*, vol. 8, no. 6, 2003.

[5] A. Addawood, A. Badawy, K. Lerman, and E. Ferrara, "Linguistic cues to deception: Identifying political trolls on social media," in *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 13, no. 01, 2019, pp. 15–25.

[6] A. Al Hasib, "Threats of online social networks," *IJCSNS International Journal of Computer Science and Network Security*, vol. 9, no. 11, pp. 288–93, 2009.

[7] A. Alarifi, M. Alsaleh, and A. Al-Salman, "Twitter turing test: Identifying social machines," *Information Sciences*, vol. 372, pp. 332–346, 2016.

[8] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Reviews of modern physics*, vol. 74, no. 1, p. 47, 2002.

[9] H. Allcott and M. Gentzkow, "Social media and fake news in the 2016 election," *Journal of Economic Perspectives*, vol. 31, no. 2, pp. 211–36, 2017.

[10] A. Anagnostopoulos, R. Kumar, and M. Mahdian, “Influence and correlation in social networks,” in *international conference on Knowledge discovery and data mining*, 2008, pp. 7–15.

[11] C. M. Angelopoulos, S. Nikoletseas, T. P. Raptis, and J. Rolim, “Design and evaluation of characteristic incentive mechanisms in mobile crowdsensing systems,” in *Simulation Modelling Practice and Theory*, vol. 55. Elsevier, 2015, pp. 95–106.

[12] C. M. Angelopoulos, S. Nikoletseas, T. P. Raptis, and J. D. Rolim, “Characteristic utilities, join policies and efficient incentives in mobile crowdsensing systems,” in *Wireless Days (WD), 2014 IFIP*. IEEE, 2014, pp. 1–6.

[13] J. Ao, P. Zhang, and Y. Cao, “Estimating the locations of emergency events from twitter streams,” *Procedia Computer Science*, vol. 31, pp. 731–739, 2014.

[14] S. Aral, L. Muchnik, and A. Sundararajan, “Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks,” *Proceedings of the National Academy of Sciences*, vol. 106, no. 51, pp. 21 544–21 549, 2009.

[15] S. Aral and D. Walker, “Identifying influential and susceptible members of social networks,” *Science*, vol. 337, no. 6092, pp. 337–341, 2012.

[16] M. Azzimonti and M. Fernandes, “Social media networks, fake news, and polarization,” National Bureau of Economic Research, Tech. Rep., 2018.

[17] L. Backstrom, C. Dwork, and J. Kleinberg, “Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography,” in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 181–190.

[18] M. Bada, A. M. Sasse, and J. R. Nurse, “Cyber security awareness campaigns: Why do they fail to change behaviour?” *arXiv preprint arXiv:1901.02672*, 2019.

[19] A. Badawy, A. Addawood, K. Lerman, and E. Ferrara, “Characterizing the 2016 russian ira influence campaign,” *Social Network Analysis and Mining*, vol. 9, no. 1, p. 31, 2019.

[20] A. Badawy, E. Ferrara, and K. Lerman, “Analyzing the digital traces of political manipulation: The 2016 russian interference twitter campaign,” in *Int. Conference on Advances in Social Networks Analysis and Mining*, 2018, pp. 258–265.

[21] A. Badawy, K. Lerman, and E. Ferrara, "Who falls for online political manipulation?" in *Companion Proceedings of The 2019 World Wide Web Conference*, 2019, pp. 162–168.

[22] J. P. Bagrow, X. Liu, and L. Mitchell, "Information flow reveals prediction limits in online social activity," *Nature Human Behaviour*, p. 1, 2019.

[23] C. A. Bail, L. P. Argyle, T. W. Brown, J. P. Bumpus, H. Chen, M. F. Hunzaker, J. Lee, M. Mann, F. Merhout, and A. Volfovsky, "Exposure to opposing views on social media can increase political polarization," *PNAS*, vol. 115, no. 37, pp. 9216–9221, 2018.

[24] E. Bakshy, J. M. Hofman, W. A. Mason, and D. J. Watts, "Everyone's an influencer: quantifying influence on twitter," in *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 2011, pp. 65–74.

[25] R. K. Balan, A. Misra, and Y. Lee, "Livelabs: Building an in-situ real-time mobile experimentation testbed," in *Proceedings of the 15th Workshop on Mobile Computing Systems and Applications*, 2014, pp. 1–6.

[26] M. Balduzzi, C. Platzer, T. Holz, E. Kirda, D. Balzarotti, and C. Kruegel, "Abusing social networks for automated user profiling," in *International Workshop on Recent Advances in Intrusion Detection*. Springer, 2010, pp. 422–441.

[27] J. Bao, Y. Zheng, and M. F. Mokbel, "Location-based and preference-aware recommendation using sparse geo-social networking data," in *Proceedings of the 20th international conference on advances in geographic information systems*. ACM, 2012, pp. 199–208.

[28] A.-L. Barabási, "Network science," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 371, no. 1987, p. 20120375, 2013.

[29] B. Baron and M. Musolesi, "Interpretable machine learning for privacy-preserving pervasive systems," *IEEE Pervasive Computing*, 2020.

[30] L. Bedogni, M. Di Felice, and L. Bononi, "By train or by car? detecting the user's motion type through smartphone sensors data," in *2012 IFIP Wireless Days*. IEEE, 2012, pp. 1–6.

[31] J. Benesty, J. Chen, Y. Huang, and I. Cohen, “Pearson correlation coefficient,” in *Noise reduction in speech processing.* Springer, 2009, pp. 1–4.

[32] A. R. Beresford, A. Rice, N. Skehin, and R. Sohan, “Mockdroid: trading privacy for application functionality on smartphones,” in *Proceedings of the 12th workshop on mobile computing systems and applications.* ACM, 2011, pp. 49–54.

[33] A. R. Beresford and F. Stajano, “Mix zones: User privacy in location-aware services,” in *IEEE Annual Conference on Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second.* IEEE, 2004, pp. 127–131.

[34] L. F. Berkman, “Assessing the physical health effects of social networks and social support,” *Annual review of public health*, vol. 5, no. 1, pp. 413–432, 1984.

[35] A. Bessi, M. Coletto, G. A. Davidescu, A. Scala, G. Caldarelli, and W. Quattrociocchi, “Science vs conspiracy: Collective narratives in the age of misinformation,” *PloS one*, vol. 10, no. 2, p. e0118093, 2015.

[36] A. Bessi and E. Ferrara, “Social bots distort the 2016 us presidential election online discussion,” *First Monday*, vol. 21, no. 11, 2016.

[37] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, “Fast unfolding of communities in large networks,” *Journal of statistical mechanics: theory and experiment*, vol. 2008, no. 10, p. P10008, 2008.

[38] O. Boichak, S. Jackson, J. Hemsley, and S. Tanupabrungsun, “Automated diffusion? bots and their influence during the 2016 us presidential election,” in *International Conference on Information.* Springer, 2018, pp. 17–26.

