Saarland University
Faculty of Mathematics and Computer Science
Department of Computer Science

An analysis of Privacy and Group Fairness issues of Online Social Network and Fitness Tracker users

Dissertation
zur Erlangung des Grades
der Doktorin der Ingenieurwissenschaften
der Fakultät für Mathematik und Informatik
der Universität des Saarlandes

von
Tahleen Awazid Rahman

Saarbrücken, 2023

Tag des Kolloquiums:	14. Mai 2024
Dekan:	Prof. Dr. Roland Speicher

Prüfungsausschuss:

Vorsitzender:	Prof. Dr-Ing. Thorsten Herfet
Berichterstattende:	Prof. Dr. Dr. h.c. Michael Backes
	Prof. Dr. Jilles Vreeken
	Dr. Mathias Humbert
Akademischer Mitarbeiter:	Dr. Zhiqiu Jiang

Zusammenfassung

Das Aufkommen intelligenter Handheld- und Wearable-Gerätetechnologien sowie Fortschritte auf dem Gebiet der künstlichen Intelligenz und des maschinellen Lernens haben den Lebensstil der Menschen im letzten Jahrzehnt nach und nach enorm verändert. Unter ihnen sind Online Social Networks (OSNs) und Fitness-Tracker zu integralen Bestandteilen des täglichen Lebens von Millionen von Menschen auf der ganzen Welt geworden. Diese Tools bieten nicht nur eine breite Palette an Funktionalitäten, sondern ermöglichen es den Menschen auch, jeden Tag riesige Datenmengen zu teilen, was aus Sicht des Datenschutzes ein ernstes Problem darstellt. Darüber hinaus reagieren die Algorithmen des maschinellen Lernens, die den Kern vieler von OSNs und Fitness-Trackern angebotener Funktionen bilden, sehr empfindlich gegenüber Verzerrungen, die in den Trainingsdaten vorherrschen.

Diese Dissertation präsentiert eine Analyse von Datenschutz- und Fairness-Problemen, die sich aus der nachgelagerten Verarbeitung solcher benutzergenerierten Daten ergeben. Wir präsentieren zunächst eine Analyse der Privatsphäre für mehrere Szenarien, nämlich: Rückschluss auf den Standort des Benutzers, soziale Beziehungen, sensible Benutzerattribute, nämlich Geschlecht, Alter und Bildung, sowie Benutzerverknüpfbarkeit entlang der zeitlichen Dimension. Wir demonstrieren die Schwere der Datenschutzangriffe durch eine umfassende Auswertung realer Datensätze und leiten daraus wichtige Erkenntnisse ab. Wir stellen ein System namens Tagvisor vor, das verschiedene Verschleierungstechniken verwendet, um einen der Angriffe zu vereiteln, nämlich die Standortinduktion aus Hashtags. Zweitens analysieren wir die Fairness von node2vec, einer beliebten Methode zur Grapheneinbettung, die wir in unserer Analyse von OSNs verwenden. Unsere Experimente zeigen die Existenz von Bias in node2vec, wenn es für Freundschaftsempfehlungen verwendet wird. Wir schlagen eine fairnessbewusste Einbettungsmethode vor, nämlich Fairwalk, die node2vec erweitert und zeigen, dass Fairwalk die Verzerrung unter mehreren Fairness-Metriken reduziert und gleichzeitig den Nutzen beibehält.

Abstract

The advent of smart handheld and wearable device technologies along with advances in the field of Artificial Intelligence and Machine Learning have progressively changed people's lifestyle immensely over the last decade. Among them, Online Social Networks (OSNs) and fitness trackers have become integral parts of daily lives of millions of people around the world. In addition to offering a wide range of functionalities, these tools allow people to share huge amounts of data everyday, which becomes a serious concern from the privacy point of view. Moreover, the machine learning algorithms which are at the core of many features offered by OSNs and fitness trackers are quite sensitive to bias that is prevalent in the training data.

This dissertation presents an analysis of privacy and fairness issues that arise out of downstream processing of such user generated data. We first present an analysis of privacy for multiple scenarios namely: inference of user location, social relationships, sensitive user attributes namely gender, age and education as well as user linkability along the temporal dimension. We demonstrate the severity of the privacy attacks by an extensive evaluation on real life datasets and derive key insights. We introduce a system called Tagvisor, that uses various obfuscation techniques for thwarting one of the attacks, namely location inference from hashtags. Secondly, we analyze the fairness of node2vec, a popular graph embedding method, which we use in our analysis of OSNs. Our experiments demonstrate the existence of bias in node2vec when used for friendship recommendation. We propose a fairness-aware embedding method, namely Fairwalk, which extends node2vec and demonstrate that Fairwalk reduces bias under multiple fairness metrics while still preserving the utility.

Background of this Dissertation

This dissertation is based on the papers mentioned in the following. I contributed to all papers as one of the main authors.

The idea for the Tagvisor paper [P1] came from a discussion with Mathias Humbert and Yang Zhang. The dataset was collected by Yang Zhang. The attacks and counter measures were designed during discussions between Mathias Humbert, Yang Zhang and Tahleen Rahman. Yang Zhang and Tahleen Rahman ran the experiments for the evaluations. All authors reviewed the paper.

The idea for the privacy analysis using multimodal information came [P2] from Yang Zhang. The idea of using ResNet for the image attack came from Mario Fritz. Tahleen Rahman designed and performed the experiments with feedback from Yang Zhang. All authors reviewed the paper.

The idea for the privacy analysis of step count data [P3] came from a discussion with Michael Backes and Yang Zhang. The feature generation part was designed and implemented jointly by Tahleen Rahman and Bartlomiej Surma. Tahleen Rahman designed, implemented and evaluated the linkability attacks. Bartlomiej Surma designed and implemented and evaluated the attribute inference attacks. All authors reviewed the paper.

The idea of conducting a fairness assessment on node2vec [P4] came from Yang Zhang during a workshop on Big Data. Tahleen Rahman and Bartlomiej Surma jointly designed performed the analysis of algorithmic fairness on node2vec. Tahleen Rahman extended the notion of Statistical Parity and Equality of Representation for problems on graphs. Bartlomiej Surma implemented the Fairwalk algorithm as a modification of node2vec. Tahleen Rahman implemented the recommendation system. Bartlomiej Surma evaluated the equality of representation for user level. Tahleen Rahman performed the statistical parity, Equality of Representation for group level and Precision and Recall parts of the experimental evaluation. All authors contributed to reviewing the paper.

[P1] Zhang, Y., Humbert, M., Rahman, T., Li, C.-T., Pang, J., and Backes, M. Tagvisor: a privacy advisor for sharing hashtags. In: *Proceedings of the 2018 World Wide Web Conference (WWW)*. 2018, 287–296.

[P2] Rahman, T., Fritz, M., Backes, M., and Zhang, Y. Everything about you: a multimodal approach towards friendship inference in online social networks. In: *arXiv:2003.00996*. 2018.

[P3] Surma, B., Rahman, T., Backes, M., and Zhang, Y. You are how you walk: quantifying privacy risks in step count data. In: *arXiv:2308.04933*. 2021.

[P4] Rahman, T., Surma, B., Backes, M., and Zhang, Y. Fairwalk: towards fair graph embedding. In: *Proceedings of the 2019 International Joint Conferences on Artifical Intelligence (IJCAI)*. 2019.

Acknowledgments

First and foremost, I would like to thank my advisor Michael Backes for giving me the opportunity to pursue my PhD in the Information Security & Cryptography Group of CISPA at Saarland University. It has been an invaluable experience and great pleasure learning from him. I am very honored to be one of his students and very grateful for his guidance and his confidence in me.

I would like to express my gratitude towards my thesis committee members Michael Backes, Jilles Vreeken and Mathias Humbert for their time and effort spent reviewing this dissertation.

I am very thankful to my collaborators and co-authors for our inspiring discussions and their contributions to this thesis. My closest collaborator Yang Zhang, who introduced me to the field of Social Networks research and was my guiding light throughout the entire duration my PhD, deserves my particular thanks. Next, I would thank my colleague and close friend Bartlomiej Surma for the productive discussions surrounding our research, for trusting me, for always being there whenever I needed motivation, for never-ending enlightening information on various topics and for always having something to eat in the office during late nights or missed lunches during deadlines. I would also like to express my deepest gratitude to my office mates Inken and Ahmed and also Kathrin for their valuable advice on Machine Learning topics, reviewing my papers and their drive for knowledge sharing that has significantly shaped my PhD.

The friends I made at CISPA made my work environment a million times better and I will hold them close to my heart forever. Thanks to Sven and Sanam for the countless times I locked myself out of the office and countless rides to the gym, Marie and David for always somehow having an answer to my questions, along with Rui always being ready to get hotpot at China Restaurant, Ivan for uplifting walks in the woods. Special thanks to Bettina for helping me get important paperwork done on time, going above and beyond to help out during crisis and all the fun conversations that uplifted my spirits.

Last but not least, I am very grateful to my husband and my parents, for their support, their faith in me, and their invaluable advice in many situations. Without all of these people, I could never manage to be in the wonderful place I am today.

Contents

List of Figures

List of Tables

List of Algorithms

1 Introduction

Nowadays, people widely use online social networks (OSNs) and instant messaging to communicate with friends and maintain social relationships. This has paved the way for new concepts such as 'like', 'follow' and 'share' which have assimilated into the daily vocabulary of OSN users within no time. Among these, the "hashtag" has gained huge popularity in online culture and campaigns. While bringing significant utility benefits, hashtags may carry sensitive private information about people using them. However, its implications on people's privacy had not been investigated earlier. In the first part of this dissertation, we address privacy raised by hashtags. We concentrate in particular on location, which is recognized as one of the key privacy concerns in the Internet era. We show that a user's precise location from hashtags can be inferred with a very high accuracy. To address this threat, we introduce a system called Tagvisor that systematically suggests alternative hashtags if the user-selected ones constitute a threat to location privacy. Tagvisor realizes this by means of three conceptually different obfuscation techniques and a semantics-based metric for measuring the consequent utility loss. Our findings show that obfuscating as little as two hashtags already provides a near-optimal trade-off between privacy and utility in our dataset. This in particular renders Tagvisor highly time-efficient, and thus, practical in real-world settings.

Meanwhile, we noticed that most previous works in privacy of OSNs focus on a restricted scenario of using one type of information to infer another type of information or using only static profile data such as username, profile picture or home location. However the multimedia footprints of users has become extremely diverse nowadays. In reality, an adversary would exploit all types of information obtainable over time, to achieve its goal. The second part of this dissertation analyses OSN privacy by jointly exploiting long term multimodal information. We focus in particular on inference of social relationships. We consider five popular components of posts shared by users, namely images, hashtags, captions, geo-locations and published friendships. Large scale evaluation on a real-world OSN dataset shows that while our monomodal attacks achieve strong predictions, our multimodal attack leads to a much stronger performance. Our results highlight the need for multimodal obfuscation approaches towards protecting privacy in an era where multimedia footprints of users get increasingly diverse.

In addition to OSNs, another technology that the modern society has closely embraced which also generates a huge amount of user data is wearable physical activity tracking devices. These devices collect health data that is privacy sensitive, however they had not yet received the necessary attention from privacy researchers. In the third part of this dissertation, we performed the first systematic study of privacy risks arising from actimetry data. In particular we focus on attribute inference for gender, age and education and temporal linkability of users. We demonstrate the severity of the privacy attacks by an extensive evaluation on a real life dataset and derive key insights.

While studying social networks, we noticed that often the machine learning algorithms that we used for our analysis were very sensitive to any bias in the training data. We had been using the power of the popular tool "graph embeddings", to analyze social networks. However, no prior works had studied potential bias issues inherent within graph embeddings. The last and final part of this dissertation explores the direction of fairness of machine learning algorithms. Specifically, we analyze the fairness of node2vec, a popular graph embedding method. Our analyses on two real-world datasets

demonstrate the existence of bias in node2vec when used for friendship recommendation. We proposed a fairness-aware embedding method, namely Fairwalk, which extends node2vec. Experimental results demonstrate that Fairwalk reduces bias under multiple fairness metrics while still preserving the utility.

2
Related Work

Here, we present the previous research in the related areas of privacy in Online Social Networks(OSNs), wearable devices as well as a brief summary of the field of Algorithmic Fairness. We also explain how they relate to or motivate our work.

2.1 Location inference and hashtags

Location privacy. Shokri et al. [102] propose a comprehensive framework for quantifying location privacy. This framework is able to capture various prior information available to an attacker and attacks the attacker can perform. The authors of [102] also propose several privacy metrics, which inspire the metrics used in this paper. Ağır et al. [4] further analyze the impact of revealing location semantics, such as restaurant or cinema, by extending the underlying graphical model used for location inference in [102]. Experimental results on a Foursquare dataset demonstrate that location semantics can raise severe privacy concerns. Bilogrevic et al. [10] concentrate on the utility implications of one popular privacy-preserving mechanism, namely generalization (both geographically and semantically). Through the data collected from 77 OSN users, their model is able to accurately predict the motivation behind location sharing, which enables design of privacy-preserving location-sharing applications. Other interesting works in this field include [84, 95, 96]. Contrary to these previous works, ours focuses on the location-privacy risks of hashtag sharing, and propose mechanisms to mitigate these risks.

Hashtag analysis. Highfield and Leaver [46] point out the research challenges for Instagram, especially in comparison to Twitter, by examining hashtags in the two OSNs. Based on a dataset collected from Instagram users' profiles and hashtags, Han et al. [40] identify teens and adults using popular supervised learning techniques and additionally reveal significant behavioral differences between the two age groups. The authors of [104] use #selfie in Instagram to study the phenomenal ubiquitous convention of self-portrait, while the authors of [71] study the health implications and obesity patterns associated with the use of #foodporn. More recently, Lamba et al. [63] study #selfie in Twitter that can potentially cause life threat. Although no previous work has addressed privacy issues arising out of hashtags to the best of our knowledge, the aforementioned works have been very influential in shaping our approach of using hashtags for training our attacker model.

Location prediction in OSNs. Multiple works aim at estimating a user's home location using the posts he has published in OSNs. Cheng et al. [18] identify words in tweets with a strong local geo-scope and estimate several possible locations for each user with decreasing probabilities. Chandra et al. [17] predict city-level user location relying on a communication model in Twitter's conversation. Jurgens et al. [55] recently provide a survey on the related works in this direction. Different from these studies, we address the problem of identifying the precise location of users' posts.

Some other works concentrate on estimating the location of each post, which is closer to the problem we address. Li et al. [64] use KL-Divergence to build a language model for each location. Due to the insufficient number of tweets for each location, they incorporate texts in web pages returned by search engines to support their model.

Agerwal et al. [3] also use geographic gazeteer data as vocabulary to identify location names from phrases in tweets. The approach of Dalvi et al. [24] assumes that a geo-located object is mentioned in the tweet. Our attack however does not need any external knowledge and already shows a high accuracy in predicting the exact point of interest. Moreover, we propose a tool for privacy-preserving hashtag sharing.

2.2 Friendship inference from multiple modalities

We present literature review in each of the 5 components of OSN data that we consider namely, link prediction, locations, text, images and hashtags followed by research on OSN data in general.

Link Prediction. Liben-Nowell et al. [65] were among the earliest to formalize the link prediction problem in social networks. Since then, link prediction has been extensively studied [36], [88], [112], [67], [32], [76] We exploit an advanced deep-learning based approach node2vec [38] which is shown to outperform other approaches. Other lines of work use social links for attribute inference such as Gong et al. [34] infer attributes such as major, employer, and cities lived from friendship and behavioural links. We, however, try to infer social links from 5 types of user generated content.

Location. Early works such as [29], [22], [92], [100], [111] infer social ties from co-occurrence in time and space and require two users to share common friends or locations. Olteanu et al. [81] study the effect of co-location information on location privacy. More recently, Zhou et al. [122] infer social links from friendships and mobility data. Finally, Backes et al. [8] take a deep learning approach to learn users' mobility features and use them for social relationship inference. We adopt this approach for our location based attack since it best fits our user and adversary model, and is shown to significantly outperform prior approaches [92], [100], [111].

Text. The growth of textual data in OSNs has been exploited by a plethora of advances in NLP and text mining. Kim et al. [61] study the influence of offline friendship on tweeting behavior, Adamic et al. [2] proposed a matchmaking algorithm that leverages text, mailing list, and in and out-link information to analyze the structure of a social network. Preotiuc-Pietro et al. [94] study use language features like unigrams, word clusters and emotions to predict political ideologies of users. Tan et al. [107] characterize relations between ideas in texts using co-occurrence within documents and prevalence correlation over time. However friendship inference from textual content has not yet been studied to the best of our knowledge.

Images. Shoshitaishvili et al. [103] combine facial recognition, and spatial analysis to determine user pairs that are dating amongst people represented in a large corpus of pictures shared on a social network. Ilia et al. [50] propose an access control mechanism that allows users to manage the visibility of their face in photos published by other users. Research on defenses also mainly focuses on obfuscation of faces such as person image generation [69], face replacement [106], redaction [82] etc. Recent work by Orekondy et al. [82] focuses on localized segments containing text-like attributes such as license plate, datetime, name, etc . However these approaches are tested only on small self-annotated datasets and their applicability to friendship inference in large scale real life datasets is

yet to be explored.

Hashtags. Highfield and Leaver [47] examine research challenges of working with hashtags in Instagram in comparison to Twitter. Han et al. [41] use Instagram and hashtag data to identify teens and adult age groups via behavioral differences between them. Using the hashtag #selfie, authors of [104] study the collective behavior of sharing selfies on Instagram and [63] study the potential dangers of self portraits. Authors of [71] study the association between emotion and health with the hashtag #foodporn. Prior work on privacy implications of hashtags studies location privacy using hashtags and friendship inference [P1]. A recent work [117] uses a user-hashtag bipartite graph embedding model to infer friendships, but are hugely outperformed by our method.

OSNs and Privacy. There has been extensive research on OSN user deanonymization, based on the network topology [77] or using metadata of tweets [90]. However the adversary needs to already have access to the true identity of the target users alongwith metadata of their older tweets for the latter work. Works on cross-site profile matching such as [33, 66] utilize only user profile metadata such as name, profile picture, home location etc. In contrast, we focus on user generated content, i.e. the multimedia footprint of users built up over time from the data they post everyday, since this collectively gives a faithful reflection of the user's interests. Liu et al. [66] do consider tweets posted over time as well as friendships but they perform cross platform linkability. We address a different problem, i.e., friendship inference and additionally include hashtags, images, location check-ins as well.

2.3 Fairness and graph embedding

Algorithmic bias has received a lot of attention in the recent years [9, 98]. Prior work distinguishes between group unfairness and individual unfairness. Group unfairness (disparate impact) occurs when the decision outcomes disproportionately benefit or hurt people of different groups based on their sensitive attribute value. Examples of group fairness are statistical parity [30], disparate mistreatment [115], equality of opportunity [42] and calibration [62]. Individual unfairness occurs when the decision of an algorithm is different for two individuals having the same non-sensitive attributes but different values for the sensitive attribute [28].

Different fairness-aware algorithms have been proposed to achieve group and individual fairness, mostly for predictive tasks. [16] extend naive Bayes models by modifying the learning algorithm. [56] modify the entropy-based splitting criterion in decision tree induction to account for sensitive attributes. [57] apply a regularization to probabilistic discriminative models, such as logistic regression. Other works include fairness in OSN timelines [43] and extension of group fairness to fair rankings, [116, 113]. Contrary to the above, we focus on obtaining fair features that can later be used for plethora of machine learning tasks. We tackle group fairness and not individual fairness since two individuals that have identical neighborhoods would have similar embeddings, independent of the sensitive attributes of their neighbors and would be treated equally.

In the area of graph embeddings, after invention of word2vec [73], many prominent graph data mining techniques were developed adopting the skip-gram model, namely

DeepWalk [91], node2vec [38], LINE [109] and PTE [108], and walk2friends [8]. In this paper we study the fairness of node2vec - the most widely used one*.

2.4 Privacy of step count data

It has been shown, that it is possible to fingerprint Android devices by accessing the accelerometer API [12, 27] from a website opened on the phone. Bittel et al. [11] shown that iPhone's accelerometer is just as accurate and precise as biomedical measurement devices. Accelerometer data from a phone carried in a pants pocket is sufficient to predict owner's activity (e.g. sitting, standing, walking, running) [39, 5, 89]. Accelerometer sensor data reveals how users hold and touch smartphones by hand, based on which Davarci et al. [25] perform child/adult detection.

The potential benefits of activity tracking for health and well-being has received a lot of attention from both academia and industry. Shameli et al. [101] study how competitions affect physical activity using a dataset of competitions within the Argus smartphone app. Althoff et al. [7] study activity distribution from smartphones of 717,527 people across 111 countries. Using the same dataset, Pierson et al. [93] use Cyclic Hidden Markov Models to detect and model activity cycles.

However, few have studied users' behaviors when sharing such personal fitness data and the privacy concerns that arise from the collection, aggregation, and sharing of this data. Vitak et al. [110] highlight the relationship between users' demographics, data sharing behaviors, privacy concerns, and internet skills. Hassan et al. [44] demonstrate an attack against Endpoint Privacy Zones, that infers users' protected locations from their information in public posts using activity data collected from the Strava app. Meteriz et al. [72] predict the location trajectory of users from publicly available elevation profiles. Nguyen et al. [78] demonstrate high accuracy location tracking just through smartphone accelerometer and magnetometer's footprints.

*according to citation count from GoogleScholar

3
Tagvisor

A Privacy Advisor for Sharing Hashtags

3.1 Motivation

Online Social Networks have now become an inseparable part of people's lives and introduced several new terms into the vocabulary of the modern society. "Hashtag" is one of these terms that has emerged as a widely used concept of popular culture and campaigns. Defined as a word or non-spaced phrase preceded by the hash character #, hashtag was created to serve as a metadata tag for people to efficiently search for information. Originally created on Twitter, hashtags have been widely adopted in many areas. Media campaigns are increasingly using hashtags to attract and engage customers (e.g., #ShareACoke). Hashtags also enable people to stay updated on trending news stories. For instance, #imwithher has been extensively used during the US 2016 election.

Hashtags have seen significant usage in online communication recently. Users have created many hashtags to convey meanings which previously did not exist in the English vocabulary. For instance, #like4like indicates a promise of the users to like back the posts of those who like their post in OSNs [118]; #nofilter means the associated photo has been posted directly as taken, without being edited. The large amount of hashtags provide us with an unprecedented chance to understand the modern society, and researchers are increasingly leveraging hashtags to conduct their study [104, 71]. While bringing significant utility benefits, hashtags may carry sensitive private information about people using them. However, privacy threats arising out of hashtags have been largely overlooked.

In this work, we address privacy raised by hashtags. Among all the private information, we concentrate in particular on mobility information, which is recognized as one of the key privacy concerns in the modern society [49, 85, 10, 80]. In fact, location arguably even constitutes the most sensitive information being collected by service providers [75] (e.g., being able to infer that a user is staying at a hospital can severely threaten his privacy), and it can be used to infer additional sensitive information such as friendship [99, 8] and demographics [120, 86]. A user can disclose his location information through many different ways such as sharing location in OSNs (often referred to as *check-in*) and allowing mobile applications to collect their geo-coordinates through GPS sensors in the smartphone. Many users have recognized the inherent risks towards their location privacy and abstain from explicitly sharing their locations. However, hashtags may also leak the exact location of a user, which is the primary focus of the first part of this dissertation paper.

3.2 Contributions

Our contributions are two-fold:

1. A novel inference attack for uncovering a user's fine-grained location based on posted hashtags, along with a comprehensive evaluation on a real-life OSN dataset,

2. A privacy-enhancing system, namely Tagvisor, that uses various obfuscation techniques for thwarting the aforementioned attack.

3.2.1 Attack

Our inference attack aims at predicting the fine-grained location of a user's post given its hashtags by learning the associations between hashtags and locations. The attack relies on a random forest classifier and achieves remarkable accuracy.

We empirically evaluate our attack based on a comprehensive dataset of 239,000 Instagram posts with corresponding locations and hashtags from three of the largest English-speaking cities (New York, Los Angeles and London). The locations we predict are the exact points of interest (POI) within each of the three cities, such as *Land of Plenty*, a Chinese restaurant in New York. The experiments show that our hashtag-based location-inference attack achieves an accuracy of 70% in New York (for a number of around 500 considered locations) and 76% in Los Angeles and London (for around 270 and 140 locations, respectively). We also compare our classification method with other popular algorithms such as support vector machine and gradient boosting machine, and show that our classifier outperforms them by at least 7%. Furthermore, we empirically identify the number of hashtags maximizing the attack success to be seven. Additionally, we evaluate our attack on a global level without prior knowledge of the city, and demonstrate that it can still reach an accuracy of more than 70%. Finally, by considering two types of attackers, one with prior knowledge on the targeted users' hashtags and locations, and the other without, we show that the accuracy drops by around 20% for the latter attacker. This demonstrates that the adversary can enhance his model by learning per-user associations between hashtags and locations.

We stress that our work is not intended for users who intentionally use hashtags to reveal their locations, for example, when a user publishes a post with #statueofliberty as one of the hashtags. Instead, our work aims at helping users that might *unintentionally* disclose where they are. The examples below, taken from the evaluation of the inference attack, illustrate this point further. A user's post shared with hashtags #music, #musicphotography, #soul, #fujifilm, #thephotoladies, and #pancakesandwhiskey may not reveal anything significant about the location to most individuals. Our attack however correctly predicts that the user is at the *Highline Ballroom*, a music venue in New York. In another example from Los Angeles, a user sharing #fighton #trojans #band #ftfo is correctly located by our attack at *University of Southern California*.

3.2.2 Defense

To counter the aforementioned privacy violations, we develop Tagvisor: a system that provides recommendations, e.g., to an OSN user who wants to share hashtags without disclosing his location information. In particular, Tagvisor implements three different obfuscation-based mechanisms: hiding (a subset of) hashtags, replacing hashtags by semantically similar hashtags, and generalizing hashtags with higher-level semantic categories (e.g., Starbucks into coffee shop). Tagvisor returns an optimal subset of obfuscated hashtags that, at the same time, guarantees some predefined level of location privacy and retains as much utility as possible. To accurately quantify utility, we rely on an advanced natural language processing model, namely word2vec [73, 74].

The empirical evaluation of our privacy-enhancing system shows that:

- the replacement mechanism outperforms both hiding and generalization methods.

- the higher number of original hashtags, the better the utility for similar levels of privacy,
- regardless of the original number of hashtags, obfuscating two hashtags provides the best trade-off between utility, privacy, and time efficiency.

The latter finding notably demonstrates the practical feasibility of our privacy-preserving system given the computational capabilities of current mobile devices.

3.3 Model

In this section, we describe the user and adversarial models, as well as the privacy metrics used throughout the paper.

3.3.1 User Model

We use two sets U and L to represent users and locations, respectively. A single user is denoted by u and a location (POI) is denoted by ℓ. All users' posts are in set P, and a single post $p \in P$ is defined as a quadruplet $\langle u, \ell, t, H_p \rangle$: semantically, it means u shares the post p at ℓ (checks in at ℓ) at time t, and $H_p = \{h_p^1, h_p^2, \ldots, h_p^m\}$ contains all hashtags associated with the post. In addition, $H_p \subseteq H_\ell \subseteq H$ where H_ℓ represents the entire set of hashtags shared (in posts) at location ℓ and H is the set of all hashtags in our dataset.

3.3.2 Adversarial Model

In this work, we consider an adversary who intends to infer the hidden location ℓ of a post p by observing the set of hashtags H_p published with the post. The adversary can be anyone who is able to crawl the data from OSNs such as Facebook, Twitter and Instagram. Moreover, it could also model a service provider which does not have access to all its users' real locations.

In order to accurately infer hidden locations based on hashtags, the adversary needs to build an accurate knowledge K of associations between hashtags and locations. In this work, we assume he learns these associations based on previously disclosed posts that included both hashtags and location check-ins. This learning phase is explained in Section 3.4. The associations can be learned in different ways. We consider here two different adversarial models. The first adversary, namely A_1, uses all the publicly shared posts he is able to collect, including those shared by the targeted users (users whose posts' locations the adversary intends to infer). In other words, in this model, we assume the targeted users have already shared some of their posts with both hashtags and check-ins. In the second model, the adversary (A_2) does not have access to any previous location-hashtag data shared by the targeted users. This models the case when the targeted users are particularly privacy-cautious [79] and do not share any location with their posts. In this case, the adversary has to rely only on the location and hashtag data shared by other users to build his knowledge K.

The output of our hashtag-based location inference attack on a post p is the posterior probability distribution over all locations $\ell \in L$ given the set of observed hashtags H_p:

$$\Pr(\mathcal{L}_p = \ell | H_p, K), \tag{3.1}$$

where $\mathcal{L}_p$ is the random variable representing the location of post p. Based on this posterior distribution, the most likely location is:

$$\ell_p^* = \arg\max_{\ell \in L} \Pr(\mathcal{L}_p = \ell | H_p, K). \tag{3.2}$$

Equation 3.2 is the core concept behind our inference attack, and we will describe its instantiation in detail in Section 3.4.

3.3.3 Privacy Metrics

We measure the location privacy of a user who shares a post p and its corresponding hashtags H_p by using the expected estimation error, as proposed in [102]. Formally, given the posterior probability distribution $\Pr(\mathcal{L}_p = \ell | H_p, K)$, the location privacy is defined as:

$$\sum_{\ell \in L} \Pr(\mathcal{L}_p = \ell | H_p, K) \cdot d(\ell, \ell_p^r), \tag{3.3}$$

where ℓ_p^r is the real location of the user sharing the post p, and $d(\cdot, \cdot)$ denotes a distance function. If the geographical distance $d_g(\ell, \ell_p^r)$ is used, we will generally refer to this metric as the *expected distance*. In this paper, we rely on the haversine distance to measure the geographical distance. If the binary distance, defined as

$$d_b(\ell, \ell_p^r) = \begin{cases} 0 & \text{if } \ell = \ell_p^r \\ 1 & \text{otherwise} \end{cases} \tag{3.4}$$

is used, Equation 3.3 is then equal to $1 - \Pr(\mathcal{L}_p = \ell_p^r | H_p, K)$ and is referred to as the incorrectness [102]. We also use the *correctness* (to quantify the performance of the inference attack), which is the opposite of the incorrectness, and is simply equal to $\Pr(\mathcal{L}_p = \ell_p^r | H_p, K)$. We further use the *accuracy*, defined in the machine learning context, as another privacy metric. Formally, accuracy is defined as $1 - d_b(\ell_p^*, \ell_p^r)$, where $d_b(\cdot, \cdot)$ is the binary distance defined above. We can alternatively use the inaccuracy, i.e., $d_b(\ell_p^*, \ell_p^r)$. Note that, depending on the context, the accuracy can refer to one sample or multiple. In the latter case, the accuracy represents in fact the *average* accuracy over all samples (e.g., from a testing set).

3.4 Inference Attack

Our location inference attack relies on a random forest classifier. We encode the presence of a hashtag h_i in a post p as a binary value x_p^i, being equal to 1 if it is published with the post, and 0 otherwise. Each post p with its set of hashtags H_p at location ℓ can be represented by a feature vector $\vec{x}_p = (x_p^1, \ldots, x_p^n)$, where $n = |H|$, and by the label

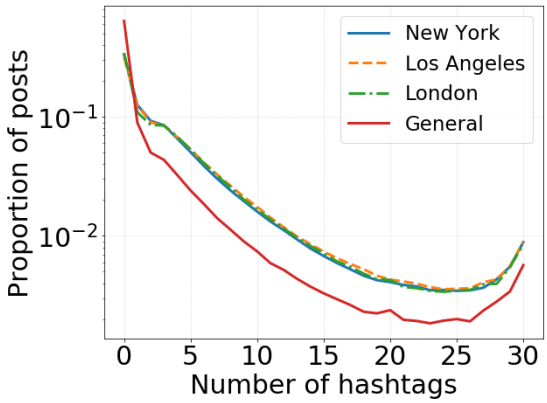


Figure 3.1: Distribution of the proportion of posts with a certain number of hashtag(s), from 0 to 30 hashtags.

Table 3.1: Pre-processed data statistics.

	New York	Los Angeles	London
No. of posts	144,263	61,767	34,018
No. of hashtags	8,552	4,600	2,395
No. of users	3,911	1,625	992
No. of locations	498	268	141

or class y_p, where $y_p = \ell_p^r$. The length of the feature vector is thus equal to the total number of unique hashtags in the entire dataset.

For training the random forest, the adversary uses as input the samples $(\vec{x}_p, y_p)\ \forall p \in P_{\textbf{train}}$ where $\vec{x}_p = (x_p^1, \ldots, x_p^n)$ and $P_{\textbf{train}}$ refers to posts in the training set that contains the auxiliary knowledge of the adversary. The adversary carries out his attack on the set $P_{\textbf{tar}}$ by trying to classify each sample post $p \in P_{\textbf{tar}}$ using the features $\vec{x}_p$ on the trained forest. Among several ways of obtaining class probabilities in an ensemble of decision trees [13], we employ averaging of the votes of all the trees of the forest. The total number of trees that vote for each class is divided by the total number of trees to obtain the posterior probability distribution $\Pr(\mathcal{L}_p = \ell | H_p, K)\ \forall \ell \in L$. This outcome is used to quantify privacy with the metrics defined in Section 3.3.3.

3.5 Dataset

We collect our experimental data from Instagram, one of the most popular OSNs and the major platform for hashtag sharing. Besides, Instagram users have been shown to share their locations an order of magnitude more often than in Twitter [70], and Instagram's localization function was linked with Foursquare, which enabled us to enrich each location with name and category, not just geograpical coordinates as in many existing datasets [20]. The category information is in fact a building block for one of our defense mechanisms, namely the semantic generalization (Section 3.7).

Our data collection was conducted in January 2016. We concentrate on three of

Table 3.2: Performance of location inference across all the cities for different adversary models and baseline.

	Correctness			Expected distance (km)			Accuracy		
	A_1	A_2	baseline	A_1	A_2	baseline	A_1	A_2	baseline
New York	0.613	0.468	0.015	0.917	1.272	4.198	0.697	0.556	0.053
Los Angeles	0.685	0.502	0.015	1.870	3.046	11.275	0.758	0.597	0.048
London	0.686	0.552	0.020	0.857	1.575	4.518	0.761	0.617	0.051
All cities	0.624	0.465	0.010	211.471	345.980	3563.082	0.712	0.560	0.045

the largest cities in the English-speaking world, namely New York, Los Angeles and London. In the first step, we rely on the Foursquare API to crawl all the location IDs in the three cities, and extract these locations' names and categories. Then, we use the function `/locations/search` provided by Instagram's public API to map the previously obtained Foursquare's location IDs to their corresponding Instagram's location IDs. Finally, for each Instagram location, we collect its publicly available posts' metadata (user ID, time and hashtags) in the second half of 2015 (2015.7.1-2015.12.31).

As in [20, 99, 119, 87, 8], we perform the following filtering operations on our collected data.

- We filter out user accounts whose number of followers are above the 90th percentile (celebrities) or under the 10th percentile (bots). To address the data sparseness issue, we also filter out users with less than 20 check-ins in each city.
- We filter out hashtags appearing less than 20 times in each city, following a classical filtering strategy used by the natural language processing community.
- For each city, we concentrate on locations with at least 50 check-ins. This helps us filter out the locations that are rarely visited by people while still ensuring that we are not only concentrating on the most famous places. Knowing a user being at the considered locations still threatens his privacy to a large extent. Additionally, from a supervised learning point of view, we need sufficient data for each class (each location) to train a robust classifier, such that a meaningful attack can be conducted.

After the filtering, we obtain 144,263 posts in New York, 61,767 posts in Los Angeles and 34,018 posts in London (Table 3.1). Despite the filtering strategies, we are still left with a large dataset with substantially more users and fine-grained locations compared to previous works. For instance, the dataset used in [102] contains 20 users' mobility data in a 5×8 grid over the city of San Francisco.

Figure 3.1 displays the distribution of the number of hashtags per post over our entire dataset. For all three cities, there are around 40% of posts without any single hashtag, between 11 and 14% with one hashtag, 8-9% with two hashtags, and 8% with three hashtags. The general curve (in red) represents the distribution of number of hashtags for posts without location check-in. The latter dataset is collected following the same methodology as in [104]. We randomly sample over 10 million user IDs on Instagram and collect all their published posts' hashtags. In the general set of posts,

there are 46% of the posts that include at least one hashtag. This shows that, almost half of the posts could be targeted by our hashtag-based location inference attack. The similar trend between the general curve and those with location information shows that users who do not reveal their location may nevertheless disclose hashtags that could then be used by an adversary to infer their locations. As expected, the distribution drops quite fast with an increasing number of hashtags, for both datasets with and without location information. The small increase close to 30 is due to Instagram's upper bound of 30 hashtags per post.

3.6 Attack Evaluation

In this section, we present the evaluation results of our location inference attack.

3.6.1 Experimental Setup

As described in Section 3.4, our attack relies on a random forest classifier, and we set the number of trees for the random forest to be 100 following common practice [83].

To comply with the different background knowledge of A_1 and A_2, we split the dataset as the following. For adversary A_1, we randomly assign 80% of the posts for training and 20% for testing. For adversary A_2, we split all users randomly: posts from 80% of the users are put into the training set and posts from the other 20% are in the testing set. The random split is repeated 10 times, and we report the average results.

3.6.2 General Inference Results

3.6.2.1 Adversaries

We present the performance of our location inference attack in Table 3.2. In each city, our attack achieves a strong prediction. For instance, the accuracy is around 70% for the New York dataset under adversary A_1, which indicates the attacker is able to predict the exact locations (out of 498) of 20,196 posts (out of 28,852 testing posts). The small expected distance in all cities further indicates that even when the prediction is wrong, the attacker is still able to narrow down the target's location into a small area. The performance of adversary A_2, however, drops in all the cities. This is because adversary A_2 has no prior knowledge of the hashtag-location association of the targeted users. Nevertheless, adversary A_2 still achieves a relatively high prediction success (e.g., the correctness is above 0.55 in London) showing that learning per-user associations between hashtags and locations is helpful but not absolutely needed.

3.6.2.2 Cities

Both adversaries achieve the strongest prediction in London, followed by Los Angeles and New York. The reasons are manifold, ranging from cultural differences to hashtag usage. We notice that New York has the largest number of locations (498), i.e., the highest number of classes for the random forest model, thus making the classification the most difficult. Meanwhile, even though both adversaries in Los Angeles achieve

the second highest accuracy and correctness, they perform the worst in the expected distance metric. This is due to the different densities of the three cities: Los Angeles covers a larger area with places being more uniformly distributed in the geographical space than New York and London.

3.6.2.3 Baseline

To demonstrate that our high prediction performance is not due to the skewed distribution of the location check-ins, we further establish a baseline model that relies only on the locations' distribution (in the training set) to predict a targeted user's location. The adversary infers that the user is at the most likely location, i.e., the location with most check-ins in the training set, independently of the hashtags. As depicted in Table 3.2, both A_1 and A_2 achieve at least a 10-time higher accuracy than the baseline, and a 27-time higher correctness than the baseline. We observe similar results when the adversary targets all cities at the same time. As for the expected distance metric, the baseline is outperformed by 3 to 5 times depending on the considered adversary and city.

3.6.2.4 Other classifiers

Besides random forest, we further experiment with two other classifiers: gradient boosting machine (GBM) and support vector machine (SVM). For GBM, we observe that 100 trees provide the best accuracy, and for SVM, we use the radial basis function kernel. GBM's accuracy is around 7% worse than random forest in all the cities, and SVM achieves the worst performance with accuracy in London of only 0.15. This suggests that random forest is the most suitable algorithm for learning the association between hashtags and locations.

3.6.3 Privacy vs. Number of Hashtags

Next, we study the relation between the number of shared hashtags and the prediction performance, i.e., how many hashtags are necessary to determine where a post is shared. To this end, we group the posts in the testing set by the number of hashtags they contain and compute the correctness for each group.