[39] P. Bonacich, “Power and centrality: A family of measures,” *American journal of sociology*, vol. 92, no. 5, pp. 1170–1182, 1987.

[40] A. Bovet and H. A. Makse, “Influence of fake news in twitter during the 2016 us presidential election,” *Nature communications*, vol. 10, no. 1, p. 7, 2019.

[41] U. Brandes, “A faster algorithm for betweenness centrality,” *Journal of mathematical sociology*, vol. 25, no. 2, pp. 163–177, 2001.

[42] S. Brooks and P. Longstreet, “Social networking’s peril: Cognitive absorption, social networking usage, and depression,” *Cyberpsychology: Journal of Psychosocial Research on Cyberspace*, vol. 9, no. 4, 2015.

[43] J. Brown, A. J. Broderick, and N. Lee, “Word of mouth communication within online communities: Conceptualizing the online social network,” *Journal of interactive marketing*, vol. 21, no. 3, pp. 2–20, 2007.

[44] E. E. Buckels, P. D. Trapnell, and D. L. Paulhus, “Trolls just want to have fun,” *Personality and individual Differences*, vol. 67, pp. 97–102, 2014.

[45] J. Buwaya and J. D. P. Rolim, “Atomic routing mechanisms for balance of costs and quality in mobile crowdsensing systems,” in *2017 13th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE, 2017, pp. 147–154.

[46] ——, “Mobile crowdsensing from a selfish routing perspective,” in *2017 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. IEEE, 2017, pp. 1457–1463.

[47] C. Cadwalladr and E. Graham-Harrison, “Revealed: 50 million facebook profiles harvested for cambridge analytica in major data breach,” *The guardian*, vol. 17, p. 22, 2018.

[48] O. Carroll, “St petersburg ‘troll farm’had 90 dedicated staff working to influence us election campaign. independent,” 2017.

[49] C. Castellano, S. Fortunato, and V. Loreto, “Statistical physics of social dynamics,” *Reviews of modern physics*, vol. 81, no. 2, p. 591, 2009.

[50] D. Centola, “The spread of behavior in an online social network experiment,” *science*, vol. 329, no. 5996, pp. 1194–1197, 2010.

[51] ——, “An experimental study of homophily in the adoption of health behavior,” *Science*, vol. 334, no. 6060, pp. 1269–1272, 2011.

[52] D. Centola and M. Macy, “Complex contagions and the weakness of long ties,” *American journal of Sociology*, vol. 113, no. 3, pp. 702–734, 2007.

[53] A. Chaabane, G. Acs, M. A. Kaafar *et al.*, “You are what you like! information leakage through users’ interests,” in *Proceedings of the 19th Annual Network & Distributed System Security Symposium (NDSS)*. Citeseer, 2012.

[54] T.-H. H. Chan, E. Shi, and D. Song, “Private and continual release of statistics,” *ACM Transactions on Information and System Security (TISSEC)*, vol. 14, no. 3, pp. 1–24, 2011.

[55] N. Chavoshi, H. Hamooni, and A. Mueen, “Debot: Twitter bot detection via warped correlation.” in *ICDM*, 2016, pp. 817–822.

[56] Y. Chen, H. Chen, S. Yang, X. Gao, and F. Wu, “Jump-start crowdsensing: A three-layer incentive framework for mobile crowdsensing,” in *2017 IEEE/ACM 25th International Symposium on Quality of Service (IWQoS)*. IEEE, 2017, pp. 1–6.

[57] Z. Chen and D. Subramanian, “An unsupervised approach to detect spam campaigns that use botnets on twitter,” *arXiv preprint arXiv:1804.05232*, 2018.

[58] Z. Cheng, J. Caverlee, and K. Lee, “You are where you tweet: a content-based approach to geo-locating twitter users,” in *Proceedings of the 19th ACM international conference on Information and knowledge management*. ACM, 2010, pp. 759–768.

[59] J. Choi and K.-E. Kim, “Bayesian nonparametric feature construction for inverse reinforcement learning,” in *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

[60] Y. Chon, N. D. Lane, F. Li, H. Cha, and F. Zhao, “Automatically characterizing places with opportunistic crowdsensing using smartphones,” in *Proceedings of the 2012 ACM conference on ubiquitous computing*, 2012, pp. 481–490.

[61] W.-H. Chong and E.-P. Lim, “Exploiting contextual information for fine-grained tweet geolocation,” in *Eleventh International AAAI Conference on Web and Social Media*, 2017.

[62] N. R. Chopde and M. Nichat, “Landmark based shortest path detection by using a* and haversine formula,” *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 1, no. 2, pp. 298–302, 2013.

[63] N. A. Christakis and J. H. Fowler, “The spread of obesity in a large social network over 32 years,” *New England journal of medicine*, vol. 357, no. 4, pp. 370–379, 2007.

[64] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” *arXiv preprint arXiv:1412.3555*, 2014.

[65] D. Crandall, D. Cosley, D. Huttenlocher, J. Kleinberg, and S. Suri, "Feedback effects between similarity and social influence in online communities," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2008, pp. 160–168.

[66] P. Dabkowski and Y. Gal, "Real time image saliency for black box classifiers," in *Advances in Neural Information Processing Systems*, 2017, pp. 6967–6976.

[67] S. Das and A. Lavoie, "The effects of feedback on human behavior in social media: An inverse reinforcement learning model," in *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2014, pp. 653–660.

[68] Y. Dauphin, H. de Vries, and Y. Bengio, "Equilibrated adaptive learning rates for non-convex optimization," in *Advances in neural information processing systems*, 2015, pp. 1504–1512.

[69] C. A. Davis, O. Varol, E. Ferrara, A. Flammini, and F. Menczer, "Botornot: A system to evaluate social bots," in *Proceedings of the 25th International Conference Companion on World Wide Web*, 2016, pp. 273–274.

[70] A. Deb, L. Luceri, A. Badawy, and E. Ferrara, "Perils and challenges of social media and election manipulation analysis: The 2018 us midterms," in *Companion Proceedings of the 2019 World Wide Web Conference*, 2019, pp. 237–247.

[71] M. Del Vicario, F. Zollo, G. Caldarelli, A. Scala, and W. Quattrociocchi, "Mapping social dynamics on facebook: The brexit debate," *Social Networks*, vol. 50, pp. 6–16, 2017.

[72] P. Dolan, M. Hallsworth, D. Halpern, D. King, and I. Vlaev, "Mindspace: influencing behaviour for public policy," 2010.

[73] P. Domingos and M. Richardson, "Mining the network value of customers," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, 2001, pp. 57–66.

[74] C. Drummond, R. C. Holte *et al.*, "C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling," in *Workshop on learning from imbalanced datasets II*, vol. 11. Citeseer, 2003, pp. 1–8.