Figure 3.2 depicts the average correctness for the two adversary models (results for accuracy follow a very similar trend). As we can see, for posts appended with only one hashtag, the prediction performances are relatively weak in all the cases since one hashtag does not provide enough information. However, when the number of hashtags increases to two, we observe significant increases under both adversary models. Especially for A_1, the correctness is increased by more than 40% for all the cities. This shows that hashtags contain highly location sensitive information. The performance of both adversary models is strongest when the targeted post contain around 7 hashtags. Beyond this number, we observe an interesting difference. On one hand, the average correctness of adversary A_1 remains stable for posts with more hashtags. We can observe a small decrease only for London due the relatively smaller size of testing data which probably creates more noisy results. On the other hand, the performance of adversary

A_2 decreases for all cities when it faces posts with increasing number of hashtags. This happens because, by including more hashtags, a user confuses the classifier which has no information about the past activity of the user, as A_2 has not built user-specific associations between hashtags and locations. Therefore, a user who has never shared any locations (the assumption of A_2) is less vulnerable to the attack. This, along with the conclusions drawn from Table 3.2 that A_2 performs worse than A_1 in general, further suggests that a cautious user is always rewarded in matters of privacy.

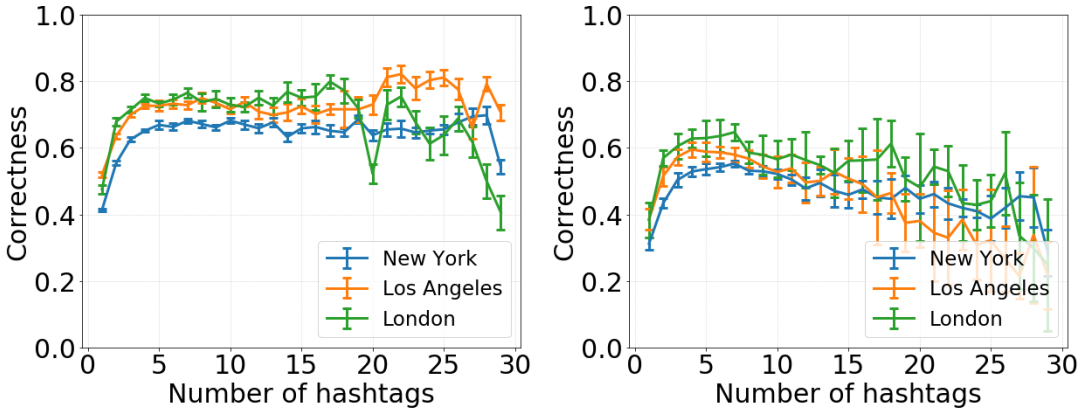


Figure 3.2: Correctness of adversary A_1 and A_2 with respect to the number of hashtags shared in posts.

3.6.4 Privacy across All the Cities

So far, our inference attack is conducted at the city level. However, it is also interesting to see whether our attack can be generalized to the global level: Given a post with hashtags, can we predict where it is published among all the locations in New York, Los Angeles and London. To this end, we combine the datasets from the three cities together and perform the aforementioned split with respect to the two adversary models for training and testing. As shown in Table 3.2, the performance of the global level attack drops compared to the city level attack, especially for London and Los Angeles. However, the attack is still rather effective, e.g, adversary A_1 achieves 0.712 accuracy and 0.624 correctness. Meanwhile, the expected distance grows much larger, which is mainly due to the misclassification among different cities. However, considering the actual distances between the three cities, this expected distance is still very small, and both A_1 and A_2 perform much better than the baseline model.

Through the above experiments, we have demonstrated the severe privacy threat carried by hashtags. In the next section, we propose a first of its kind privacy advisor which we coin Tagvisor, that uses three defensive mechanisms to provide users with advice when sharing hashtags.

3.7 Countermeasures

In this section, we present *Tagvisor*, a privacy advisor that is based on different obfuscation mechanisms for enhancing the user's privacy and preserving as much semantic utility as possible. We first introduce the different components of Tagvisor. Then, we describe our approach for measuring hashtags' utility. Finally, we present the three obfuscation mechanisms.

3.7.1 Tagvisor

Tagvisor resides on the user's device and, whenever the user wishes to share a post with some hashtags without revealing his location, Tagvisor tests whether its classifier can correctly predict the post's location, i.e., whether the user's location privacy is compromised. If so, Tagvisor suggests alternate sets of hashtags that result in an enhancement of privacy without significant utility loss. The system can also suggest the optimal set of hashtags that minimizes the utility loss while providing a certain level of location privacy.

Formally, the Tagvisor's defense mechanism can be interpreted as an optimization problem. For each new post p that a user wants to share with an original set of hashtags H_p, Tagvisor suggests the optimal set of hashtags H_p^* as follows:

$$H_p^* = \underset{H_p' \in \mathcal{H}_p}{\arg\min} \Delta_{H_p, H_p'} \quad \text{subject to} \quad LP_{H_p'} \geq \alpha. \tag{3.5}$$

Here, $\Delta_{H_p,H_p'}$ represents the utility loss, quantified by the semantic distance between H_p and H_p'. $LP_{H_p'}$ represents the privacy level for the set H_p', which can be quantified with any of the metrics defined in Section 3.3, and α is the minimal privacy level desired by the user. For example, a simple and intuitive profile would be to set $\alpha = 1$ with LP measured by $d_b(\ell_p^*, \ell_p^r)$, the inaccuracy. This requirement ensures that the user only shares hashtags that do not enable the attacker to infer his exact location ℓ_p^r.

$\mathcal{H}_p$ can be as big as the power set of H. In order to keep complexity at a reasonable level, we can restrain it to a smaller subset depending on the different protection mechanisms and the original set of hashtags H_p. The user can also set a maximum number of hashtags that the privacy mechanism can modify. We evaluate these more practical constraints in Section 3.8.

Tagvisor consists of four main blocks. Figure 3.3 shows the main components of the Tagvisor system. R, H and G in Recommender represents obfuscation by Replacement, Hiding and Generalization, respectively. If the privacy is preserved with the original set of hashtags chosen by the user, they are directly published (1:YES). Otherwise, they go to the obfuscator (1:NO) which then sends the modified sets to the privacy quantifier.

3.7.1.1 Privacy quantifier

The *privacy quantifier* contains the *trained classifier*, i.e., our random forest model, and a *condition checker*. The *trained classifier* is regularly updated with new knowledge from the external *classifier trainer*. This classifier trainer uses the data K to constantly improve its random forest model. It could be running on the user's personal computer, or

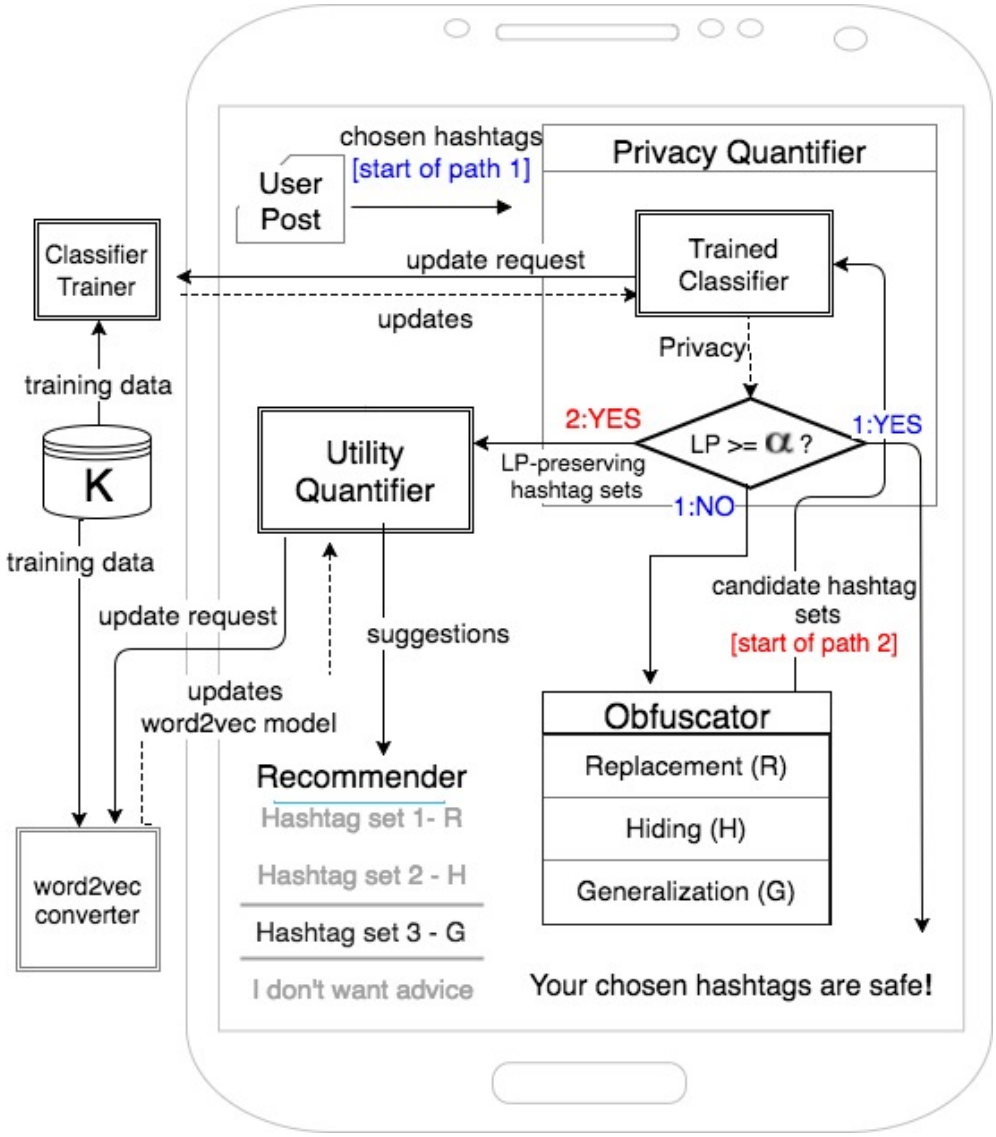


Figure 3.3: Main building blocks of Tagvisor

deployed by a non-governmental organization such as the Electronic Frontier Foundation. The *condition checker* checks whether the level of location privacy of the user (using the classifier output) is above the threshold α. If not, the user chosen hashtags are sent to the *obfuscator* for determining alternative hashtag sets. Among these suggested sets of hashtags, those for which the privacy condition in the *condition checker* is satisfied are then fed to the *utility quantifier*.

3.7.1.2 Obfuscator

The *obfuscator* consists of three components that represent the different obfuscation mechanisms: *hiding, replacement* and *generalization*. Each component produces a set

of hashtags by employing the corresponding defense mechanism. These hashtag sets are fed to the *privacy quantifier* which filters out those that do not meet the required privacy threshold α (2:YES in Figure 3.3).

3.7.1.3 Utility quantifier

The *utility quantifier* also resides on the user's device. It maintains records of the utility model and periodically updates itself by querying the external *word2vec converter* (which continuously updates itself with any new data from the knowledge database K). As will be described in Subsection 3.7.2, the *utility quantifier* uses the semantic representation of each hashtag in the set of input hashtags that it receives from the *privacy quantifier* to calculate their utility with respect to the original set of hashtags. Finally, it returns the hashtag set(s) providing minimal utility loss to the *recommender*. Depending on the user's preferences, it can return one recommended hashtag set per obfuscation method (as shown in Figure 3.3), or only one optimal hashtag set for all methods.

3.7.1.4 Recommender

The *recommender* suggests, for each obfuscation method, location-privacy-preserving hashtag sets having minimum semantic distance to the original hashtags chosen by the user. It can also suggest the optimal hashtag set among all methods, if the user does not want to make a decision by himself. It could also provide additional recommendations for a desired obfuscation method and a desired maximum number of hashtags to be obfuscated.

3.7.2 Utility Metric

The defense mechanisms should yield hashtags that retain the semantics of the original hashtags to the largest possible extent, while still defeating the attacker by misleading the random forest classifier. Removal of all hashtags, for example, reduces the attacker to a random guesser. However, for the user, the consequence is a complete loss of utility.

We rely on the semantic distance between the original set of hashtags and the sanitized set to quantify the corresponding utility loss. We capture the semantic meaning of each hashtag, and set of hashtags, by using *word2vec*, the state-of-the-art method for representing language semantics [73, 74]. word2vec maps each hashtag into a continuous vector space of d dimension,* i.e.,

$$word2vec : H \rightarrow \mathbb{R}^d, \tag{3.6}$$

using a (shallow) neural network with one hidden layer. The objective function of word2vec is designed to preserve each hashtag's context, i.e., neighboring hashtags. Therefore, if two hashtags are often shared together in posts, they will be close to each other in the learned vector space. The concept that word2vec can represent a word's semantic meaning originates from the distributional hypothesis in linguistics, which

*Following the original works [73, 74], d is set to 100 in our experiments.

states that words with similar contexts have similar meanings. For example, in our dataset, #travel and #tourist are semantically close under word2vec. We define the word2vec representation of a hashtag h as $\vec{v}_h = word2vec(h)$.

We train our word2vec model on the whole set of posts available to obtain a semantic vector for each hashtag in H. Following previous works [114, 60], we express the semantic meaning of hashtags H_p in a post p as the average of their semantic vectors:

$$\vec{m}_{H_p} = \frac{\sum_{h \in H_p} \vec{v}_h}{|H_p|}. \tag{3.7}$$

Then, the utility loss of replacing the original set H_p by H'_p is measured as the Euclidean distance between $\vec{m}_{H_p}$ and $\vec{m}_{H'_p}$:

$$\Delta_{H_p, H'_p} = ||\vec{m}_{H_p} - \vec{m}_{H'_p}||. \tag{3.8}$$

We use Euclidean distance since it is the most common method to measure distances in the word2vec generated vector space [73, 74].

3.7.3 Obfuscation Mechanisms

We now describe the three obfuscation mechanisms used by Tagvisor for protecting location privacy.

3.7.3.1 Hiding

Hiding (also referred to as deletion) simply suggests a subset of the original hashtags H_p chosen by the user. There are in total $2^{|H_p|} - 1$ such subsets. One can limit the number of hashtags to be removed by the threshold t_h to optimize the running time, or utility. All generated subsets of hashtags are sent to the condition checker to verify whether they satisfy the location privacy constraint. The subsets providing enough privacy are sent to the utility quantifier that picks the one that minimizes utility loss.

3.7.3.2 Replacement

This mechanism replaces the original hashtags by others in the set of hashtags H to mislead the adversary. In order to keep the search complexity reasonable, we have to restrain the set of potential hashtags to replace each original hashtag with. We fix a threshold t_s and focus on the t_s hashtags that are semantically closest to the original hashtag, as defined in Equation 3.8. This ensures that the set of candidate hashtags minimizes utility loss. This bounds the search space from above by $(t_s + 1)^{|H_p|} - 1$. As in the hiding mechanism, one can further reduce the time complexity by bounding the number of hashtags to be replaced by a threshold t_r similar to t_h. The generated subsets are eventually sent to the privacy condition checker and the utility quantifier.

3.7.3.3 Generalization

This mechanism generalizes each original hashtag by one representing a semantically broader category. Our evaluation only focuses on hashtags corresponding to locations in

Foursquare. Every location identifier in Foursquare is mapped to a category identifier at two levels. For example, *Harrods* has the category *department store* at the second, lower, level and category *shop* at the first, higher, level. Since not all hashtags are generalizable (e.g., #love or #instagood), we represent the subset of generalizable hashtags in a given post as $H_g \subseteq H_p$. The search space is then equal to $3^{H_g} - 1$. To reduce the time complexity, we can also fix a threshold t_g of the maximum number of hashtags to be generalized.

3.8 Defense Evaluation

In this section, we present the evaluation results of the defense mechanisms presented in the previous section. We concentrate on the results for the strongest adversary, A_1, in New York since it includes the maximum number of users and locations. The results for Los Angeles and London follow a similar trend, which we skip due to space constraints. We consider $t_s = 2$ for the replacement mechanism in order to reduce the search space, and thus improve the time efficiency. The other parameters t_h, t_r and t_g are left free and their impact is evaluated in our experiments.

We begin with the impact of obfuscating hashtags on the attack accuracy (thus privacy) with respect to each obfuscation mechanism. Then, we compare the utility (corresponding to the obtained privacy level) between the hashtags suggested by the three obfuscation mechanisms separately and the optimal global solution. Finally, we evaluate the time efficiency of Tagvisor.

3.8.1 Accuracy vs. Obfuscation Level

Figure 3.4 shows the impact of (a) hiding, (b) replacement, and (c) generalization on the attack accuracy in New York. Except for generalization, we notice that the larger the number of hashtags obfuscated, the lower the average accuracy (different curves), thus the higher the privacy, as expected. We additionally observe that, even with 10 original number of hashtags, the average accuracy with two obfuscated hashtags only is already very low (smaller than 0.2). This means that the attacker is able to correctly infer the location of only less than 20% of the sample posts. These results demonstrate the effectiveness of the hiding and replacement mechanisms. However, generalization (Figure 3.4c) cannot bound the accuracy of the attack when the number of hashtags to be shared increases. Note that the curves are less smooth in this latter case because we have more constraint on our testing set: our testing set is much smaller for this scenario since not all hashtags are generalizable.

3.8.2 Utility vs. Privacy

While the previous subsection does not consider any optimization of utility, in this subsection we assume that the user relies on the optimization problem defined in (3.5). For our experiments, we measure location privacy by the inaccuracy and set $\alpha = 1$. This means that the user wants to release a subset H'_p that leads to a misclassification of the location and minimizes the utility loss. Note that, for a very few samples in the

testing set, the adversary can still infer the correct location based solely on the locations distribution (baseline in Section 3.6), against which Tagvisor cannot help.

Figure 3.5a shows the cumulative distribution function of the minimal utility loss over all test samples, for the three obfuscation mechanisms, and the optimal one (when the optimization considers all mechanisms together). First, we observe that replacement provides close to optimal utility and is selected for the optimal solution in 85% of the cases, against 14% for hiding, and 1% for generalization. We also notice that hiding provides higher utility than generalization if the mechanisms are considered separately. We explain this result as follows: First, replacement misleads the classifier but at the same time keeps enough utility as the fake hashtags are by design chosen to be semantically close to the original hashtags. Second, generalization provides worse utility than hiding because it must modify more hashtags than hiding to reach the same privacy level, thereby increasing the semantic distance.

We further compare the absolute semantic distances given by all four approaches and the semantic distance in random pairs of posts (purple plain curve). This baseline utility curve is constructed by randomly sampling 10,000 pairs of posts in our dataset. We observe in Figure 3.5a that 90% of the privacy-preserving hashtag sets have a semantic distance smaller than 3 to the original hashtag sets for replacement whereas, for deletion and generalization, the corresponding distance is around 6, and the distance between random pairs goes up to 11. This demonstrates that we can keep a relatively fair amount of semantic utility with the replacement mechanism.

Finally, we observe in Figure 3.5b ("unbounded" curve) that the utility loss is in general smaller when the original number of hashtags is larger. This can be explained by the fact that, when we increase the number of hashtags, we have a larger set of hashtags to obfuscate. Obfuscating 2 or 3 hashtags already brings enough privacy, as shown in Figure 3.4. In other words, the relative change in the semantics of hashtags is smaller when we have a larger original set of hashtags. When the user has originally one or two hashtags with which the adversary correctly infers his location, he has to remove one or two hashtags to mislead the adversary, and thus loses most of his utility.

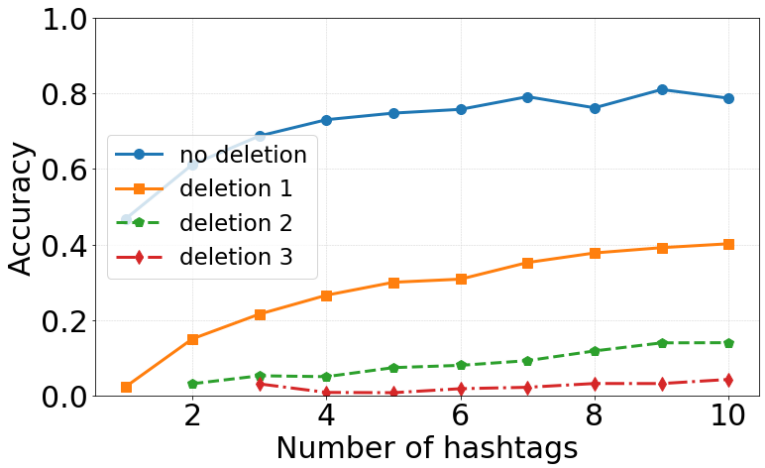
3.8.3 Time Performance

We now present the running time which is an important factor for Tagvisor's usability, and we propose solutions for trading off time efficiency with optimality. All the experiments are conducted on a laptop computer with 3.3 GHz CPU and 16 GB memory, and are implemented relying on Python packages such as pandas, numpy, and scikit-learn. It is worth noting that our experiments can be further optimized by using more efficient programming languages or libraries, and by parallelizing the computations, e.g., of the three obfuscation methods. Also, current smartphones' computing capabilities are shown to be close to those of laptops [51].

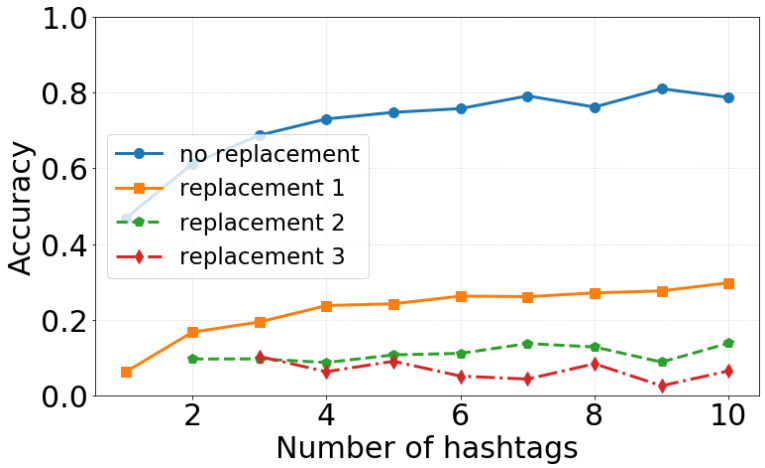
Figure 3.5c shows how the running time evolves with an increasing number of hashtags. It clearly demonstrates that the optimal solution ("unbounded" curve) is not practical when the original number of hashtags goes beyond 5. For instance, it takes more than 2 seconds for 6 hashtags. Even if most posts are shared with fewer than 6 hashtags, we want to provide a (nearly) optimal *and* efficient mechanism for users

willing to share 6 or more hashtags.

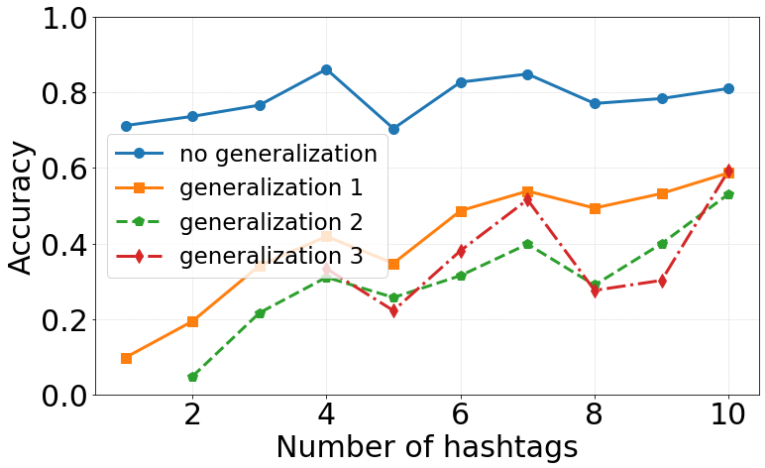
Each curve in Figure 3.5c denotes a different maximum number of hashtags that can be obfuscated. We observe that, by bounding this maximum number to two, we obtain a total running time of around one second for any number of original hashtags, which is very satisfactory. We can even reach half a second by bounding it to one hashtag. At the same time, we can observe in Figure 3.5b that the utility obtained with this maximum number bounded by one is very close to the optimal utility already. With this number bounded by two, the corresponding utility curve and the optimal one can hardly be differentiated. From a privacy perspective, even by bounding to two obfuscated hashtags, we get an average accuracy equal to 0.038, which is smaller than the baseline value (0.053) reported in Section 3.6. Only with an upperbound of one hashtag, we get an accuracy (equal to 0.21) greater than the baseline. This shows that bounding the number of possible hashtags to be obfuscated to two provides the best utility-privacy-efficiency trade-off.



(a)

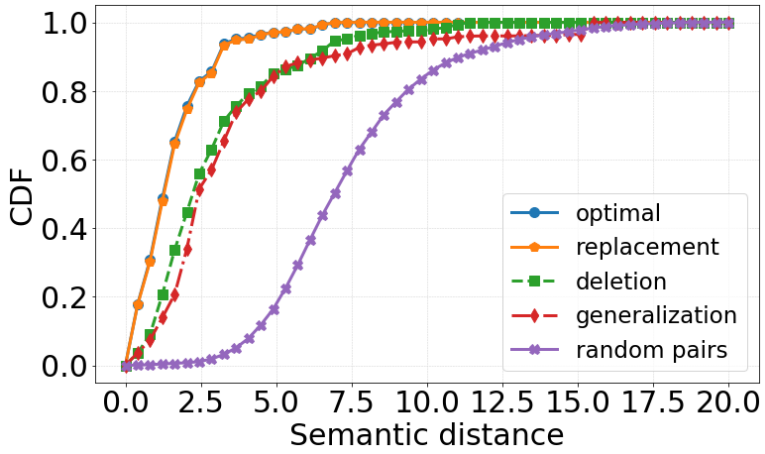


(b)

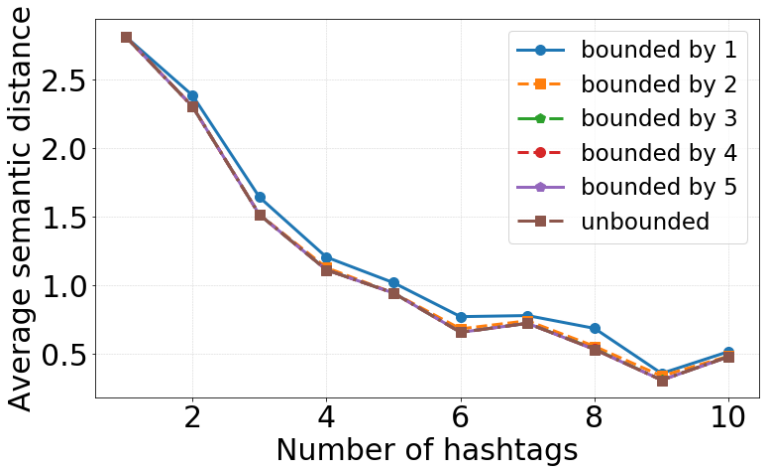


(c)

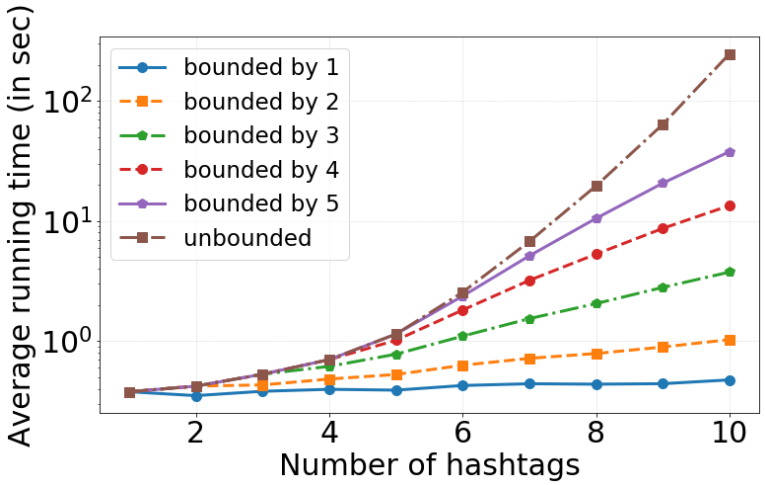
Figure 3.4: Evolution of the accuracy (A_1) with respect to different original numbers of hashtags to be shared (x-axis) and numbers of hashtags to be obfuscated (from 0 to 3) for (a) hiding, (b) replacement, and (c) generalization in New York.



(a)



(b)



(c)

Figure 3.5: (a) Cumulative distribution function (CDF) of the minimum utility loss, i.e., semantic distance, for maximum privacy constraint, of the three obfuscation mechanisms and the optimal one among all mechanisms. (b) Average minimum semantic distance of the original hashtags from the optimal solution (unbounded) and solutions with an upperbound on the number of hashtags to be obfuscated with respect to the number of original hashtags (x-axis) (c) Average running time of Tagvisor (per sample) with respect to the number of original hashtags.

3.9 Discussion

We discuss here some important details of our proposal. First of all, our attack is orthogonal to the use of computer vision techniques as it does not rely on the photo content or any visual background at all. It relies only on hashtags to make it applicable to a wider variety of settings. Of course, if the posts contain photos in addition to hashtags and the background of the photos is comprehensible, a human or advances in computer vision might sometimes have better chances of identifying the location of a photo. The evaluation of the privacy impact of such cases is left for future work.

Second, one may think that the set of sanitized hashtags shared via Tagvisor could be used by the adversary to improve his machine-learning model. Despite being intuitive, this statement does not hold. First, the location is by definition not shared with the hashtags released using Tagvisor, thus cannot be used to train the adversary's classifier. Indeed, someone uses Tagvisor if he wants to hide his location. Moreover, the location that the adversary infers via hashtags does not provide him with any extra information than what is already contained in his trained classifier. Only the new knowledge brought by users sharing both hashtags and their locations (i.e., not using Tagvisor) will be incorporated into the knowledge base K.

Third, our semantics-based utility metric does not necessarily encompass the purposes of all users. This motivated us to leave some degrees of freedom to the users in Tagvisor, as explained in Section 3.7. Concretely, the user can decide to either let the application directly suggest the optimal set of hashtags, or suggest the best sets for each obfuscation mechanism. The user can even set a minimum/maximum number of hashtags to be shared, also for each mechanism, and receive recommendations. We believe that this approach brings both usability and maximal utility for all users. A user study to analyze the acceptance of Tagvisor among OSN users would be an interesting future work.

Finally, by requiring that posts contain both hashtags and locations, our dataset may be biased towards users that are more willing to share locations. Our high accuracy in the location inference attack could be partially attributed to this bias in the testing set. However, one must realize that we cannot avoid such a restriction as we need the ground truth on the location to evaluate the attack performance. In the worst case, our attack evaluation provides a lower bound on the average privacy of sharing hashtags. Moreover, as shown in Figure 3.1, users usually share hashtags, to a large extent regardless of whether they jointly share their location. The training set is a priori not biased, as even the adversary must have access to both location and hashtag information to train his classifier.

3.10 Conclusion

The rapid adoption of new concepts and tools in the modern society poses novel risks towards privacy that users are not always aware of. In this paper, we thoroughly investigate the impact of hashtag sharing on location privacy. We present and show the effectiveness of a random forest classifier that uses hashtags for fine-grained location inference within and across cities. In order to counter the proposed attack, we propose three obfuscation mechanisms and an efficient system that determines the optimal/near-

optimal set of hashtags to preserve privacy without degrading significantly the original utility.

Our work demonstrates a clear threat towards location privacy stemming from the use of hashtags. Even though some users intentionally use hashtags to reveal their (more or less accurate) locations, many of them do not make use of hashtags that are clearly linked to location information, and thus should expect some level of location privacy. In the context of privacy, even a moderately large absolute number of affected individuals represents a serious enough negative effect [23], which is effectively reduced by our proposed privacy-preserving system, Tagvisor.

As future work, we plan to study how the publication time of the post can help improve the classifier's success. Besides temporal information, other cues could boost the accuracy of the inference attack, for example, social proximity to already geo-located users or use of deep learning on the images to identify background locations, etc. All these can further degrade users' location privacy, and it will be interesting to incorporate them in our framework in the future. Another direction for future work would be to incorporate defenses against other privacy threats arising out of hashtags such as demographics, linkability and friendship between users.

4

Everything About You

A Multimodal Approach towards Friendship Inference in Online Social Networks

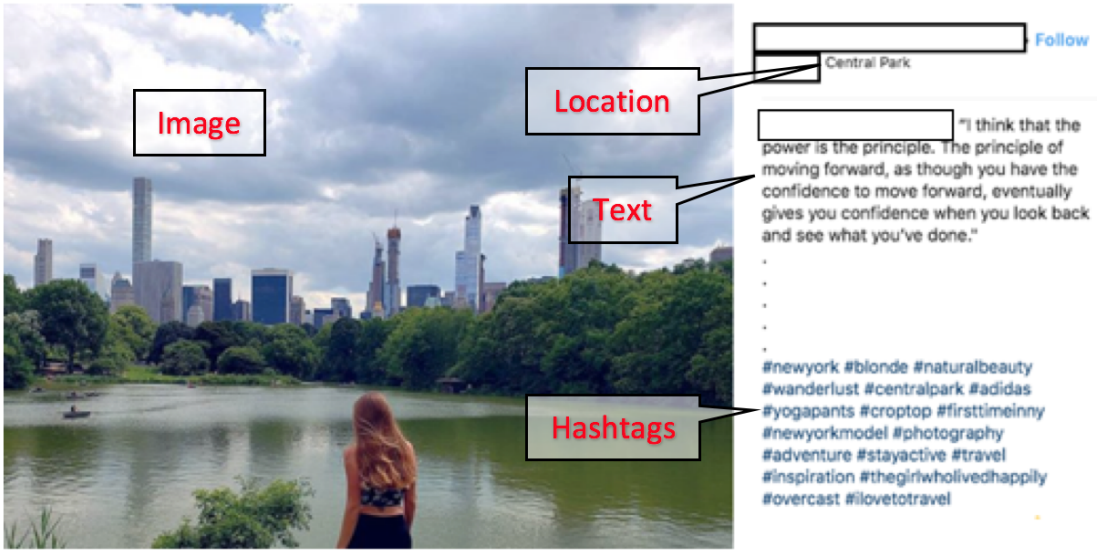


Figure 4.1: Example of an Instagram post containing multimodal information. We investigate privacy threats emerging in such a multimodal setting against an adversary that uses a combination of images, locations, text and hashtags shared with a post as well as already revealed friendships between users of an OSN.

4.1 Motivation

As discussed earlier, Online Social Networks (OSNs) have become an indispensable part of people's life in the modern society. Leading players in the businesses, such as Facebook and Instagram, have acquired a large number of users. Initially, people used OSNs mostly to articulate their friendships. With time, online activities have become increasingly elaborate and diverse. Users publish tweets on Twitter, statuses on Facebook and captions on Instagram. Photos constitute a large proportion of the social media content. People also share their locations, popularly known as check-ins, via GPS-enabled smartphones. More recently, hashtags have become very popular for content discovery and advertisement. Figure 4.1 shows an example of a typical OSN/Instagram post that consists of a photo, shared with a textual caption, supplemented with hashtags, and a geo-tagged location.

Extensive amounts of such diverse user data is generated in OSNs everyday which has been exploited by various parties to gain a deeper understanding of our society. On the downside, privacy risks stemming from data sharing in OSNs have become a major concern. Researchers have highlighted potential attacks on various aspects, e.g., social security number prediction [1], stalking and re-identification [37] and various inference attacks. However, most prior works in this direction explore only a single kind of user generated content to infer another hidden attribute. For example, Backes et al. [8] infer social relations from users' locations; Zhang et al. [P1] infer user locations from hashtags; Shoshitaishvili et al. [103] infer pairs that are dating using facial recognition. In a realistic scenario, adversaries would utilize not just a single kind but as many different kinds of data as possible to achieve their goal. Works that do consider heterogeneous

data [33] utilize only a single instance of static profile data such as username, profile picture, home location, etc or a footprint of only one modality [66]. However, users generate much more than only one kind of content and they generate this multimodal content continuously. The multimodal footprint of users built up over time from the data they post everyday, collectively gives a much more faithful reflection of the user's interests. Such multimedia footprints of users are also getting increasingly diverse.

In order to fully assess the privacy risks arising from such multimodal footprints over time, a comprehensive evaluation is needed which, to our knowledge, has not been addressed in the literature. This paper aims to fill this gap with a framework incorporating multimodal data in OSNs to assess users' privacy risks.

Out of the various privacy risks in OSNs, we concentrate on inferring social relations. Social relationships are recognized to pose high threats to user privacy. In addition to revealing users' social identities, social relationships can potentially leak further sensitive attributes about a user, which the user explicitly intends to hide in their own profile. In practice, OSN users have also begun to increasingly hide their social circles [26].

4.2 Contributions

We conduct the first large-scale privacy assessment of collective multimodal information shared continuously over time in OSNs against an adversary that leverages not just a single component but combines all available components of user-generated content. In particular, we consider five kinds of information: images, texts, location, hashtag, and (incomplete) social relations. We evaluate on a real-life dataset containing 22 million user posts in two cities which we collected from Instagram.

Our contributions are two-fold. First, we introduce three novel monomodal friendship inference attacks based on hashtags, images, and text, respectively. To the best of our knowledge, this is the first work that uses text and image modalities to conduct friendship inference in social networks. Our hashtag attack massively outperforms the only prior work on friendship inference from hashtags [117] by 7%. We use self-engineered statistical features, such as the entropy and frequency of usage patterns for our hashtag and text based attacks. For images, we rely on a state-of-the-art ResNet model to extract information from images. Experiments show that our hashtag and text attacks achieve strong performance with AUCs above 0.86 and 0.83 respectively. Our image attack, while being the weakest, still performs well with AUCs up to 0.637.

Second, we design a multimodal attack that combines five components of posts to infer social relations. Extensive experiments show that this attack outperforms all monomodal attacks with AUCs above 0.9. We also show that the multimodal attack exploits post components that compliment each other and the success of most attacks increases significantly when combined with additional components. We further demonstrate the robustness of the multimodal attack against information hiding, and show that even after deleting 50% of the users' posts, the drop in the attack performance is relatively low.

4.3 Model

4.3.1 User Model

We define an OSN by an undirected, unweighted graph $\mathcal{G} = (U, E)$, where the set U contains all the users and $E \subseteq \{\{u, v\} | u \in U, v \in U, u \neq v\}$ is the set of friendships. An edge exists between 2 users if they mutually and visibly declare each other as friends in their profiles.

A user u shares a set of posts P_u which can be composed of images, texts, hashtags and location check-ins. We define a single post $p_u \in P_u$ as a 4-tuple $\langle i_p,\ \ell_p,\ t_p,\ H_p \rangle$ which denotes that a user $u \in U$ posts an image i_p, at location ℓ_p (checks in at ℓ), with a text t_p annotated with the set of hashtags H_p. All the posts of u is defined by $P_u = \{p_u^1, p_u^2 \dots p_u^{|P_u|}\}$.

We denote the overall set of posts by P, images by I, textual posts by T, the set of hashtags by H and the set of locations L. Accordingly, for u, $I_u \subset I$ represents the set of images, $H_u \subset H$ represents the set of hashtags, $T_u \subset T$ the set of texts and $L_u \subset L$ the set of check-ins shared by u.

4.3.2 Adversary Model

We consider an adversary that tries to predict whether a pair of individuals u and v that explicitly hide their friendship are socially related or not. A pair of users can hide their friendship in different ways depending on the OSN application, for example by not following each other or not enlisting each other in their respective friend lists.

Users may want to do so for several reasons such as suppressing a workplace romance, or protecting themselves from *target development* by surveillance agencies ([48] reveals how NSA collects mobility and travel habit data to spot new unknown associates from already known targets). The adversary could be any party with the ability to crawl an OSN and access user posts.

Our method can also be used as a tool by users who want to maintain some social distance online from certain people. To this end, our attacks output a probability score of a user pair being related, based on their (as well as other people's) past activity on the social network. This score is an indication of the closeness or similarity of the user pair in the current state of the social network. The knowledge of the current state is publicly available and can be assumed to exist in a regularly trained machine learning model that resides on the user device. Whenever a privacy conscious user wants to make a post on social media, (s)he can first use such a tool on users that she does not want to be associated with, to check the closeness scores that would result after making the post.