[75] R. Dutt, A. Deb, and E. Ferrara, ““senator, we sell ads”: Analysis of the 2016 russian facebook ads campaign,” in *International Conference on Intelligent Information Technologies*. Springer, 2018, pp. 151–168.

[76] C. Dwork, “Differential privacy,” in *International conference on theory and applications of models of computation*. Springer, 2008, pp. 1–19.

[77] C. Dwork, A. Roth *et al.*, “The algorithmic foundations of differential privacy,” *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[78] J. Eisenstein, B. O’Connor, N. A. Smith, and E. P. Xing, “A latent variable model for geographic lexical variation,” in *Proceedings of the 2010 conference on empirical methods in natural language processing*. Association for Computational Linguistics, 2010, pp. 1277–1287.

[79] O. Evangelatos, K. Samarasinghe, and J. Rolim, in *2013 IEEE international conference on distributed computing in sensor systems*. IEEE, 2013, pp. 325–330.

[80] X. Fang, P. J.-H. Hu, Z. Li, and W. Tsai, “Predicting adoption probabilities in social networks,” *Information Systems Research*, vol. 24, no. 1, pp. 128–145, 2013.

[81] J. Fernandes, M. Nati, N. Loumis, S. Nikoletseas, T. P. Raptis, S. Krco, A. Rankov, S. Jokic, C. M. Angelopoulos, and S. Ziegler, “Iot lab: Towards co-design and iot solution testing using the crowd,” in *2015 International Conference on Recent Advances in Internet of Things (RIoT)*. IEEE, 2015, pp. 1–6.

[82] E. Ferrara, “Manipulation and abuse on social media by emilio ferrara with ching-man au yeung as coordinator,” *ACM SIGWEB Newsletter*, no. Spring, p. 4, 2015.

[83] ——, “Disinformation and social bot operations in the run up to the 2017 french presidential election,” *First Monday*, vol. 22, no. 8, 2017.

[84] ——, “Bots, elections, and social media: a brief overview,” *arXiv preprint arXiv:1910.01720*, 2019.

[85] ——, “The history of digital spam,” *Communications of the ACM*, vol. 62, no. 8, pp. 82–91, 2019.

[86] E. Ferrara, O. Varol, C. Davis, F. Menczer, and A. Flammini, “The rise of social bots,” *Communications of the ACM*, vol. 59, no. 7, pp. 96–104, 2016.

[87] E. Ferrara, O. Varol, F. Menczer, and A. Flammini, “Detection of promoted social media campaigns,” in *Tenth International AAAI Conference on Web and Social Media*, 2016, pp. 563–566.

[88] A. Ferrari, D. Gallucci, D. Puccinelli, and S. Giordano, “Detecting energy leaks in android app with poem,” in *2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*. IEEE, 2015, pp. 421–426.

[89] A. Ferrari, D. Puccinelli, and S. Giordano, “Managing your privacy in mobile applications with mockingbird,” in *2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*. IEEE, 2015, pp. 288–291.

[90] J. Fogel and E. Nehmad, “Internet social network communities: Risk taking, trust, and privacy concerns,” *Computers in human behavior*, vol. 25, no. 1, pp. 153–160, 2009.

[91] R. C. Fong and A. Vedaldi, “Interpretable explanations of black boxes by meaningful perturbation,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3429–3437.

[92] A. Förster, K. Garg, H.-A. Nguyen, and S. Giordano, “On context awareness and social distance in human mobility traces,” in *Proceedings of the third ACM international workshop on Mobile Opportunistic Networks*, 2012, pp. 5–12.

[93] S. Fortunato, “Community detection in graphs,” *Physics reports*, vol. 486, no. 3-5, pp. 75–174, 2010.

[94] freeCodeCamp, “An introduction to q-learning: reinforcement learning.” [Online]. Available: https://www.freecodecamp.org/news/an-introduction-to-q-learning-reinforcement-learning-14ac0b4493cc/

[95] L. C. Freeman, “Centrality in social networks conceptual clarification,” *Social networks*, vol. 1, no. 3, pp. 215–239, 1978.

[96] V. Gadde and Y. Roth, “Enabling further research of information operations on twitter,” 2018.

[97] D. Garcia, “Privacy beyond the individual,” *Nature Human Behaviour*, vol. 3, no. 2, p. 112, 2019.

[98] L. Garton, C. Haythornthwaite, and B. Wellman, “Studying online social networks,” *Journal of computer-mediated communication*, vol. 3, no. 1, p. JCMC313, 1997.

[99] P. Georgiev, A. Noulas, and C. Mascolo, “The call of the crowd: Event participation in location-based social services,” 2014.

[100] T. P. Gerber and J. Zavisca, “Does russian propaganda work?” *The Washington Quarterly*, vol. 39, no. 2, pp. 79–98, 2016.

[101] S. Giordano, V. Morel, M. Onen, M. Musolesi, D. Andreoletti, F. Cardoso, A. Ferrari, L. Luceri, C. Castelluccia, D. Le Metayer *et al.*, “Uprise-iot: User-centric privacy & security in the iot,” *Security and Privacy in Internet of Things: Challenges and Solutions*, 2019.

[102] M. Girvan and M. E. Newman, “Community structure in social and biological networks,” *Proceedings of the national academy of sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.

[103] J. Goldenberg, B. Libai, and E. Muller, “Talk of the network: A complex systems look at the underlying process of word-of-mouth,” *Marketing letters*, vol. 12, no. 3, pp. 211–223, 2001.

[104] J. Gottfried and E. Shearer, *News Use Across Social Medial Platforms 2016*. Pew Research Center, 2016.

[105] T. Govani and H. Pashley, “Student awareness of the privacy implications when using facebook,” *unpublished paper presented at the “Privacy Poster Fair” at the Carnegie Mellon University School of Library and Information Science*, vol. 9, pp. 1–17, 2005.

[106] A. Goyal, F. Bonchi, and L. V. Lakshmanan, “Learning influence probabilities in social networks,” in *ACM international conference on Web search and data mining*, 2010, pp. 241–250.

[107] P. Goyal and E. Ferrara, “Graph embedding techniques, applications, and performance: A survey,” *Knowledge-Based Systems*, vol. 151, pp. 78–94, 2018.

[108] M. Graham, S. A. Hale, and D. Gaffney, “Where in the world are you? geolocation and language identification in twitter,” *The Professional Geographer*, vol. 66, no. 4, pp. 568–578, 2014.

[109] S. Graham, “The end of geography or the explosion of place? conceptualizing space, place and information technology,” *Progress in human geography*, vol. 22, no. 2, pp. 165–185, 1998.

[110] C. W. Granger, “Investigating causal relations by econometric models and cross-spectral methods,” *Econometrica: Journal of the Econometric Society*, pp. 424–438, 1969.

[111] N. Grinberg, K. Joseph, L. Friedland, B. Swire-Thompson, and D. Lazer, “Fake news on Twitter during the 2016 U.S. presidential election,” *Science*, vol. 363, no. 6425, pp. 374–378, 2019.

[112] M. Grossglauser and D. N. Tse, “Mobility increases the capacity of ad hoc wireless networks,” *IEEE/ACM Transactions on Networking (ToN)*, vol. 10, no. 4, pp. 477–486, 2002.

[113] D. Gruhl, R. Guha, D. Liben-Nowell, and A. Tomkins, “Information diffusion through blogspace,” in *Proceedings of the 13th international conference on World Wide Web*, 2004, pp. 491–501.

[114] M. Gruteser and D. Grunwald, “Anonymous usage of location-based services through spatial and temporal cloaking,” in *Proceedings of the 1st international conference on Mobile systems, applications and services*. ACM, 2003, pp. 31–42.

[115] A. Guess, J. Nagler, and J. Tucker, “Less than you think: Prevalence and predictors of fake news dissemination on facebook,” *Science Advances*, vol. 5, no. 1, p. eaau4586, 2019.

[116] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi, “A survey of methods for explaining black box models,” *ACM computing surveys (CSUR)*, vol. 51, no. 5, p. 93, 2018.

[117] A. Guille, H. Hacid, C. Favre, and D. A. Zighed, “Information diffusion in online social networks: A survey,” *ACM Sigmod Record*, vol. 42, no. 2, pp. 17–28, 2013.

[118] I. Gulenko, “Social against social engineering: Concept and development of a facebook application to raise security and risk awareness,” *Information Management & Computer Security*, vol. 21, no. 2, pp. 91–101, 2013.

[119] D. Gunning, “Explainable artificial intelligence (xai),” *Defense Advanced Research Projects Agency (DARPA), nd Web*, vol. 2, 2017.

[120] B. Guo, H. Chen, Z. Yu, X. Xie, S. Huangfu, and D. Zhang, “Fliermeet: a mobile crowdsensing system for cross-space public information reposting, tagging, and sharing,” in *IEEE Transactions on Mobile Computing*, vol. 14, no. 10. IEEE, 2014, pp. 2020–2033.

[121] B. Guo, Z. Yu, X. Zhou, and D. Zhang, “From participatory sensing to mobile crowd sensing,” in *2014 IEEE International Conference on Pervasive Computing and Communication Workshops (PERCOM WORKSHOPS)*. IEEE, 2014, pp. 593–598.

[122] W. Hamilton, Z. Ying, and J. Leskovec, “Inductive representation learning on large graphs,” in *Advances in Neural Information Processing Systems*, 2017, pp. 1024–1034.

[123] G. Han, L. Liu, S. Chan, R. Yu, and Y. Yang, “Hysense: A hybrid mobile crowdsensing framework for sensing opportunities compensation under dynamic coverage constraint,” *IEEE Communications Magazine*, vol. 55, no. 3, pp. 93–99, 2017.

[124] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[125] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, “Neural collaborative filtering,” in *Proceedings of the 26th international conference on world wide web*, 2017, pp. 173–182.

[126] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[127] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady, “Preserving privacy in gps traces via uncertainty-aware path cloaking,” in *Proceedings of the 14th ACM conference on Computer and communications security*. ACM, 2007, pp. 161–171.

[128] Hootsuite, “The global state of digital in 2019 report,” 2019. [Online]. Available: https://hootsuite.com/pages/digital-in-2019

[129] H. Hosseinmardi, S. Li, Z. Yang, Q. Lv, R. I. Rafiq, R. Han, and S. Mishra, “A comparison of common users across instagram and ask. fm to better understand cyberbullying,” in *2014 IEEE Fourth International Conference on Big Data and Cloud Computing*. IEEE, 2014, pp. 355–362.

[130] P. N. Howard, G. Bolsover, B. Kollanyi, S. Bradshaw, and L.-M. Neudert, "Junk news and bots during the us election: What were michigan voters sharing over twitter," *CompProp, OII, Data Memo*, 2017.

[131] P. N. Howard, B. Kollanyi, and S. Woolley, "Bots and automation over twitter during the us election," *Computational Propaganda Project: Working Paper Series*, 2016.

[132] N. Huete-Alcocer, "A literature review of word of mouth and electronic word of mouth: Implications for consumer behavior," *Frontiers in psychology*, vol. 8, p. 1256, 2017.

[133] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. Miu, E. Shih, H. Balakrishnan, and S. Madden, "Cartel: a distributed mobile sensor computing system," in *Sensys*. ACM, 2006.

[134] J. Im, E. Chandrasekharan, J. Sargent, P. Lighthammer, T. Denby, A. Bhargava, L. Hemphill, D. Jurgens, and E. Gilbert, "Still out there: Modeling and identifying russian troll accounts on twitter," *arXiv preprint arXiv:1901.11162*, 2019.

[135] M. Jacomy, T. Venturini, S. Heymann, and M. Bastian, "Forceatlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software," *PloS one*, vol. 9, no. 6, p. e98679, 2014.

[136] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.

[137] H.-T. Kao, S. Yan, D. Huang, N. Bartley, H. Hosseinmardi, and E. Ferrara, "Understanding cyberbullying on instagram and ask. fm via social role detection," in *Companion Proceedings of The 2019 World Wide Web Conference*. ACM, 2019, pp. 183–188.

[138] R. Ke, W. Li, Z. Cui, and Y. Wang, "Two-stream multi-channel convolutional neural network (tm-cnn) for multi-lane traffic speed prediction considering traffic volume impact," *arXiv preprint arXiv:1903.01678*, 2019.

[139] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the international conference on Knowledge discovery and data mining*. ACM, 2003, pp. 137–146.

[140] H. Kim, P. Carrido, A. Tewari, W. Xu, J. Thies, M. Niessner, P. Pérez, C. Richardt, M. Zollhöfer, and C. Theobalt, "Deep video portraits," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, p. 163, 2018.

[141] M. Kimura, K. Saito, and R. Nakano, "Extracting influential nodes for information diffusion on a social network," in *AAAI*, vol. 7, 2007, pp. 1371–1376.

[142] S. Kinsella, V. Murdock, and N. O'Hare, "I'm eating a sandwich in glasgow: modeling locations with tweets," in *Proceedings of the 3rd international workshop on Search and mining user-generated contents*. ACM, 2011, pp. 61–68.

[143] B. Kollanyi, P. N. Howard, and S. C. Woolley, "Bots and automation over twitter during the first us presidential debate," *Comprop data memo*, vol. 1, pp. 1–4, 2016.

[144] M. Kosinski, D. Stillwell, and T. Graepel, "Private traits and attributes are predictable from digital records of human behavior," *Proceedings of the National Academy of Sciences*, vol. 110, no. 15, pp. 5802–5805, 2013.