The adversary has access to posts shared by the pair of users $\{u, v\}$ namely P_u and P_u as well as the friendships explicitly published on the OSN, E. A feature $\vec{x}^D_{\{u,v\}}$ for a pair $\{u, v\}$, is a function of the posts P_u and P_v shared by them and friendships published on the OSN, $\mathcal{G}$. The adversary constructs features $\vec{x}^D_{\{u,v\}}$ from each component of posts: hashtags, location check-ins, texts and images as well as published friendships such that $D \subseteq \{H, T, I, L, E\}$. The prediction attack is then a binary classification task $f : \vec{x}^D_{\{u,v\}} \rightarrow \{0, 1\}$, where we encode the existence of friendship by the positive

class and the absence of friendship by the negative class. In the upcoming sections, we describe the process of constructing the features $\vec{x}^{D}_{\{u,v\}}$ from each of the individual data components.

We distinguish between 2 kinds of adversaries for ease of readability. The first kind of adversary has access to and utilizes only a single component of OSN posts. We name them according to the information they have access to/use:

1. The Text Adversary, `TA` which uses the text component of OSN posts. We denote the feature vector used for the corresponding prediction task by $\vec{x}^{T}_{\{u,v\}}$.

2. The Hashtag Adversary, `HA` which uses the hashtag component of OSN posts. We denote the corresponding feature vector by $\vec{x}^{H}_{\{u,v\}}$.

3. The Image Adversary, `IA` which uses the image component of OSN posts. We denote the corresponding feature vector by $\vec{x}^{I}_{\{u,v\}}$. 4. The Location Adversary, `LA` which uses the location component of OSN posts. We denote the corresponding feature vector by $\vec{x}^{L}_{\{u,v\}}$.

5. The Network Adversary, `NA` which uses the friendships explicitly published by OSN users. Since friendships on the OSN are represented by edges E, we denote the corresponding feature vector by $\vec{x}^{E}_{\{u,v\}}$.

We refer to this group of attacks as *monomodal attacks*.

The second kind of adversary combines multiple information components shared by users of OSNs. We name such an adversary as the Multimodal Adversary $\mathcal{M}$ and this group of attacks as *multimodal attacks*. When the $\mathcal{M}$ uses a subset D' of post components, we call it a Subset Adversary $\texttt{SA}^{D'}$, $D' \subset \{H, T, I, L, E\}$ and $2 \leq |D'| \leq 4$.

4.4 Dataset

To evaluate our attacks, we use a dataset of Instagram, one of the largest OSNs * of this era, which has been used previously for other works [8], [P1], [P4], [117]. Ideal as it might be, it remains infeasible to obtain the ground truth for a dataset where people hide their friendship online in spite of secretly being friends in real life. Therefore we adopt the best possible alternative and evaluate our attacks on a simulation of the ideal dataset by using a real life Instagram dataset. Compared to other OSNs like Twitter which is mainly used for sharing news, current events and public statements, Instagram is better suited for friendship inference as it is mainly used to share topics related to lifestyle, interests, hobbies, leisure and travel etc. Consequently, Instagram friends tend to be socially closer in real life as well. On the other hand, crawling Facebook (the largest OSN) is prohibited. Moreover, Instagram focuses more on sharing of visual content and hashtags and is used by many brands to humanize their content for better connecting customers and growing brand awareness. A typical Instagram post as shown in Figure 4.1 consists of a photo(s) or video along with captions and hashtags and is geo-tagged with locations or Points of Interest (POIs).

The dataset contains 8,309,482 posts from 10430 users in Los Angeles and 14,671,556 posts from users in New York comprising of photos, captions, hashtags and tagged locations that were publicly posted along with the corresponding user ID, and for each

*https://www.dreamgrow.com/top-15-most-popular-social-networking-sites/

user, the respective followers. Moreover, there are 18,221 friend pairs for LA out of which we obtain 17,043 pairs that have at least 1 out of the 5 types of feature data. From NY we collected 41,501 friend pairs, out of which 40,883 pairs have feature data in at least 1 out of the 5 attack groups. For individual attacks, we eliminate friend pairs that do not have the corresponding feature data. We further do not target friendship inference between Instagram accounts that are likely to be celebrities or bots. Therefore, we filter out accounts whose numbers of followers are above the 90th percentile (celebrities) or below the 10th percentile (bots) following prior works, [19, P1, 100, 8].

Ethics. Our institution neither provides an IRB nor mandates (or enables) approval for such experiments. Nevertheless, only publicly available data is collected via Instagram's public API in 2016. We anonymize the datasets by removing all users' screen names, and replacing their Instagram IDs with randomly generated numbers. Only these anonymized datasets are then stored for experiments in a central server with up-to-date software, encrypted hard drives and access only possible from two white-listed internal IP addresses using authorized SSH keys.

4.5 Attack using Text

In this section, we describe our friendship inference attack for the Text Adversary `TA` followed by an experimental evaluation.

4.5.1 Attack Methodology

The Text Adversary `TA` has access to the text like tweets, statuses or captions shared by users. We design features for this attack in two phases:

4.5.1.1 Phase I

We first collect all textual data shared by each user and tokenize the text sequences. We weigh the tokens according to the number of users in the overall OSN that use them such that tokens used by a majority of users have lower weight than those used by fewer users. To this end we use a Bag of Words model with *TF-IDF* (Term-Frequency Inverse Document-Frequency). For a term t and a document d, *TF-IDF(t,d)* is the product of the term frequency tf(t,d) and inverse document frequency idf(t).

$$\textit{TF-IDF(t,d)} = \textit{tf(t,d)} \times \textit{idf(t)}$$

where term frequency is defined as $\textit{tf(t,d)} = \frac{n_{t,d}}{\sum_{t' \in d} n_{t',d}}$ with $n_{t,d}$ as the number of times term t appears in document d and $\sum_{t' \in d} n_{t',d}$ as the total number of terms in d. Meanwhile, inverse document frequency is $\textit{idf}(t) = log \frac{n_d}{1+\text{df}(d,t)}$ with n_d as the total number of documents and df(d,t) as the number of documents that contain term t. As we see, the IDF reweights the popular but less interesting words.

The resulting *TF-IDF* vectors are then normalized by the Euclidean norm:

$$v_{norm} = \frac{v}{||v||_2} = \frac{v}{\sqrt{{v_1}^2 + {v_2}^2 + \cdots + {v_n}^2}}.$$

We therefore convert the text sequences of all users into a matrix of *TF-IDF* vectors. Assuming W is the overall set of words in the textual dataset for a city or in other words, the vocabulary of the Text Adversary `TA`, we denote each user's *TF-IDF* vector by $\vec{T_u} \in R^{|W|}$ such that $\vec{T_u} = (\textit{TF-IDF}(t_1, u),\ \textit{TF-IDF}(t_2, u), \ldots, \textit{TF-IDF}(t_{|W|}, u))$.

4.5.1.2 Phase II

For each user pair $\{u, v\}$, we compare the *TF-IDF* vectors $\vec{T_u}$ and $\vec{T_v}$. We calculate 8 common pairwise distances to form the feature vector for the Text Adversary namely $\vec{x}^T_{\{u,v\}} = (\text{cosine}(\vec{T_u}, \vec{T_v}),\ \text{euclidean}(\vec{T_u}, \vec{T_v}),\ \text{correlation}(\vec{T_u}, \vec{T_v}),$
$\text{chebyshev}(\vec{T_u}, \vec{T_v}),\ \text{bray\_curtis}(\vec{T_u}, \vec{T_v}),\ \text{canberra}(\vec{T_u}, \vec{T_v}),$
$\text{manhattan}(\vec{T_u}, \vec{T_v}),\ \text{sq\_Euclidean}(\vec{T_u}, \vec{T_v}))$. Table 4.1 contains the definitions of each measure.

Table 4.1: Definitions of the common pairwise distance measures used for the textual attack. $\overline{\vec{T_u}}$ represents the mean of $\vec{T_u}$. $\vec{T}_{ui}$ represents the ith element in $\vec{T_u}$.

measure	definition
$\text{cosine}(\vec{T_u}, \vec{T_v})$	$\frac{\vec{T_u} \cdot \vec{T_v}}{\|\|\vec{T_u}\|\|_2\ \|\|\vec{T_v}\|\|_2}$
$\text{euclidean}(\vec{T_u}, \vec{T_v})$	$\|\|\vec{T_u} - \vec{T_v}\|\|_2$
$\text{correlation}(\vec{T_u}, \vec{T_v})$	$\frac{(\vec{T_u} - \overline{\vec{T_u}}) \cdot (\vec{T_v} - \overline{\vec{T_v}})}{\|\|\vec{T_u} - \overline{\vec{T_u}}\|\|_2\ \|\|\vec{T_v} - \overline{\vec{T_v}}\|\|_2}$
$\text{chebyshev}(\vec{T_u}, \vec{T_v})$	$\max_{i=1}^{d} \lvert\vec{T}_{ui} - \vec{T}_{vi}\rvert$
$\text{bray\_curtis}(\vec{T_u}, \vec{T_v})$	$\frac{\sum_{i=1}^{d} \lvert\vec{T}_{ui} - \vec{T}_{vi}\rvert}{\sum_{i=1}^{d} \lvert\vec{T}_{ui} + \vec{T}_{vi}\rvert}$
$\text{canberra}(\vec{T_u}, \vec{T_v})$	$\sum_{i=1}^{d} \frac{\lvert\vec{T}_{ui} - \vec{T}_{vi}\rvert}{\lvert\vec{T}_{ui}\rvert + \lvert\vec{T}_{vi}\rvert}$
$\text{manhattan}(\vec{T_u}, \vec{T_v})$	$\sum_{i=1}^{d} \lvert\vec{T}_{ui} - \vec{T}_{vi}\rvert$
$\text{sq\_Euclidean}(\vec{T_u}, \vec{T_v})$	$\|\|\vec{T_u} - \vec{T_v}\|\|_2^2$

4.5.2 Evaluation

We now present the preprocessing performed on the raw dataset, followed by the experimental set-up and the metric we use for evaluating the prediction performance. Finally we present the results of the experiments.

4.5.2.1 Data preprocessing

In LA, we have 10,018 users that share a total of 7,146,810 captions with their Instagram posts. For NY we have 12,756,384 captions shared by 18,713 users. In order to remove terms that are too rare and too frequent, we filter out words that appear in caption

sequences of more than 100 users and less than 2 users. Following this step, we obtain 16,878 out of a total of 17,043 friend pairs from LA and 40,228 out of 40,883 friend pairs from NY. We randomly sample an equal number of stranger pairs. Table 4.2 shows the number of users, terms and friend pairs that we have after pre-processing.

Table 4.2: Pre-processed caption data statistics.

	LA	NY
users	10,018	18,713
terms	587,190	886,120
friend pairs	16,878	40,228

4.5.2.2 Experimental Set-up

We rely on a random forest with 100 trees to carry out this and all further attacks as it outperforms other classifiers like Gradient Boosting Machine, Ada Boost, Support Vector Machine and Logistic Regression in our experiments. Increasing the number of trees beyond 100 did not significantly change our performance. We perform a 5-fold cross validation with a 80-20 split into training and test set and repeat it 5 times with shuffle. We do not tune any parameters and therefore do not require a validation subset.

4.5.2.3 Metric

We evaluate our attack using the AUC (Area Under the receiver operating characteristic (ROC) Curve) metric. AUC exhibits a number of desirable properties when compared to accuracy such as decision threshold independence and invariance to a priori class probabilities. The latter is particularly desired for friendship inference in OSNs in order to correctly evaluate the attacks in our down-sampled prediction space. Moreover, there exists a conventional standard for interpreting AUC in a rather intuitive manner. For AUCs in the range [0.5, 1], with 0.5 associated with random guessing and 1 with perfect guessing, values above 0.8 represent a very good prediction and values above 0.9 represent excellent prediction. This simplicity renders comparison against baseline models dispensable.

4.5.2.4 Results

We first assess how each the 8 individual distance measures perform on their own and thus compare their AUC values in the unsupervised setting in Figure 4.2. We see that correlation achieves the highest score in both cities, with an AUC value of 0.82 for LA and 0.84 for NY. Squared Euclidean, Bray-Curtis, Euclidean and cosine distance achieve only slightly lower AUCs. This shows that the success of our attack is not dependant on the city from which we collect our data. In the supervised setting, we further achieve a much stronger performance in both cities with AUC of 0.83 for LA and 0.85 for NY.

It is worth noting that newer, more advanced NLP techniques might be more optimal for the prediction task. However, this paper is focussed on highlighting the privacy risks

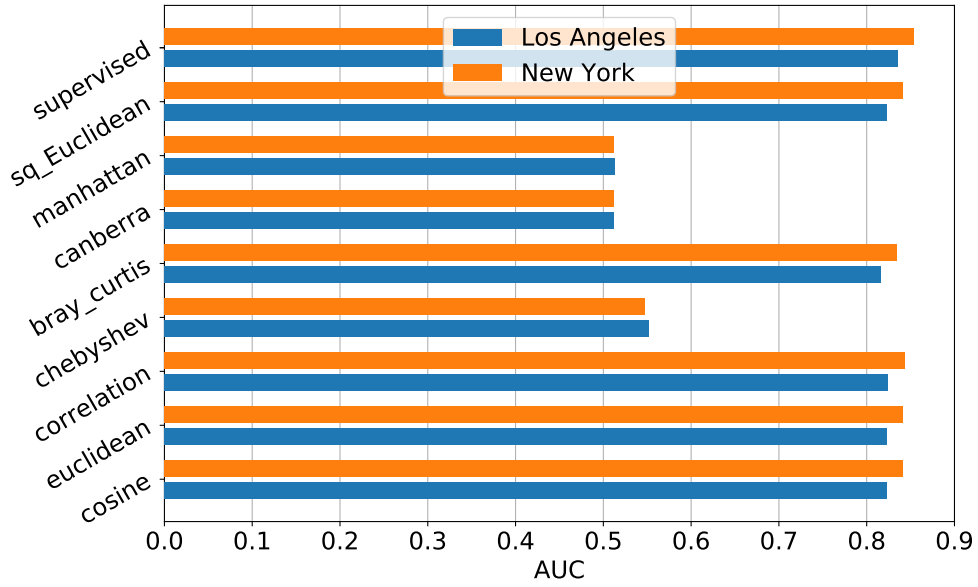


Figure 4.2: Comparison of individual features for our Text Adversary

from an adversary that combines multiple modalities and the need to defend against such multimodal attacks. Other state-of-the-art techniques can be adopted similarly in a straightforward manner.

4.6 Attacks using Hashtags

We now describe our friendship inference attack for the Hashtag Adversary `HA` along with an experimental evaluation.

4.6.1 Attack Methodology

We train a machine learning classifier with the auxiliary knowledge of the adversary which consists of a 10-dimensional feature space which we present below. The adversary uses the trained classifier and carries out his inference attack on a target user pair by computing these scores for the pair using their hashtag sharing information.

Our feature design is influenced by certain properties we observed during preliminary analyses on our dataset. These properties fall into 2 main directions namely the hashtags shared in common by a user pair and the information content or the relevance of individual hashtags.

Firstly in Figure 4.3 we observe that in both cities, friends tend to have a higher number of hashtags in common as compared to strangers. The fraction of common hashtags, (number of overlapping hashtags divided by the total number of hashtags shared by both users), also shows a similar property; a high fraction of the overall hashtags shared by a user overlaps with the hashtags shared by a friend of the user. Moreover, we only consider stranger pairs that have at least 1 hashtag in common.

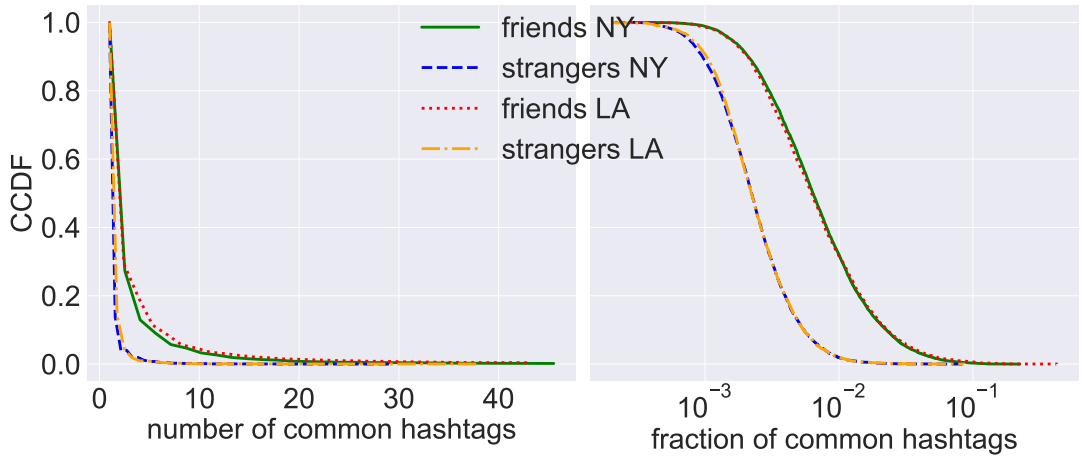


Figure 4.3: CCDF of the number and fraction of hashtags in common between friends versus strangers.

Including stranger pairs with no common hashtags would only further increase the gap between the friend and stranger curves. The number of common hashtags shared by a user pair therefore appears to be a very good indicator of the existence of friendships between them which motivates the design of our first 4 features. Denoting the set of hashtags shared by user u and user v by H_u and H_v respectively, we define x_{comm} and x_{frac} as the number and fraction respectively of hashtags shared by both users.

$$x_{comm} = |H_u \cap H_v| \tag{4.1}$$

$$x_{frac} = \frac{|H_u \cap H_v|}{|H_u \cup H_v|} \tag{4.2}$$

We define x_{dotp} as the (dot product of) number of times the common hashtags were shared by both users. We represent the number of times a particular user u shares each hashtag $h_i \in H$ as a user-hashtag count vector $\vec{n_u} = (n_{u,h_1}, n_{u,h_2}, \ldots n_{u,h_{|H|}})$, where n_{u,h_i} is the number of times h_i is shared by u. We use $\vec{n_v}$ analogously for user v in a pair. We define x_{cos} as the cosine similarity between the user-hashtag count vectors $\vec{n_u}$ and $\vec{n_v}$ for a user pair $\{u, v\}$. Thus,

$$x_{dotpro} = \vec{n_u}\vec{n_v} \tag{4.3}$$

$$x_{cosine} = \frac{\vec{n_u}\vec{n_v}}{\sqrt{\vec{n_u}^2\vec{n_v}^2}} \tag{4.4}$$

The second property that can be observed in Figure 4.4 (left) is that hashtags shared by only a small number of users regularly are more likely to hold special significance for them, such as hobbies, places visited, private clubs, jargon, acronyms and idioms and so on. Conversely, hashtags shared at least as frequently but by a larger number of users are likely to be merely common terms and not characteristic of a socially related

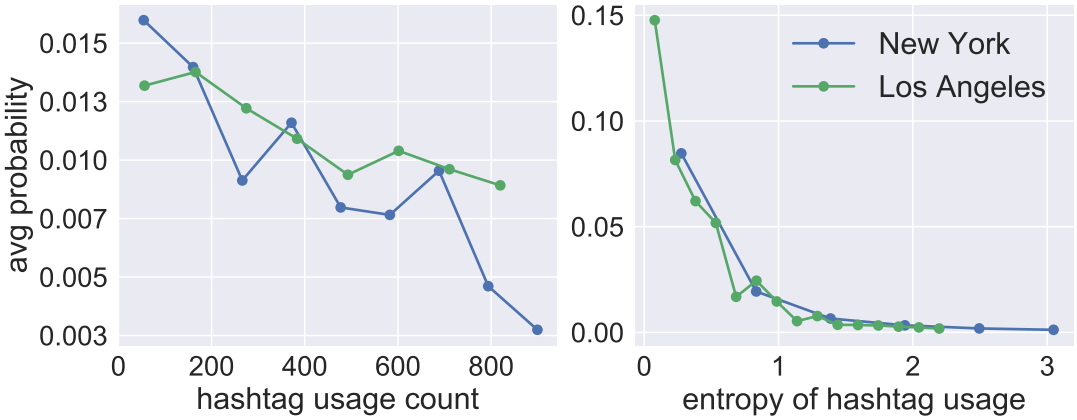


Figure 4.4: Average probability that two users who share the same hashtag are friends as a function of (left) overall popularity of the hashtag and (right) entropy of the hashtag usage. The left x-axis marks the usage count which is the total number of times the corresponding hashtag was shared by all users in the OSN. The right x-axis marks the entropy of the vector counting the number of times the corresponding hashtag was shared by various users.