[145] S. Kudugunta and E. Ferrara, "Deep neural networks for bot detection," *Information Sciences*, vol. 467, no. October, pp. 312–322, 2018.

[146] R. Kumar, J. Novak, and A. Tomkins, "Structure and evolution of online social networks," in *Link mining: models, algorithms, and applications*. Springer, 2010, pp. 337–357.

[147] T. La Fond and J. Neville, "Randomization tests for distinguishing social influence and homophily effects," in *Proceedings of the 19th international conference on World wide web*, 2010, pp. 601–610.

[148] G. Larkou, M. Mintzis, S. Taranto, A. Konstantinidis, P. G. Andreou, and D. Zeinalipour-Yazti, "Demonstration abstract: Sensor mockup experiments with smartlab," in *IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*. IEEE, 2014, pp. 339–340.

[149] D. M. Lazer, M. A. Baum, Y. Benkler, A. J. Berinsky, K. M. Greenhill, F. Menczer, M. J. Metzger, B. Nyhan, G. Pennycook, D. Rothschild *et al.*, "The science of fake news," *Science*, vol. 359, no. 6380, pp. 1094–1096, 2018.

[150] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[151] K. Lerman and T. Hogg, “Using a model of social dynamics to predict popularity of news,” in *Proceedings of the 19th international conference on World wide web.* ACM, 2010, pp. 621–630.

[152] K. Lerman, S. Intagorn, J.-H. Kang, and R. Ghosh, “Using proximity to predict activity in social networks,” in *Proceedings of the 21st International Conference on World Wide Web.* ACM, 2012, pp. 555–556.

[153] S. Levine, Z. Popovic, and V. Koltun, “Feature construction for inverse reinforcement learning,” in *Advances in Neural Information Processing Systems*, 2010, pp. 1342–1350.

[154] ——, “Nonlinear inverse reinforcement learning with gaussian processes,” in *Advances in Neural Information Processing Systems*, 2011, pp. 19–27.

[155] Y. Li, Y. Li, Q. Yan, and R. H. Deng, “Privacy leakage analysis in online social networks,” *Computers & Security*, vol. 49, pp. 239–254, 2015.

[156] B. Y. Lim and A. K. Dey, “Toolkit to support intelligibility in context-aware applications,” in *Proceedings of the 12th ACM international conference on Ubiquitous computing.* ACM, 2010, pp. 13–22.

[157] J. Lingad, S. Karimi, and J. Yin, “Location extraction from disaster-related microblogs,” in *Proceedings of the 22nd international conference on world wide web.* ACM, 2013, pp. 1017–1020.

[158] Z. C. Lipton, “The mythos of model interpretability,” *Queue*, vol. 16, no. 3, pp. 31–57, 2018.

[159] L. Liu, J. Tang, J. Han, and S. Yang, “Learning influence from heterogeneous social networks,” *Data Mining and Knowledge Discovery*, vol. 25, no. 3, pp. 511–544, 2012.

[160] X. Liu, Q. He, Y. Tian, W.-C. Lee, J. McPherson, and J. Han, “Event-based social networks: linking the online and offline social worlds,” in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2012, pp. 1032–1040.

[161] X.-Y. Liu, J. Wu, and Z.-H. Zhou, “Exploratory undersampling for class-imbalance learning,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 2, pp. 539–550, 2008.

[162] L. Luceri, “Infer mobility patterns and social dynamics for modelling human behaviour,” in *Data Mining Workshops (ICDMW), 2016 IEEE 16th International Conference on*. IEEE, 2016, pp. 1223–1224.

[163] L. Luceri, D. Andreoletti, M. Tornatore, T. Braun, and S. Giordano, “Measurement and control of geo-location privacy on twitter,” *Conditionally accepted in Online Social Networks and Media, Elsevier*, 2019.

[164] L. Luceri, T. Braun, and S. Giordano, “Social influence (deep) learning for human behavior prediction,” in *International Workshop on Complex Networks*. Springer, 2018, pp. 261–269.

[165] ——, “Analyzing and inferring human real-life behavior through online social networks with social influence deep learning,” *Applied Network Science*, vol. 4, no. 1, p. 34, 2019.

[166] L. Luceri, F. Cardoso, M. Papandrea, S. Giordano, J. Buwaya, S. Kundig, C. M. Angelopoulos, J. Rolim, Z. Zhao, J. L. Carrera, T. Braun, A. C. Tossou, C. Dimitrakakis, and A. Mitrokotsa, “Vivo: A secure, privacy-preserving, and real-time crowd-sensing framework for the internet of things,” *Pervasive and Mobile Computing*, vol. 49, pp. 126–138, 2018.

[167] L. Luceri, A. Deb, A. Badawy, and E. Ferrara, “Red bots do it better: Comparative analysis of social bot partisan behavior,” in *Companion Proceedings of The 2019 World Wide Web Conference*, 2019, pp. 1007–1012.

[168] L. Luceri, A. Deb, S. Giordano, and E. Ferrara, “Evolution of bot and human behavior during elections,” *First Monday*, vol. 24, no. 9, 2019.

[169] L. Luceri, S. Giordano, and E. Ferrara, “Don’t feed the troll: Detecting troll behavior via inverse reinforcement learning,” *Submitted for publication*, 2019.

[170] L. Luceri, A. Vancheri, T. Braun, and S. Giordano, “On the social influence in human behavior: Physical, homophily, and social communities,” in *International Conference on Complex Networks and Their Applications*. Springer, 2017.

[171] Y. Matsubara, Y. Sakurai, B. A. Prakash, L. Li, and C. Faloutsos, “Rise and fall patterns of information diffusion: model and implications,” in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012, pp. 6–14.

[172] M. McPherson, L. Smith-Lovin, and J. M. Cook, “Birds of a feather: Homophily in social networks,” *Annual review of sociology*, vol. 27, no. 1, pp. 415–444, 2001.

[173] P. Metaxas, E. Mustafaraj, K. Wong, L. Zeng, M. O’Keefe, and S. Finn, “What do retweets indicate? results from user survey and meta-review of research,” in *Ninth International AAAI Conference on Web and Social Media*, 2015.

[174] P. T. Metaxas and E. Mustafaraj, “Social media and the elections,” *Science*, vol. 338, no. 6106, pp. 472–473, 2012.

[175] S. Milgram, “The small world problem,” *Psychology today*, vol. 2, no. 1, pp. 60–67, 1967.

[176] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee, “Measurement and analysis of online social networks,” in *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*. ACM, 2007, pp. 29–42.

[177] P. Mohan, V. N. Padmanabhan, and R. Ramjee, “Nericell: rich monitoring of road and traffic conditions using mobile smartphones,” in *Proceedings of the 6th ACM conference on Embedded network sensor systems*, 2008, pp. 323–336.

[178] B. Mønsted, P. Sapieżyński, E. Ferrara, and S. Lehmann, “Evidence of complex contagion of information in social media: An experiment using twitter bots,” *PloS one*, vol. 12, no. 9, p. e0184148, 2017.

[179] F. Montori, L. Bedogni, A. Di Chiappari, and L. Bononi, “Sensquare: A mobile crowdsensing architecture for smart cities,” in *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*. IEEE, 2016, pp. 536–541.

[180] S. Mudda, M. Zignani, S. Gaito, S. Giordano, and G. P. Rossi, “Timely and personalized services using mobile cellular data,” *Online Social Networks and Media*, vol. 13, p. 100048, 2019.

[181] S. A. Myers, C. Zhu, and J. Leskovec, “Information diffusion and external influence in networks,” in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012, pp. 33–41.

[182] A. Nandugudi, A. Maiti, T. Ki, F. Bulut, M. Demirbas, T. Kosar, C. Qiao, S. Y. Ko, and G. Challen, “Phonelab: A large programmable smartphone testbed,” in *Proceedings of First International Workshop on Sensing and Big Data Mining*. ACM, 2013, pp. 1–6.

[183] National Institute of Standards and Technology (NIST), "Fips-180-2: Secure hash standard," 2002, [www.itl.nist.gov/fipspubs].

[184] M. E. Newman, "The structure and function of complex networks," *SIAM review*, vol. 45, no. 2, pp. 167–256, 2003.

[185] A. Y. Ng, S. J. Russell *et al.*, "Algorithms for inverse reinforcement learning." in *Icml*, vol. 1, 2000, pp. 663–670.

[186] A. Noulas, S. Scellato, N. Lathia, and C. Mascolo, "Mining user mobility features for next place prediction in location-based services," in *2012 IEEE 12th international conference on data mining*. IEEE, 2012, pp. 1038–1043.

[187] A. Noulas, S. Scellato, C. Mascolo, and M. Pontil, "An empirical study of geographic user activity patterns in foursquare." 2011.

[188] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web." Stanford InfoLab, Tech. Rep., 1999.

[189] B. Pan, Y. Zheng, D. Wilkie, and C. Shahabi, "Crowd sensing of traffic anomalies based on human mobility and social media," in *Proceedings of the 21st ACM SIGSPATIAL international conference on advances in geographic information systems*, 2013, pp. 344–353.

[190] M. Papandrea, K. K. Jahromi, M. Zignani, S. Gaito, S. Giordano, and G. P. Rossi, "On the properties of human mobility," *Computer Communications*, vol. 87, pp. 19–36, 2016.

[191] D. Pedreschi, F. Giannotti, R. Guidotti, A. Monreale, S. Ruggieri, and F. Turini, "Meaningful explanations of black box ai decision systems," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 9780–9784.

[192] B. Perez, M. Musolesi, and G. Stringhini, "You are your metadata: Identification and obfuscation of social media users using metadata information," in *Twelfth International AAAI Conference on Web and Social Media*, 2018.

[193] N. Persily, "The 2016 us election: Can democracy survive the internet?" *Journal of democracy*, vol. 28, no. 2, pp. 63–76, 2017.

[194] I. Polakis, G. Argyros, T. Petsios, S. Sivakorn, and A. D. Keromytis, "Where's wally?: Precise user discovery attacks in location proximity services," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2015, pp. 817–828.

[195] J. Poncela-Casasnovas, M. Gutiérrez-Roig, C. Gracia-Lázaro, J. Vicens, J. Gómez-Gardeñes, J. Perelló, Y. Moreno, J. Duch, and A. Sánchez, "Humans display a reduced set of consistent behavioral phenotypes in dyadic games," *Science advances*, vol. 2, no. 8, p. e1600451, 2016.

[196] S. Pötzsch, "Privacy awareness: A means to solve the privacy paradox?" in *IFIP Summer School on the Future of Identity in the Information Society*. Springer, 2008, pp. 226–236.

[197] I. Pozzana and E. Ferrara, "Measuring bot and human behavioral dynamics," *arXiv preprint arXiv:1802.04286*, 2018.

[198] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.

[199] D. Ramachandran and E. Amir, "Bayesian inverse reinforcement learning." in *IJCAI*, vol. 7, 2007, pp. 2586–2591.

[200] J. Ratkiewicz, M. Conover, M. R. Meiss, B. Gonçalves, A. Flammini, and F. Menczer, "Detecting and tracking political abuse in social media." in *Fifth international AAAI conference on weblogs and social media*, vol. 11, 2011, pp. 297–304.

[201] N. D. Ratliff, J. A. Bagnell, and M. A. Zinkevich, "Maximum margin planning," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 729–736.

[202] S. Reddy, A. Dragan, and S. Levine, "Where do you think you're going?: Inferring beliefs about dynamics from behavior," in *Advances in Neural Information Processing Systems*, 2018, pp. 1454–1465.

[203] F. Restuccia, N. Ghosh, S. Bhattacharjee, S. K. Das, and T. Melodia, "Quality of information in mobile crowdsensing: Survey and research challenges," *ACM Transactions on Sensor Networks (TOSN)*, vol. 13, no. 4, pp. 1–43, 2017.

[204] N. Rhinehart and K. Kitani, “First-person activity forecasting from video with online inverse reinforcement learning,” *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[205] M. Richardson and P. Domingos, “Mining knowledge-sharing sites for viral marketing,” in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2002, pp. 61–70.

[206] D. Rosenblum, “What anyone can know: The privacy risks of social networking sites,” *IEEE Security & Privacy*, vol. 5, no. 3, pp. 40–49, 2007.

[207] A. Sadilek, H. Kautz, and J. P. Bigham, “Finding your friends and following them to where you are,” in *Proceedings of the fifth ACM international conference on Web search and data mining*. ACM, 2012, pp. 723–732.

[208] K. Saito, R. Nakano, and M. Kimura, “Prediction of information diffusion probabilities for independent cascade model,” in *International conference on knowledge-based and intelligent information and engineering systems*, 2012, pp. 67–75.

[209] M. Salehan and A. Negahban, “Social networking on smartphones: When mobile phones become addictive,” *Computers in Human Behavior*, vol. 29, no. 6, pp. 2632–2639, 2013.

[210] D. A. Scheufele and N. M. Krause, “Science audiences, misinformation, and fake news,” *Proceedings of the National Academy of Sciences*, vol. 116, no. 16, pp. 7662–7669, 2019.

[211] J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural networks*, vol. 61, pp. 85–117, 2015.

[212] A. Schulz, A. Hadjakos, H. Paulheim, J. Nachtwey, and M. Mühlhäuser, “A multi-indicator approach for geolocalization of tweets,” in *Seventh international AAAI conference on weblogs and social media*, 2013.