users. We therefore formalize the notion that a shared hashtag is more likely to reveal the relationship between users if it is less popular in general. We denote the number of times a particular hashtag h was shared by each user $u_i \in U$ as a hashtag-user count vector $\vec{n_h} = (n_{h,u_1}, n_{h,u_2}, \ldots n_{h,u_{|U|}})$, where n_{h,u_i} is the number of times u_i shares h. We define x_{minH} as the minimum of the total number of times all users shared a hashtag among the set of common hashtags between u and v. We represent the total number of times all users U shared a hashtag h_i by $n_{h_i} = \sum_{u \in U} n_{h_i,u}$.

$$x_{minH} = min(n_{h_i} : (h_i \in H_u \cap H_v)) \tag{4.5}$$

Following the common Adamic Adar (Frequency-Weighted Common Neighbors) [2] score, we define x_{aaH} as the sum of the reciprocals of the logarithm of n_{h_i} which is the number of times a hashtag h_i common between u and v was shared by all users.

$$x_{aaH} = \sum_{h_i \in H_u \cap H_v} \frac{1}{\log n_{h_i}} \tag{4.6}$$

The Adamic Adar measure improves counting of common features by giving more weight to rarer features. We adapt the measure in an Information Theoretic approach to formalize the intuitive notion that lower entropies are more informative. We define entropy E of a hashtag h_i as the entropy of its hashtag-user count vector over all users U such that $E_i = E(\vec{n_{h_i}})$. Therefore, we define x_{minE} as the minimum among the entropies of all common hashtags,

$$x_{minE} = min(E_i : (h_i \in H_u \cap H_v)) \tag{4.7}$$

and x_{aa} as the sum of the reciprocals of entropies of each common hashtag

$$x_{aaE} = \sum_{h_i \in H_u \cap H_v} \frac{1}{E_i} \tag{4.8}$$

In the direction of the second property, we also measure the number of times different users share a particular hashtag as a count vector for a hashtag and then calculate the entropy of the count vectors for all hashtags in our dataset. Figure 4.4 (right) further shows that hashtags whose usage count vectors have lower entropies are more indicative of social relations between their users. We therefore formalize the property that rare hashtags with lower entropies are more useful. We define x_{w_aaE} as x_{aa} weighted by the number of hashtags shared by both users and x_{f_aaE} as x_{aa} weighted by the sum of the reciprocals of entropies of hashtags shared by both users.

$$x_{w\_aaE} = \frac{x_{aaE}}{|H_u \cup H_v|} \tag{4.9}$$

$$x_{f\_aaE} = \frac{x_{aaE}}{\sum_{h_i \in H_u \cup H_v} \frac{1}{E_i}} \tag{4.10}$$

Thus we represent the feature vector for the Hashtag Adversary `HA` for a pair of users $\{u, v\}$ as $\vec{x}^{H}_{\{u,v\}} = (x_{comm}, x_{frac}, x_{dotpro}, x_{cosine}, x_{minH}, x_{aaH}, x_{minE}, x_{aaE}, x_{w\_aaE}, x_{f\_aaE})$

4.6.2 Evaluation

We evaluate the attack in the same way as for the caption attack, i.e. using AUC for a random forest with five-fold cross validation. We first explain the data preprocessing, followed by a baseline method proposed in a recent work. Finally we present the results of the experiments.

Table 4.3: Pre-processed hashtag data statistics.

	LA	NY
posts	1,712,500	2,999,603
users	10,359	18,896
hashtags	406,692	595,502
friend pairs	5,229	12,765

4.6.2.1 Data preprocessing

We have 2,306,631 unique hashtags shared by 10,418 users in our raw dataset from LA. For NY, we have 3,090,451 unique hashtags shared by 19,015 users.In order to remove hashtags that are shared too frequently and too infrequently, we filter out hashtags that are shared by less than 2 and more than 10 users. Following this step, we obtain 5229 out of a total of 17,043 friend pairs from LA and 12,765 out of 40,883 friend pairs from NY. We randomly sample an equal number of stranger pairs. Table 4.3 shows the number of users, terms and friend pairs that we have after preprocessing.

4.6.2.2 Baseline

We consider as baseline, the only prior work by Zhang et al. [117] that rely on hashtags to perform friendship prediction. This method uses a user-hashtag bipartite graph embedding model to automatically summarize hashtag profiles as features for users.

4.6.2.3 Results

We first compare the performance of the 10 individual features on their own by evaluate each in an unsupervised setting. Figure 4.5 shows the AUC values. We see that x_{w_aaE} achieves the highest score with an AUC of 0.87 for NY and 0.85 for LA, closely followed by x_{f_aaE} with an AUC of 0.87 for NY and 0.85 for LA. Both features are based on the property that rare hashtags with lower entropies are more informative. At the third place is the cosine distance between the user-hashtag count vectors x_{cosine} followed by x_{aaH}, x_{minH} and x_{frac}. We see that the remaining features are significantly outperformed. Nevertheless, they perform much better than random guessing. These results show that the success of our hashtag attack is independent of the city from which we collect our data. Although 6 out the 10 features perform sufficiently well on their own, using them in a supervised setting with the random forest yields stronger performances in both cities, with an AUC value of 0.86 for LA and 0.87 for NY.

In contrast, the baseline method achieves an AUC of 0.8 in Los Angeles and 0.82 in New York. Its also noteworthy that this method involves training many hyperparameters using cross validation just for learning features. This huge overhead in processing required in this baseline, combined with around 7% lower AUC compared to our method, clearly demonstrates the strength of hand-engineered features for a privacy attack such as friendship inference.

4.7 Attack using Images

In this section, we describe our friendship inference attack using images followed by an experimental evaluation.

4.7.1 Attack Methodology

The Image Adversary (`IA`) tries to infer social relationships using the content of the images posted by users on an OSN. We do not use facial recognition in these images following the assumption that users who wish to hide their friendship with a certain people would refrain from posting a picture with them in the first place.

4.7.1.1 Extracting Information from Images.

The recent advent of deep neural networks (NNs) and the availability of massive computing power and datasets, has facilitated huge breakthroughs in image classification. In addition to high image classification accuracies, pre-trained NNs can directly be used on any image to automatically obtain a meaningful feature representation of the image. This is why we choose a NN based approach for our Image Adversary.

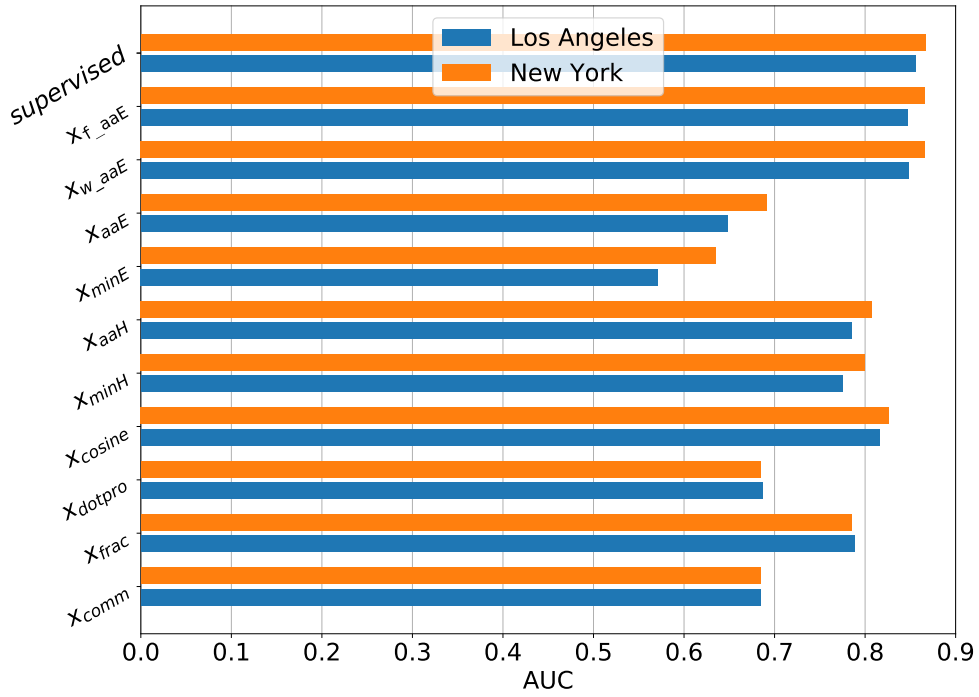


Figure 4.5: Comparison of individual features for our Hashtag Adversary

We use a Residual Neural Network (ResNet) which is an advanced type of a convolutional NN for meaningfully extracting information from images. ResNets solve the problem of degrading/saturating accuracy that is faced by increasing the depth of deep convolutional NNs [45]. Moreover, training ResNets is easier than training convolutional NNs when they need to get deeper in order to solve increasingly complex tasks. We adopt a ResNet namely the Places365-ResNet18 model for PyTorch trained on the Places365-Standard Database [121] which allows learning of deep scene features for various scene recognition tasks. The training set consists of 1.8 million images from 365 scene categories, with at most 5000 images per category, consistent with real-world frequencies of occurrence. †

4.7.1.2 Feature Design.

We extract the final softmax output layer produced by the neural network for each of the images shared by users in our dataset. For an image I_i, the final layer is a posterior probability distribution over the 365 classes that correspond to the 365 different scene categories $c_1, \ldots, c_{365}$ which we represent by $\vec{C_i} = \Pr(c_1, \ldots, c_{365}|I_i)$. For each user u we count the number of images that belong to each of the 365 scene categories. An image belongs to a category if its corresponding class membership probability outputted by the ResNet is higher than a certain percentile threshold (which we describe during the dataset pre-processing in section 4.7.2.1). We denote this count by the vector

†The scene category list can be found on `https://github.com/metalbubble/places_devkit/blob/master/categories_places365.txt`

$\vec{n_u} = (n_{u,1}, n_{u,2}, \ldots n_{u,365})$ where $n_{u,c}$ represents the number of images shared by user u that depict the scene corresponding to category number c. We use $\vec{n_v}$ analogously for user v in a pair. We then design a set of 4 feature types for each user pair $\{u, v\}$ as follows:

1. We define the feature `min_count` as the category-wise minimum of the 2 count vectors $\vec{n_u}$ and $\vec{n_v}$ i.e. out of the number of images posted by both users in each of the 365 categories, we count the minimums, $\texttt{min\_count} = (\min(\mathtt{n_{u,1}}, \mathtt{n_{v,1}}), \mathtt{min}(\mathtt{n_{u,2}}, \mathtt{n_{v,2}}), \ldots \min(\mathtt{n_{u,365}}, \mathtt{n_{v,365}}))$

2. We define the feature $\mathtt{x_{cosine}}$ as the cosine distance between the category-wise count vectors $\vec{n_u}$ and $\vec{n_v}$.

3. We define the feature $\mathtt{x_{F\_maxcat}}$ as the maximum over the 365 components of the vector `min_count`, which represents the frequency of the mutually most frequently shared category in images shared by both users. $\mathtt{x_{F\_maxcat}} = \mathtt{max}(\mathtt{min}(\mathtt{n_{u,1}}, \mathtt{n_{v,1}}), \mathtt{min}(\mathtt{n_{u,2}}, \mathtt{n_{v,2}}), \ldots \mathtt{min}(\mathtt{n_{u,365}}, \mathtt{n_{v,365}}))$

4. We perform a category-wise sum of the posterior probabilities over all images shared by each user which we represent by a usage-vector $\vec{s_c} = (s_{c,u_1}, s_{c,u_2}, \ldots s_{c,u_{|U|}})$, where s_{c,u_j} represents the summed up probability that a category c is depicted over all images shared by u_j. We define the feature $\mathtt{x_{E\_maxcat}}$ as the entropy of the usage-vector $\vec{s_{maxcat}}$ of the mutually most frequently shared category $maxcat$.

Finally, we represent the feature vector for the Image Adversary for a pair of users $\{u, v\}$ as $\vec{x}^I_{\{u,v\}} = (\texttt{min\_count}, \mathtt{x_{cosine}}, \mathtt{x_{F\_maxcat}}, \mathtt{x_{E\_maxcat}})$.

Table 4.4: Pre-processed image data statistics.

	LA	NY
users	5,082	12,575
images	2,052,619	4,840,670
friend pairs	4,262	18,847

4.7.2 Evaluation

We evaluate the attack in the same way as for the previous attacks, i.e. using the AUC metric for a random forest classifier and assigning 80% of each of the friend and stranger pairs for training and 20% of each of the friend and stranger pairs for testing.

4.7.2.1 Data preprocessing

We could download 4,857,971 photos from 8,309,482 urls for our image-based attack from LA and 5,092,951 photos from 14,671,592 urls from NY. We feed each of these photos onto our pre-trained ResNet model and extract the final layer, i,e the probability distribution over the 365 scene categories. We only require the significant scene categories that are predicted to be depicted in an image. Therefore, we first remove the category probabilities lower than 5% by setting them to 0. We obtain 4,262 out of a total of 17,043 friend pairs from LA and 18,837 out of a total of 40,883 friend pairs from NY.

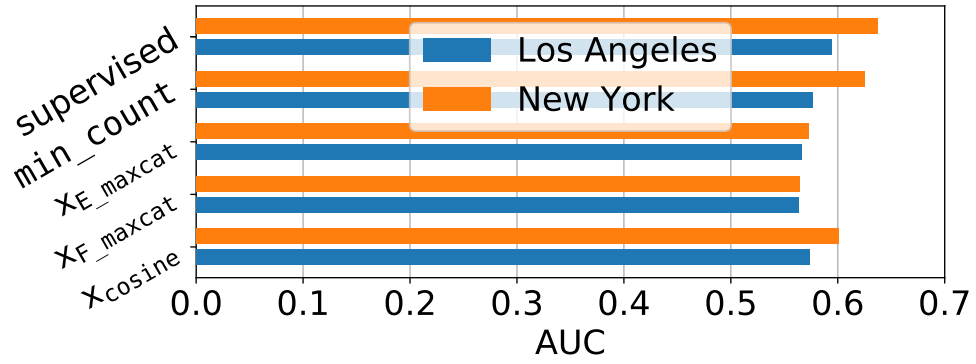


Figure 4.6: Comparison of individual features for our Image Adversary

We randomly sample an equal number of stranger pairs. Table 4.4 shows the number of users, images and friend pairs that we have after preprocessing.

4.7.2.2 Results

In order to evaluate how each of the features perform on their own, we evaluate each of them in an unsupervised setting. Figure 4.6 shows the AUC values for the individual features for our attack namely `min_count`, $\mathtt{x}_{\mathtt{cosine}}$, $\mathtt{x}_{\mathtt{F\_maxcat}}$ and $\mathtt{x}_{\mathtt{E\_maxcat}}$. We see that both `min_count` and $\mathtt{x}_{\mathtt{cosine}}$ perform equally in both cities. In NY, `min_count` performs the best (AUC 0.63) followed by $\mathtt{x}_{\mathtt{cosine}}$ (AUC 0.61). In LA, $\mathtt{x}_{\mathtt{cosine}}$ performs the best (AUC 0.58) followed by `min_count` (AUC 0.57). In the supervised setting, we achieve an AUC value of 0.64 for NY and 0.60 for LA. Figure 4.7 shows the AUC values of the image attack in the supervised setting compared with the other attacks using hashtag, text, location, partial network data as well as the multimodal attack.

Although the performance is relatively weak, we can still infer a signal and perform much better than random guessing. Inferring the content of images is an inherently hard problem. Technology has not yet been able to achieve what an average human can in terms of drawing conclusions from the contents of a single image. Most research on computer vision and face recognition techniques work only with small labelled datasets and the scalability to real world, unseen data is yet to be explored. We are the first to perform image content analysis on a large scale real world dataset. Since in the large scale, an adversary using computer vision techniques can perform inference attacks in a matter of seconds, privacy analysis of image content remains an important problem.

4.8 Attack using Locations and Network Information

In this section we present the details of our Location Adversary `LA` and Network Adversary `NA`, both of which are based on state-of-the-art methods. We follow up with experimental evaluation.

4.8.1 Location Adversary

The Location Adversary (`LA`) has access to locations shared by users and tries to infer whether a pair of users are socially related. Using check-in information for inference of users' social relations has been extensively studied. We adopt the state-of-the-art method *walk2friends*[8] that has been shown to outperform other approaches.

The *walk2friends* method can be explained in three steps: First, we organize users and locations into a bipartite graph, with the number of visits by a user to a location as the weight of the corresponding edge. Second, multiple random walks are performed on the graph to generate a set of walk traces with the transition probability from one node to another defined over the edge weight. Finally, we utilize the skip-gram model [73, 74] to embed each user into a continuous vector. The vector for a user u, is denoted by $\vec{E_u}$.

To perform friendship prediction, we follow the image and text attacks above. Concretely, for two users u and v, we define the following feature vector $\vec{x}^L_{\{u,v\}} = (\text{cosine}(\vec{E_u}, \vec{E_v}),\ \text{euclidean}(\vec{E_u}, \vec{E_v}),\ \text{correlation}(\vec{E_u}, \vec{E_v}), \text{chebyshev}(\vec{E_u}, \vec{E_v}),$ $\text{bray\_curtis}(\vec{E_u}, \vec{E_v}),\ \text{canberra}(\vec{E_u}, \vec{E_v}), \text{manhattan}(\vec{E_u}, \vec{E_v}),\ \text{sq\_Euclidean}(\vec{E_u}, \vec{E_v}))$ and train a random forest classifier to perform our attack.

4.8.2 Network Adversary

Our network adversary (`NA`) has access to a limited fraction of relationships in the online social network and tries to predict these missing relationships. This is inline with the traditional setting of link prediction. We leverage state-of-the-art tool *node2vec* to implement `NA` [38], which has been shown to outperform traditional methods, such as Jaccard Index, Adamic-Adar score, and Preferential Attachment.

node2vec first performs random walk on the partial social network, this again results in a set of truncated random walk traces.[‡] Then, skip-gram is adopted to embed each user u into a vector $\vec{F_u}$. For two users u and v, we define their feature vector as the Hadamard product between their vectors [38], i.e., $\vec{x}^E_{\{u,v\}} = \vec{F_u} \circ \vec{F_v}$.

Table 4.5: Pre-processed location and network data statistics

(a)

	LA	NY
users	10,430	19,038
locations	23,452	30,073
friend pairs	17,043	40,883

(b)

	LA	NY
users	9,347	17,853
friend pairs	16,294	39,756

[‡]We follow the default parameter setting of *node2vec* in the random walk.

4.8.3 Evaluation

We evaluate the attacks in the same way as described previously, i.e. using the AUC metric for a random forest classifier and assigning 80% of each of the friend and stranger pairs for training and 20% of each of the friend and stranger pairs for testing.

4.8.3.1 Data Preprocessing

For the Location Adversary, following [8], we filter out users who have not visited at least two *different* locations since such accounts with many check-ins at one location are most likely to be local businesses for example bars and pubs. We further filter out users with less than 20 check-ins, whom we consider to be *active users*. We obtain 17,043 friends pairs from LA and 40,883 from NY.

For the Network Adversary, we start with the friend pairs generated for the hashtag, location, image, and text based attacks as described above. Since we perform 5 fold cross validation, we use 80% of the friend pairs to generate a partial network graph $\mathcal{G}'$ which we use for building the *node2vec* embedding for users. These set of edges in our training set is a proper subset of the original graph $\mathcal{G}$. We obtain 16,294 friends pairs from LA and 39,756 from NY. For both attacks, we randomly select an equal number of stranger pairs out of the set of already sampled stranger pairs for the hashtag, image, and text attacks. Tables 4.5a and 4.5b present the pre processed data statistics for both attacks. We repeat the feature generation process (random walk followed by embedding generation and hadamard distance calculation) 5 times with a 80-20 split, leaving a different set of 20% pairs for testing each time. We further shuffle the samples and repeat the 5 fold cross validation 5 times overall.

4.8.3.2 Results

We achieve AUC values of 0.77 in average with standard deviation of 0.005 in LA and 0.76 with 0.003 in NY, for the location based monomodal attack. For the network based mononmodal attack, we achieve a mean AUC value of 0.58 for both cities and a standard deviation of 0.004 and 0.007 for LA and NY respectively .