[213] C. Shao, G. L. Ciampaglia, O. Varol, K.-C. Yang, A. Flammini, and F. Menczer, “The spread of low-credibility content by social bots,” *Nature communications*, vol. 9, no. 1, p. 4787, 2018.

[214] R. Shokri, G. Theodorakopoulos, J.-Y. Le Boudec, and J.-P. Hubaux, "Quantifying location privacy," in *2011 IEEE symposium on security and privacy*. IEEE, 2011, pp. 247–262.

[215] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media: A data mining perspective," *ACM SIGKDD Explorations Newsletter*, vol. 19, no. 1, pp. 22–36, 2017.

[216] P. Singla and M. Richardson, "Yes, there is a correlation:-from social networks to personal behavior on the web," in *Proceedings of the 17th international conference on World Wide Web*, 2008, pp. 655–664.

[217] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[218] K. Starbird, "Examining the alternative media ecosystem through the production of alternative narratives of mass shooting events on twitter," in *Eleventh International AAAI Conference on Web and Social Media*, 2017.

[219] M. Stella, M. Cristoforetti, and M. De Domenico, "Influence of augmented humans in online interactions during voting events," *PloS one*, vol. 14, no. 5, p. e0214210, 2019.

[220] M. Stella, E. Ferrara, and M. De Domenico, "Bots increase exposure to negative and inflammatory content in online social systems," *Proceedings of the National Academy of Sciences*, vol. 115, no. 49, pp. 12 435–12 440, 2018.

[221] A. Striegel, S. Liu, L. Meng, C. Poellabauer, D. Hachen, and O. Lizardo, "Lessons learned from the netsense smartphone study," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 51–56, 2013.

[222] V. Subrahmanian, A. Azaria, S. Durst, V. Kagan, A. Galstyan, K. Lerman, L. Zhu, E. Ferrara, A. Flammini, F. Menczer *et al.*, "The darpa twitter bot challenge," *Computer*, vol. 49, no. 6, pp. 38–46, 2016.

[223] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[224] S. Suwajanakorn, S. M. Seitz, and I. Kemelmacher-Shlizerman, “Synthesizing obama: learning lip sync from audio,” *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, p. 95, 2017.

[225] J. Tang, J. Sun, C. Wang, and Z. Yang, “Social influence analysis in large-scale networks,” in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 807–816.

[226] M. Thelwall, K. Buckley, G. Paltoglou, D. Cai, and A. Kappas, “Sentiment strength detection in short informal text,” *Journal of the American Society for Information Science and Technology*, vol. 61, no. 12, pp. 2544–2558, 2010.

[227] D. S. Touretzky and L. M. Saksida, “Operant conditioning in skinnerbots,” *Adaptive Behavior*, vol. 5, no. 3-4, pp. 219–247, 1997.

[228] M. Tsvetkova, T. Yasseri, E. T. Meyer, J. B. Pickering, V. Engen, P. Walland, M. Lüders, A. Følstad, and G. Bravos, “Understanding human-machine networks: a cross-disciplinary survey,” *ACM Computing Surveys (CSUR)*, vol. 50, no. 1, pp. 1–35, 2017.

[229] H. Vandebosch and K. Van Cleemput, “Defining cyberbullying: A qualitative research into the perceptions of youngsters,” *CyberPsychology & Behavior*, vol. 11, no. 4, pp. 499–503, 2008.

[230] E. Vanderhoven, T. Schellens, and M. Valcke, “Exploring the usefulness of school education about risks on social network sites: A survey study,” *Journal of media literacy education*, vol. 5, no. 1, pp. 285–294, 2013.

[231] O. Varol, E. Ferrara, C. A. Davis, F. Menczer, and A. Flammini, “Online human-bot interactions: Detection, estimation, and characterization,” in *Int. AAAI Conference on Web and Social Media*, 2017, pp. 280–289.

[232] O. Varol, E. Ferrara, F. Menczer, and A. Flammini, “Early detection of promoted campaigns on social media,” *EPJ Data Science*, vol. 6, no. 13, 2017.

[233] F. J. Villanueva, D. Villa, M. J. Santofimia, J. Barba, and J. C. Lopez, “Crowdsensing smart city parking monitoring,” in *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*. IEEE, 2015, pp. 751–756.

[234] S. Vosoughi, D. Roy, and S. Aral, “The spread of true and false news online,” *Science*, vol. 359, no. 6380, pp. 1146–1151, 2018.

[235] G. Wang, X. Zhang, S. Tang, H. Zheng, and B. Y. Zhao, “Unsupervised clickstream clustering for user behavior analysis,” in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems.* ACM, 2016, pp. 225–236.

[236] C. J. C. H. Watkins, “Learning from delayed rewards,” Ph.D. dissertation, King’s College, Cambridge, 1989.

[237] D. J. Watts and S. H. Strogatz, “Collective dynamics of ‘small-world’networks,” *nature*, vol. 393, no. 6684, p. 440, 1998.

[238] S. Wojcik, S. Messing, A. Smith, L. Rainie, and P. Hitlin, “Bots in the twittersphere,” *Pew Research Center. Retrieved May*, vol. 22, p. 2018, 2018.

[239] S. C. Woolley and D. R. Guilbeault, “Computational propaganda in the united states of america: Manufacturing consensus online,” *Computational Propaganda Research Project*, vol. 22, 2017.

[240] M. Wulfmeier, P. Ondruska, and I. Posner, “Maximum entropy deep inverse reinforcement learning,” *arXiv preprint arXiv:1507.04888*, 2015.

[241] Z. Xu, L. Mei, K.-K. R. Choo, Z. Lv, C. Hu, X. Luo, and Y. Liu, “Mobile crowd sensing of human-like intelligence using social sensors: A survey,” *Neurocomputing*, vol. 279, pp. 3–10, 2018.

[242] K.-C. Yang, O. Varol, C. A. Davis, E. Ferrara, A. Flammini, and F. Menczer, “Arming the public with artificial intelligence to counter social bots,” *Human Behavior and Emerging Technologies*, p. e115, 2019.

[243] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. M. Thalmann, “Who, where, when and what: discover spatio-temporal topics for twitter users,” in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 2013, pp. 605–613.

[244] S. Zannettou, T. Caulfield, W. Setzer, M. Sirivianos, G. Stringhini, and J. Blackburn, “Who let the trolls out?: Towards understanding state-sponsored trolls,” in *Proceedings of the 10th ACM Conference on Web Science.* ACM, 2019, pp. 353–362.

[245] J. Zhang, B. Liu, J. Tang, T. Chen, and J. Li, “Social influence locality for modeling retweeting behaviors.” in *Twenty-Third International Joint Conference on Artificial Intelligence*, vol. 13, 2013, pp. 2761–2767.

[246] J. Zhang, J. Tang, J. Li, Y. Liu, and C. Xing, "Who influenced you? predicting retweet via social influence locality," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 9, no. 3, p. 25, 2015.

[247] R. Zhao, C. Yue, and Q. Han, "Cross-site input inference attacks on mobile web users," in *International Conference on Security and Privacy in Communication Systems*. Springer, 2017, pp. 629–643.

[248] Z. Zhao, T. Braun, Z. Li, A. Neto *et al.*, "A real-time robust indoor tracking system in smartphones," *Computer communications*, vol. 117, pp. 104–115, 2018.

[249] E. Zheleva and L. Getoor, "To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles," in *Proceedings of the 18th international conference on World wide web*. ACM, 2009, pp. 531–540.

[250] X. Zheng, J. Han, and A. Sun, "A survey of location prediction on twitter," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 9, pp. 1652–1671, 2018.

[251] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *AAAI*, vol. 8. Chicago, IL, USA, 2008, pp. 1433–1438.

Declaration of consent

on the basis of Article 28 of the RSL Phil.-nat. 05

Name/First Name: Luca Luceri

Registration Number: 15-115-199

Study program: Computer Science

Bachelor □ Master □ Dissertation ☑

Title of the thesis: Privacy Leakage and the Manipulation of Public Opinion in Online Social Networks

Supervisor: Prof. Dr. Torsten Braun, Prof. Dr. Silvia Giordano

I declare herewith that this thesis is my own work and that I have not used any sources other than those stated. I have indicated the adoption of quotations as well as thoughts taken from other authors as such in the thesis. I am aware that the Senate pursuant to Article 36 paragraph 1 litera r of the University Act of 5 September, 1996 is authorized to revoke the title awarded on the basis of this thesis.
For the purposes of evaluation and verification of compliance with the declaration of originality and the regulations governing plagiarism, I hereby grant the University of Bern the right to process my personal data and to perform the acts of use this requires, in particular, to reproduce the written thesis and to store it permanently in a database, and to use said database, or to make said database available, to enable comparison with future theses submitted by others.

Place/Date
Lugano, 20.02.2020

Signature

Curriculum Vitae

LUCA LUCERI

Via Bellinzona 17, Como, Italy | P: +39 3482119756 | luceriluc90@gmail.com

EDUCATION

University of Bern *Bern, Switzerland*
PhD in Computer Science *June 2015 - April 2020*

Polytechnic University of Milan *Milan, Italy*
*MSc in Telecommunication Engineering (*110/110 with honors*)* *September 2012 - December 2014*

Polytechnic University of Bari *Bari, Italy*
*BSc in Telecommunication Engineering (*110/110 with honors*)* *September 2009 - October 2012*

EXPERIENCE

SUPSI *Manno, CH*
Researcher *June 2015 - February 2020*

Information Science Institute - University of Southern California *Marina del Rey, CA, USA*
Research Intern *September 2018 - May 2019*

University of New South Wales *Sydney, NSW, Australia*
Research Intern – MSc Thesis *March 2014 - August 2014*

Publication in this Thesis

[1] Luca Luceri, Davide Andreoletti, Massimo Tornatore, Torsten Braun, and Silvia Giordano. “Measurement and Control of Geo-Location Privacy on Twitter”. *Conditionally accepted in Online Social Networks and Media*

[2] Luca Luceri. “Infer Mobility Patterns and Social Dynamics for Modelling Human Behaviour”. In: *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW).*

[3] Luca Luceri, Torsten Braun, and Silvia Giordano. “Analyzing and inferring human real-life behavior through online social networks with social influence deep learning”. In: *Applied Network Science,* 2019*.*

[4] Luca Luceri, Torsten Braun, and Silvia Giordano. “Social Influence (Deep) Learning for Human Behavior Prediction”. In: *International Workshop on Complex Networks IX (CompleNet 2018).*

[5] Luca Luceri, Alberto Vancheri, Torsten Braun, and Silvia Giordano. “On the Social Influence in Human Behavior: Physical, Homophily, and Social Communities”. In: *International Conference on Complex Networks and their Applications,* 2017.

[6] Luca Luceri, Ashok Deb, Silvia Giordano, and Emilio Ferrara. “Evolution of bot and human behavior during elections”. In: *First Monday,* 2019*.*

[7] Luca Luceri, Ashok Deb, Adam Badawy, and Emilio Ferrara. “Red Bots Do It Better: Comparative Analysis of Social Bot Partisan Behavior”. In: *WWW '19 Companion Proceedings of The 2019 World Wide Web Conference.*

[8] Luca Luceri, Silvia Giordano, and Emilio Ferrara. “Don’t feed the troll: Detecting troll behavior via inverse reinforcement learning*”. Submitted for publication.*

[9] Luca Luceri, Felipe Cardoso, Michela Papandrea, Silvia Giordano, Julia Buwaya, Stephane Kundig, Constantinos Marios Angelopoulos, Jose Rolim, Zhongliang Zhao, Jose Luis Carrera, Torsten Braun, Aristide Charles Yedia Tossou, and Christos Dimitrakakis. “VIVO: a Secure, Privacy-Preserving and Real-Time Crowd-Sensing Framework for the Internet of Things”. In: *Pervasive and Mobile Computing,* 2018*.*

[10] Ashok Deb, Luca Luceri, Adam Badawy, and Emilio Ferrara, "Perils and challenges of social media and election manipulation analysis: The 2018 US midterms". In: *WWW '19 Companion Proceedings of The 2019 World Wide Web Conference.*

[11] Silvia Giordano, Victor Morel, Melek Onen, Mirco Musolesi, Davide Andreoletti, Felipe Cardoso, Alan Ferrari, Luca Luceri, Claude Castelluccia, Daniel Le Metayer *et al.*, "Uprise-IoT: User-centric privacy & security in the IoT". In: *Security and Privacy in Internet of Things: Challenges and Solutions,* 2019.

AWARDS AND ACHIEVEMENTS

Best students' paper award (issued by Facebook at the Web Conference 2019) *May 2019*
Description: scholarships for the best students' papers during the Web Conference (WWW) workshop on Misinformation, Computational Fact-Checking and Credible Web

Mobility grant (issued by SNSF) *September 2018 - May 2019*
Description: mobility grants for doctoral students to improve their scientific profile by going abroad while being employed in an on-going SNSF research project.

Exchange program scholarship (issued by Politecnico di Milano) *March 2014 - August 2014*
Description: scholarship for exchange programs within the Master of Science at Politecnico di Milano

MEDIA COVERAGE

How malicious Twitter bots evolve to evade detection
The Irish Times

Twitter bots were more active than previously known during the 2018 midterms, a new study suggests
CNBC

Bots might prove harder to detect in 2020 elections
EurekAlert

Get ready for more humanlike bots, better deep-fake videos and wall-to-wall disinformation in 2020 race
The Baltimore Sun

Bots harder to discern from humans have multiplied in online 'arms race'
Silicon Republic

Twitter Bots Are Becoming More Human-Like: Study
Defense One