4.9 Multimodal Attack

We now present the Multimodal Adversary $\mathcal{M}$, which leverages information from all 5 components of user posts presented in the previous sections, to infer social relationships. We first present the attack methodology along with a variant called the subset adversary. We follow up with an empirical evaluation as well as robustness of the attacks against information hiding.

Overview of the Fusion Mechanism The previous attacks were based on training a machine learning classifier which required a feature design process (hand-engineered or aided by deep learning techniques). For this attack, we use the features already designed for the 5 individual post components as described in each previous attack section. However, we weigh the information from each of the 5 components according to their relative trustworthiness, which we infer from the strength of the individual attacks.

To this end, we divide the training set $V_{\textbf{train}}$ into 2 sets. We use one set to train the 5 random forest classifiers using the 5 respective feature vectors. We use the second set to assign a confidence score a to each of the 5 trained classifiers using their respective AUC values. We use these AUC values as indicators of the strength or trustworthiness of the predictions made by the 5 classifiers on the target set $V_{\textbf{tar}}$.

4.9.1 Attack Methodology

We describe this attack in 4 stages.

In the first stage, we divide the training set $V_{\textbf{train}}$ into 2 parts: $V_{\textbf{train-train}}$ and $V_{\textbf{train-test}}$. We train the classifiers for the five modalities separately using the feature vectors $\vec{x}^H_{\{u,v\}}, \vec{x}^T_{\{u,v\}}, \vec{x}^I_{\{u,v\}}$, $\vec{x}^L_{\{u,v\}}$ and $\vec{x}^E_{\{u,v\}}$ on the set $V_{\textbf{train-train}}$.

In the second stage, from each of the 5 trained random forests, we extract the respective posterior probabilities for the friend class, i.e. the probability with which a sample pair in the set $V_{\textbf{tar}}$ is socially related. In a random forest, the predicted posterior probabilities of an input sample are computed as the mean predicted class probabilities of the trees. The class probability of a single tree is the fraction of samples of the same class in a leaf.

Let x^H, x^T, x^I, x^L and x^E denote the posterior probabilities for the friend class for a sample $\{u, v\} \in V_{\textbf{tar}}$ as predicted by the classifier trained using hashtag, text, image, location and friendship information respectively. Therefore we have for $D \in \{H, T, I, L, E\}$, $x^D_{\{u,v\}} = \Pr(\mathcal{R}_{\{u,v\}} = 1|\vec{x}^D_{\{u,v\}})$, where $\mathcal{R}_{\{u,v\}}$ is the random variable representing the relationship between the user pair $\{u, v\}$.

In the third stage, we calculate a confidence score a for each of the 5 attack components above. Let a^H, a^T, a^I, a^L and a^E denote the confidence scores of the hashtag, text, image, location and friendship attacks respectively. We use the Area under the ROC curve (AUC) score between the true class labels and the predicted posterior probabilities for the set $V_{\textbf{train-test}}$ for the friend class as the confidence score. Therefore we have for $D \in \{H, T, I, L, E\}$ and $\{u, v\} \in V_{\textbf{train-test}}$,

$$a^D = \texttt{AUC}(y_{\{u,v\}}, x^D_{\{u,v\}})$$

where y denotes true class labels and $x^D_{\{u,v\}}$ denotes the predicted posterior probabilities.

In the fourth and final stage, we perform a weighted averaging of the 5 posterior probabilities for each sample $\{u, v\} \in V_{\textbf{tar}}$ using their normalized confidence scores. We obtain the final score $s^{\mathcal{M}}_{\{u,v\}}$ for a sample $\{u, v\} \in V_{\textbf{tar}}$ for the Multimodal Adversary $\mathcal{M}$ as,

$$s^{\mathcal{M}}_{\{u,v\}} = \sum_{D \in \{H,T,I,L,E\}} x^D_{\{u,v\}} * \frac{a^D}{\sum\limits_{D \in \{H,T,I,L,E\}} a^D}$$

This score represents the likelihood that $\{u, v\}$ are friends.

4.9.2 Subset Adversaries

We further analyze how various combinations of individual post components perform towards inferring friendship between OSN user pairs. To this end we model a reduced

Multimodal Adversary which uses a proper subset of the set of texts, hashtags, images, locations and friendship-edges published by OSN users, such that the adversary does not use at least 1 component out of the 5 post components. We call such an adversary that uses a subset D' of post components, a Subset Adversary $\texttt{SA}^{D'}$. The adversary uses any combination of a minimum of 2 upto a maximum of 4 out of the 5 post components, thus we have 25 different possible combination. Therefore we calculate the corresponding score $s^{D'}_{\{u,v\}}$ as:

$$s^{D'}_{\{u,v\}} = \sum_{D \in D'} x^{D}_{\{u,v\}} * \frac{a^D}{\sum\limits_{D \in D'} a^D}$$

where $D' \subset \{H, T, I, L, E\}$ and $2 \leq |D'| \leq 4$

4.9.3 Evaluation

We first present a baseline approach which we design to evaluate the multimodal attack. We then describe the dataset and evaluation metric. We then discuss the results, followed by an analysis of the interplay between the different modalities. We end the section with an analysis of the robustness of the attack against the amount of data present in the OSN.

4.9.3.1 Baseline

We implement a baseline that performs a simple averaging of the posterior probabilities from each data component. Thus the baseline score $s^{BL}_{\{u,v\}}$ for a sample $\{u, v\} \in V_{\mathbf{tar}}$ with posterior probabilities $x^{D}_{\{u,v\}}$ where D denotes the individual data component which is used by each attack is defined as $s^{BL}_{\{u,v\}} = \frac{\sum_{D \in \{H,T,I,L,E\}} x^{D}_{\{u,v\}}}{|D|}$

4.9.3.2 Dataset

We use the same dataset as for the individual attacks. As mentioned earlier, we obtained 17,043 friend pairs for Los Angeles and 40,883 friend pairs for New York. We collect the stranger pairs sampled in the hashtag, image, and textual attacks. Each pair in this resultant dataset for the Multimodal Adversary has feature values for at least one out of the set $\{\vec{x}^{H}_{\{u,v\}}$ (hashtag), $\vec{x}^{I}_{\{u,v\}}$ (image), $\vec{x}^{T}_{\{u,v\}}$ (text), $\vec{x}^{L}_{\{u,v\}}$ (location) and $\vec{x}^{E}_{\{u,v\}}$ (partial network) $\}$.

4.9.3.3 Metric

Unlike the monomodal attacks, for the multimodal attack, we only calculate a score $s_{\{u,v\}}$ for each sample pair in our target set instead of designing a set of features. The score $s^{\mathcal{M}}_{\{u,v\}}$ represents the likelihood that $\{u, v\}$ are friends, i.e.

$$f(x^{\mathcal{M}}_{\{u,v\}}) = s^{\mathcal{M}}_{\{u,v\}} \overset{?}{>} t$$

for a threshold t. Experimenting with different values of t gives us a range of false and true positive values. Plotting the false-positive rate on the x-axis and the true-positive

Table 4.6: Performance (mean and standard deviation of AUCs) of the subset adversaries (multimodal attacks using size-2, size-3 and size-4 subsets of the 5 post components) for Los Angeles and New York. The best performance for subsets of each size is marked in bold.

(a) Attacks using 2 components

Data subset	Los Angeles	New York
$\{\boldsymbol{T}, \boldsymbol{L}\}$	**0.897** $\pm$0.003	**0.874** $\pm$0.002
$\{T, E\}$	0.828 $\pm$0.004	0.845 $\pm$0.003
$\{L, E\}$	0.836 $\pm$0.005	0.772 $\pm$0.003
$\{H, T\}$	0.822 $\pm$0.004	0.840 $\pm$0.003
$\{H, I\}$	0.767 $\pm$0.007	0.764 $\pm$0.003
$\{H, L\}$	0.805 $\pm$0.005	0.796 $\pm$0.002
$\{H, E\}$	0.673 $\pm$0.006	0.682 $\pm$0.003
$\{T, I\}$	0.819 $\pm$0.004	0.835 $\pm$0.003
$\{L, I\}$	0.836 $\pm$0.004	0.771 $\pm$0.003
$\{I, E\}$	0.626 $\pm$0.006	0.652 $\pm$0.004

(b) Attacks using 3 components

Data subset	Los Angeles	New York
$\{\boldsymbol{T}, \boldsymbol{L}, I\}$	**0.901** $\pm$0.003	**0.876** $\pm$0.002
$\{\boldsymbol{T}, \boldsymbol{L}, E\}$	**0.900** $\pm$0.003	**0.878** $\pm$0.002
$\{\boldsymbol{T}, \boldsymbol{L}, H\}$	**0.899** $\pm$0.003	**0.875** $\pm$0.002
$\{H, L, E\}$	0.853 $\pm$0.005	0.805 $\pm$0.003
$\{H, T, I\}$	0.819 $\pm$0.004	0.836 $\pm$0.003
$\{H, T, E\}$	0.831 $\pm$0.004	0.846 $\pm$0.003
$\{H, I, L\}$	0.853 $\pm$0.004	0.802 $\pm$0.003
$\{H, I, E\}$	0.710 $\pm$0.006	0.731 $\pm$0.003
$\{T, I, E\}$	0.833 $\pm$0.004	0.850 $\pm$0.003
$\{I, L, E\}$	0.846 $\pm$0.004	0.784 $\pm$0.003

(c) Attacks using 4 components

Data subset	Los Angeles	New York
$\{\boldsymbol{T}, \boldsymbol{L}, I, E\}$	**0.904** $\pm$0.003	**0.881** $\pm$0.002
$\{\boldsymbol{T}, \boldsymbol{L}, H, E\}$	0.901 $\pm$0.003	0.879 $\pm$0.002
$\{\boldsymbol{T}, \boldsymbol{L}, H, I\}$	0.902 $\pm$0.003	0.876 $\pm$0.002
$\{H, T, I, E\}$	0.836 $\pm$0.004	0.851 $\pm$0.003
$\{H, I, L, E\}$	0.863$\pm$0.004	0.816 $\pm$0.003

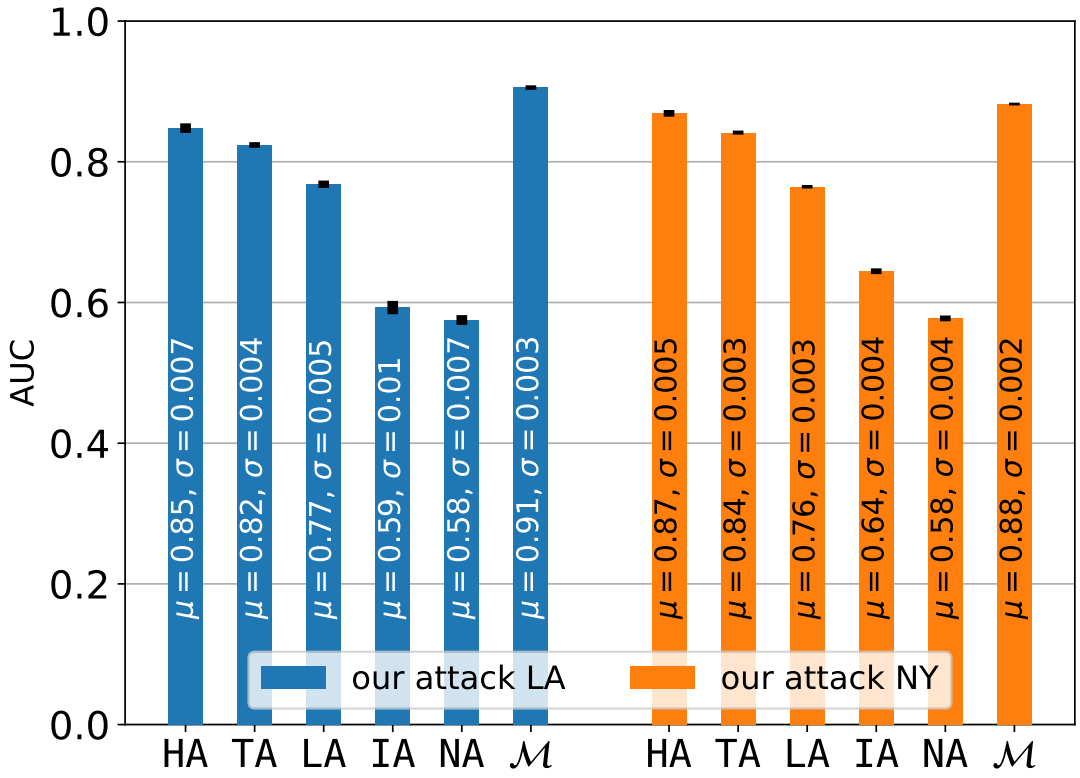


Figure 4.7: Comparison of the AUC values (mean and standard deviation) across all folds and iterations achieved by all 6 adversaries (the Hashtag (`HA`), Image (`IA`), Text (`TA`), Location (`LA`), Network (`NA`) and the Multimodal Adversary $\mathcal{M}$) for Los Angeles in blue and New York in orange.

rate on the y-axis produces the ROC (Receiver Operating Characteristic) curve. We use the area under this curve (AUC), to analyze the success of the attack. Unlike other metrics, AUC summarizes the performance in a straightforward manner: 0.5 is as bad as random guessing and 1.0 indicates perfect performance. AUC also exhibits a number of desirable properties when compared to accuracy such as decision threshold independence and invariance to a priori class probabilities. We assume the attacker to know the best threshold. Nevertheless we also report the precision and recall for a threshold that corresponds to the percentage of friends in the training set.

4.9.3.4 Results

We now discuss the results of our multimodal and subset adversaries. We perform a 5-fold cross validation, with different target pairs each time and shuffle and repeat the 5-fold cross validation 5 times. For each attack, we report the mean and standard deviation.

Multimodal Adversary $\mathcal{M}$. For our Multimodal Adversary using the weighted average score $s^{\mathcal{M}}_{\{u,v\}}$ as described above, we achieve a strong performance in both cities

with a mean AUC of 0.91 (standard deviation of 0.003) for Los Angeles and 0.884 (standard deviation of 0.002) for New York. For the baseline approach, using the simple average score $s^{BL}_{\{u,v\}}$, we achieved an AUC of 0.899 on average with standard deviation of 0.003 for Los Angeles and 0.880 with a standard deviation of 0.002 for New York.

We also compare the multimodal attack with the 5 monomodal attacks in Figure 4.7. The multimodal attack achieves the strongest performance for both cities. Moreover, the attacks using only hashtags and only text significantly outperform the attacks using only images, only locations and only friendship edges.

Let p denote the percentage of true friends in a training set. We label the top $p\%$ pairs ranked by their scores $s^{\mathcal{M}}_{\{u,v\}}$, as positive (predict as friends). At thresholds chosen this way, both precision and recall are the same: an average of 0.817 for Los Angeles with standard deviation of 0.005 and an average of 0.807 for New York with the same standard deviation.

Subset Adversaries $\texttt{SA}^{D'}$. Table 4.6 presents the AUCs of size-2, size-3 and size-4 subset adversaries. The best performance for subsets of each size is marked in bold. Among all subset sizes, the subsets containing both text and locations perform the best overall in both cities.

The best performing size-4 subset consists of texts, locations, friendship edges and images with a mean AUC of 0.904 for Los Angeles and 0.881 for New York. The best performing size-3 subset for Los Angeles consists of texts, locations and images with a mean AUC of 0.901 and for New York consists of text, locations and friendship edges with a mean AUC of 0.878. The highest performance among size-2 subsets is achieved by the combination of text and locations (means of 0.897 for Los Angeles and 0.874 for New York).

The worst performance in both cities is seen in the attacks combining images and friendship-edges. This result is not surprising given that the individual attacks using only image have the worst performance for both cities and for Los Angeles, the attack using only friendship-edges is the second worst.

4.9.3.5 Interplay between Modalities

We now discuss some interesting observations on the effect of different modalities on each other.

Complementary Components. The diagonal elements in Figure 4.8 further show the AUCs of the monomodal adversaries and the non-diagonal elements show size-2 subset adversaries. The greener elements show increasing AUCs and the redder elements show decreasing AUCs. We see from Figure 4.8 that certain components complement each other e.g. for the Los Angeles dataset, friendship edges and images on their own achieve very low AUCs of 0.58 and 0.59 respectively but when combined together the AUC goes up to 0.63! Location achieve an AUC of 0.77 on its own, but when it is combined with images, the AUC shoots to 0.84! Yet another example is text which achieves an AUC of 0.82 on its own but together with locations achieves a much higher AUC of 0.90! The same patterns can be seen for the New York dataset. These examples demonstrate that our fusion mechanism for the Multimodal Adversary exploits these complementing components. Hashtag however does not perform well in combination

	H	T	L	I	E
H	0,87				
T	0,84	0,84			
L	0,80	0,87	0,76		
I	0,76	0,84	0,77	0,64	
E	0,68	0,85	0,77	0,65	0,58

	H	T	L	I	E
H	0,85				
T	0,82	0,82			
L	0,80	0,90	0,77		
I	0,77	0,82	0,84	0,59	
E	0,68	0,83	0,77	0,63	0,58

Figure 4.8: Matrix showing AUCs (average across all folds and iterations) achieved by complementary components (size-2 subset adversary) left: New York, right: Los Angeles.

with any other component for both cities. We investigated by looking at examples of friend pairs where the hashtag only classifier gave high probabilities for the friend class and found that for many of those pairs, at least one of the other components had high probabilities for the stranger class. For example one friend pair used some hashtags for a social event they went to in New York, however they actually lived in different continents and one of them was a food blogger whereas the other one didn't post about food at all.

Combination of Weaker Components. The monomodal attacks using images, locations and friendship edges showed the weakest performances individually. However, when combined with only 1 additional component, their performance improves tremendously in all cases. For Los Angeles, the size-3 subset of images, location and edges (mean AUC of 0.84) and even the size-2 subsets of location and edges and location and images (both having mean AUC of 0.83), significantly outperform the AUC achieved by the best of them individually i.e. 0.77 by location. Results for New York follows the same patterns.

Thus we deduce that even though some components are clearly outperformed by others when used individually, they may contribute significantly towards a much stronger inference attack when combined together with other weak components.

Increasing Number of Components, In a majority of the cases, we see from Table 4.6 that combining additional components improves the attack success. For example adding one component to either of the size-3 subsets $\{H, T, I\}$, $\{H, I, L\}$, $\{T, I, L\}$ and $\{H, T, L\}$ to get the size-4 subset $\{T, L, H, I\}$ always improved the AUC. The same can be observed for other size-3 to size-4 transitions such as $\{T, L, H\}$, $\{H, T, E\}$ or $\{H, L, E\}$ to get $\{T, L, H, E\}$ as well as size-2 to size-3 transitions such as $\{T, E\}$, $\{T, L\}$ and $\{E, L\}$ to $\{T, L, E\}$. This underlines the effectiveness of our proposed fusion mechanism for the Multimodal Adversary. Further, the success of the attacks continue to increase consistently across cities as well.

Thus, increasing the number of components increases the attack success for almost all cases. Moreover, the Multimodal Adversary $\mathcal{M}$ using all 5 modalities does not have a significant increase in performance over the best performing size-3 or the size-4 subsets adversaries. This can be explained by the impact of the relatively weak image and network adversaries, which are unable to contribute to the inference attack anymore beyond the performance achieved by the remaining components.

We summarize our key observations as follows:

1. Various pairs of components complement each others' attack success when combined.

2. A combination of weaker components often increased the attack performance more than a combination of stronger components.

3. Increasing the number of components almost always increase the attack success.

4.9.4 Robustness

So far we have seen the severity of privacy risks arising from sharing various types of posts on an OSN. An immediate question that arises at this point is: Is the problem solved when users simply shared posts less frequently, how is the performance is affected if users shared a lower number of posts. We therefore evaluate the robustness of the four attacks namely the attacks using hashtags, texts, images and a combination of all three of them in the multimodal setting. Robustness of the Location and Network Adversaries has been extensively studied in prior work that proposed *walk2friends* and *node2vec*.

We first randomly remove a certain proportion of posts published by users and collect the images, hashtags, texts contained in the remaining posts. Then we recompute the corresponding feature values for the training set of user pairs with which we then train three random forest classifiers, one each for hashtags, texts and images. We also use these three feature sets to carry out the multimodal attack on the subset $D' = \{H, C, I\}$ as described in table 4.6. We begin by removing 10% of the original number of posts in either dataset and increase the amount of posts to be removed in steps of 10% until 50% of the original number of posts are removed. The selection of posts to be removed is random.

Figure 4.9 shows the robustness of the attacks against removal for New York. The y-axis shows the AUC attained at various percentage of posts removed shown in the x-axis. Results for Los Angeles follow a similar trend and can be found in the appendix

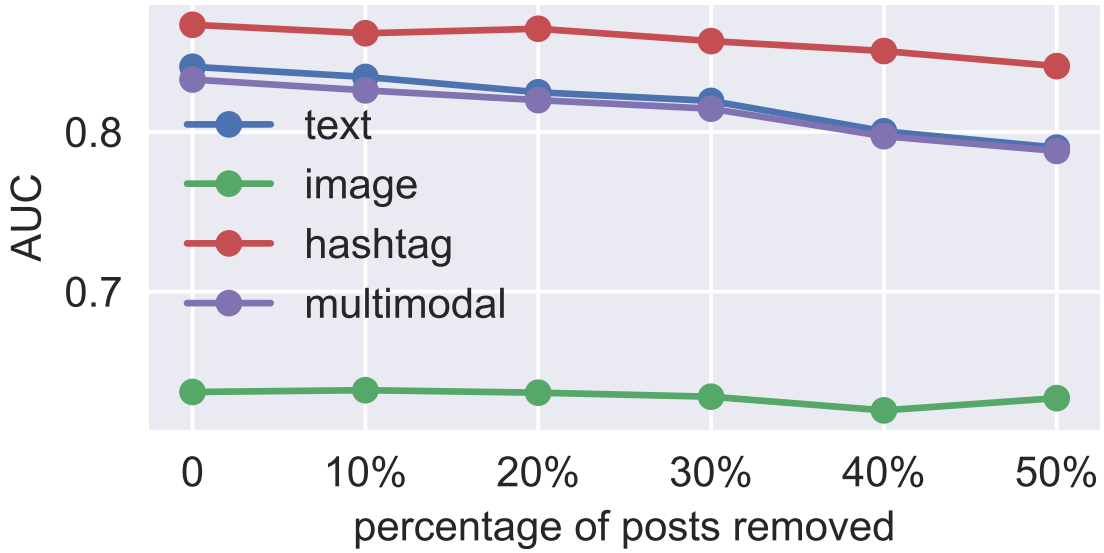


Figure 4.9: Robustness of our attacks in the New York dataset against removing posts.

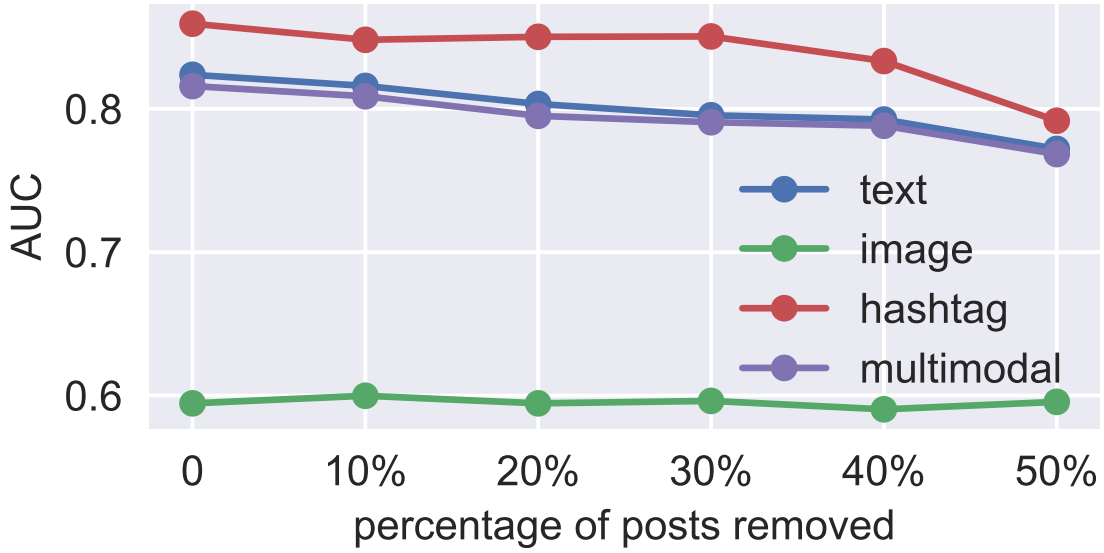


Figure 4.10: Robustness of our attacks in the LA dataset. The y-axis shows the AUC attained at various percentage of posts removed shown in the x-axis

in Figure 4.10. We see that the performance drops gradually for all attacks except the attack using images only. However the performance does not degrade significantly even with 50% of the posts removed from the dataset. This shows that the attacks are quite robust to the amount of data present in the OSN. This calls for advanced defense mechanisms against adversaries that utilize multiple kinds of user information on OSNs for preserving both privacy as well as utility.

4.10 Discussion and Future Work

Our main goal is to show whether a combination of multiple modalities improves the strength of an inference attack. The three individual monomodal attacks, i.e., text, images, and hashtags, presented in this paper are first of its kind, and use the most popular and standard techniques. However, they may be further optimized using newer techniques. Secondly, some of the unsupervised methods work almost as good as the supervised attacks, which demonstrates the power of the features we proposed. Thirdly, our attack may results in false positives in case of users who are not socially related but post about similar, popular or trending topics or events. However, our attacks eliminate this effect, by favoring hashtags with low entropy and using TF-IDF for text.

We demonstrated that our attacks are robust to the amount of information shared by users. This highlights potential future work in analysis of the attack performance by obfuscating components or posts whose components have the highest contribution in predicting friendship. A recent work AttriGuard [52] takes an adversarial example based approach towards classification evasion to defend against users location (city) inference. However, these defenses are only on the personal attribute level and uses only type of public data (user's app rating scores in case of AttriGuard), whereas our attack deals with multimodal data shared by the user (thousands of images, captions, hashtags, locations). Applicability of evasion or poisoning attacks requires further research. Secondly, we demonstrated that the privacy threat is equally severe in datasets from two different cities. The same framework can be used to analyse data collected from other OSNs or other cities all around the world. Finally, we only provide a lower bound on the performance of the attacks. Considering factors like the date or time when 2 users shared certain posts can potentially make the attacker stronger.

4.11 Conclusion

We present novel attacks on the privacy of social relationships using hashtags, images and caption data. We also evaluate state-of-the-art tools to infer friendships using locations and friendships (between other pairs) shared on OSNs. We finally focus on adversaries that use not just one type of user information but combine all available types of information for the inference attack. We demonstrate that the success of the friendship inference attacks are greatly amplified when multiple modes are combined, for example in the LA dataset, our multimodal attack achieves an AUC of 0.92 whereas the maximum AUC achieved by any of our monomodal attacks is only 0.8555 (using hashtags). Our experiments on the subset adversaries demonstrate that although some components clearly outperform others when used individually, they contribute towards a much stronger inference when combined together with additional components. The fusion mechanism for our multimodal adversary is capable of exploiting post components that complement each other.

We demonstrate that our attacks are robust to the amount of information shared by users. As newer data types or trends will emerge in the world of OSNs in future, the multimedia footprints of users would get increasingly diverse. The threat to user privacy arising from such multimodal adversaries would potentially be much larger.

Development of effective defense mechanisms or privacy advisors is therefore the need of the hour.

5

You Are How You Walk

Quantifying Privacy Risks in Actimetry Data

5.1 Motivation

In the current era of the Internet of Things, extensive amounts of data are generated by users not just from internet browsing and social media but also smart devices like wearables, cars or even household appliances. While some data is shared willingly and purposely by the users themselves (e.g. social media posts), some data is shared unknowingly (e.g. web browser telemetry, tracking pixels). Yet some other data, such as biomedical or genomic data is shared only with a trusted party for a specific purpose (e.g. health situation monitoring and improvement, activity tracking and analysis, dietary and lifestyle advice/consultation). This has led to huge interest in privacy concerns arising out of such data. Recently, acts like the General Data Privacy Regulation or the California Consumer Privacy Act have imposed strict regulations for collecting, processing and sharing user data.

Certain types of data are classified as personal or even sensitive personal*. However, many kinds of biomedical data (e.g. step counts, heart rate, SpO2, nutrition and sleep data) are in the gray zone, which are considered sensitive personal information only under special circumstances when they indicate diseases or disabilities. In case of biomedical services, the user typically does not have fine grained control over the data shared, i.e., it is an all or nothing scenario. Such data is extremely sensitive, as it can expose other information about the user, such as their medical condition, ethnicity, habits, kinship or financial status. Therefore, biomedical data needs to be treated with extreme caution by the service provider. Especially in the recent years, wearable devices have gained huge popularity and become inseparable parts of people's lives. Data collected from these devices is used for a plethora of purposes by multiple parties, such as private companies, health organizations and government agencies. On the other hand, this gives rise to increasing concerns over privacy of the user.

In two recent works [53, 54] Jiang et al. devise a new deep neural network model to predict age and gender from pedometer data. The authors use number of steps made each day collected over 259 consecutive days and are able to find different walking behaviors during week days, weekends and holidays. If an attacker wanted to use such a technique for attribute inference, they would need data collected over hundreds of days, which is not very practical.

In this paper, we perform the first, extensive study of privacy risks arising from fine grained user's step count data collected over much shorter time spans such as one week or even a single day. Such data is collected by various activity trackers (e.g. Endomondo, FitBit, Apple's Health), but can also be collected from the accelerometer on smartphones either by an app or an opened website. Such accelerometer data is not considered sensitive and therefore can be collected without explicit permission from the user. While concerns about privacy of walking patterns data were raised before †, to the best of our knowledge, a systematic in-depth analysis of the privacy issues has not been performed yet.

*https://www.burges-salmon.com/news-and-insight/legal-updates/gdpr-personal-data-and-sensitive-personal-data/

†https://w3c.github.io/sensors/#user-identifying

5.2 Contributions

We perform the first large-scale study of the privacy issues within fine grained user's step count data. In particular, we design various attribute inference and linkability attacks. For the attribute inference attack, we assume that the adversary has access to the step count data of the target user for example collected from a smartphone. The adversary then tries to infer certain personal attributes to which he normally would not have access. For the linkability attack, the adversary has access to target user's step count data as well as an anonymized database containing further sensitive information along with the step count data collected at a certain point in time. The adversary then tries to link the target user with the record in the database. This could have a wide variety of implications like targeted advertisements, surveillance, unfair credit score and pricing by insurance companies or banks, to name a few. We demonstrate the performance of the attacks with an extensive evaluation on a real world dataset of 1000 participants.

We make the following key contributions:

- In order to leverage the power of deep learning, we develop three conceptually different feature extraction methods, namely, statistical, distributions and autoencoders.
- Our attribute inference attack is aimed at finding the correlation between the walking patterns of the user and three personal attributes namely gender, age and education. We find that gender and age can be inferred with a high confidence, while education does not show a strong correspondence with our data.
- We also run an ensemble version of our attribute inference attack by splitting the user step data into actions based on active and non-active periods. We classify each action separately and infer the users' attribute based on all the actions of a user.
- We make a comparison between all the feature selection methods and classifiers on all the attribute inference tasks.
- We run three different types on linkability attacks. The first one is unsupervised and it relies on the distance (under some metric) between feature vectors of different samples. The second attack uses a pairwise vector similarity metric between features of different samples to fit a traditional machine learning classifier, namely, random forest. The third one uses the One Shot Learning method with Siamese networks.

5.2.1 Organization

The chapter is organized as follows: We introduce our dataset in Section 5.3. In Section 5.4 we describe how we extract features from the raw data. Next in Section 5.5 we present our attribute inference attack followed by the experimental evaluation in Section 5.6. In Section 5.7 we present our user linkability attack, followed by its experimental evaluation in Section 5.8. Finally we conclude our paper in Section 5.9.

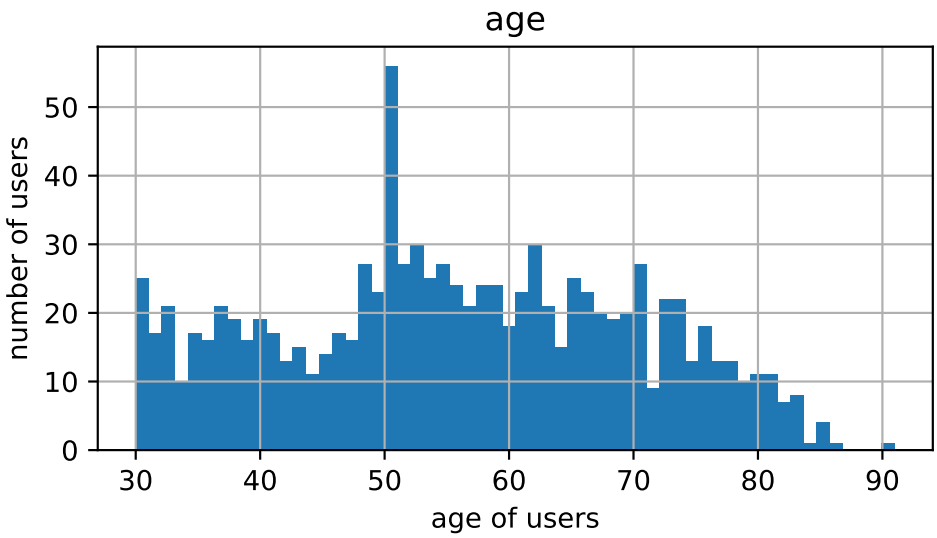


Figure 5.1: Distribution of age in our dataset

	Users	997
gender	Males	44.13%
	Females	55.76%
education	High	52.35%
	Medium	45.13%
	Low	1.80%

Table 5.1: Statistics of our dataset

5.3 Dataset

Our step count dataset was collected by a medical organization among inhabitants of a middle-size German city.[‡]

The data was collected using an ActivPal Sensor, a small device worn on the thigh , which was carried by users for 7 consecutive days. This can, to a good extent, simulate step count data collected from any other wearable device (e.g. smartwatch) or from a phone kept in a pocket throughout the day.

5.3.1 Data description

Our raw dataset consists of number of step in each 15s period, for 1000 participants. We exclude 3 users due to incomplete data. Each user has age, gender and education attributes.

[‡]We omit the detail of the organization for complying with the anonymous submission policy, and will disclose it upon acceptance.

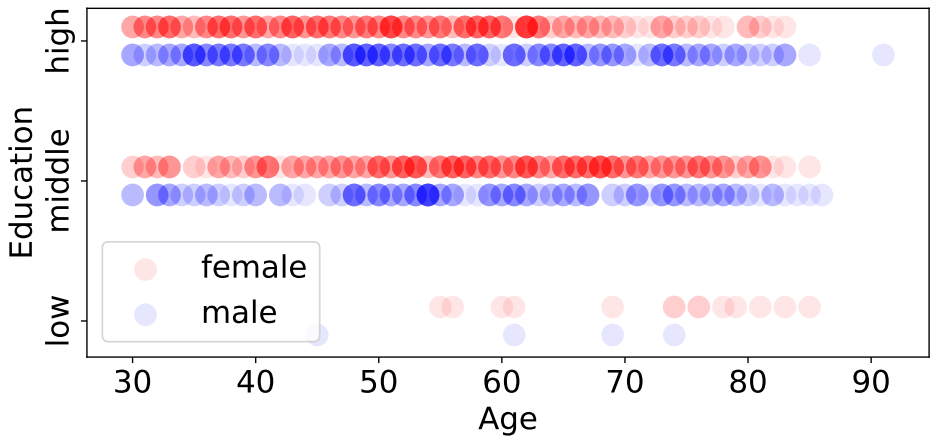


Figure 5.2: Distribution of users between all three attributes, the darker the color the more users with identical attributes

	age	education
education	-0.209	
gender	-0.034	-0.130

Table 5.2: Correlation coefficients between all three attributes of users

The age of participants in our dataset ranges from 30 to 91 years old. Figure 5.1 shows the distribution of users of different ages in our dataset. We divide users into two classes based on the median age (55 years).

Table 5.1 shows the percentage of users in each class by gender and education. For education there are 3 levels:

- **Low.** Early childhood education up to lower secondary education.
- **Middle.** Upper secondary education up to Bachelor degree or equivalent level.
- **High.** Master degree or equivalent level up to Doctoral degree or equivalent level.

For the classification task of education, we discard users with low education, since they comprise only 1.8% of all participants.

Figure 5.2 shows distribution of users among all 3 attributes. You can notice weak negative correlation of age and education. To measure it we encode education as $0, 1, 2$ for low, medium, high respectively and gender as 0 for male and 1 for female. Table 5.2 confirms our observations.

Figure 5.3 shows the result of performing Principal Component Analysis (PCA) on users' raw step counts. The two colors denote the two classes for each attribute. We note that PCA on raw data does not suffice to separate users by gender, age or education.

Therefore, in the next section we present methods to preprocess this raw data in order to extract meaningful features for our tasks.

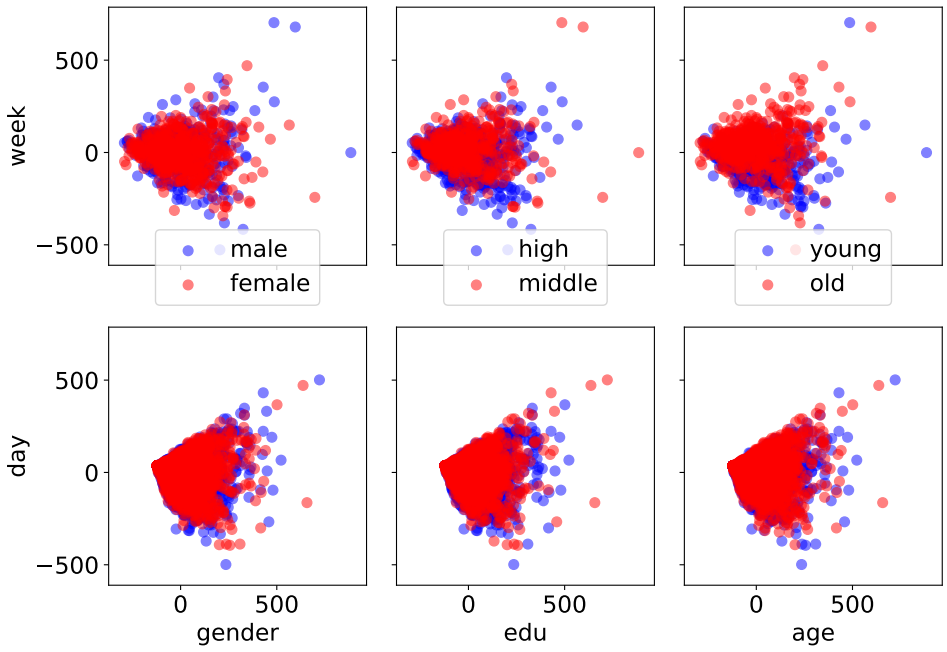


Figure 5.3: PCA of users raw step counts over the whole week (top) and split over 7 days (bottom) for each of our 3 attributes

5.3.2 Ethical Considerations

We note that our primary institution does not provide an IRB nor mandate (or enable) approval for such experiments. Meanwhile, the organization who collected the data followed the standard protocol for all the ethical compliance. We also store the anonymized data encrypted in an independent server with access only from white listed internal IP addresses and during experiments, we strictly follow the data processing guideline defined by the data collecting organization.

5.4 Feature Extraction

In this section we describe our three conceptually different feature extraction methods, followed by three different normalization techniques we use.

We refer to the sequence of the number of steps made by a user u every 15 seconds, as the raw step count vector, $\vec{s}_{raw,u}$. These raw step count vectors $\vec{s}_{raw,u}$ are time series and thus are hard to work with when directly used as feature vectors for machine learning tasks. Most standard machine learning techniques learn based on features that should be comparable between data points, which is not the case for time series. A small displacement of an event in time would result in a feature vector much different from the original one (e.g., by cosine similarity). For example if a user begins a morning walk even five minutes later than usual, the beginning and the end of the walk are now different features and this cannot be captured by traditional machine learning models.

To this end, we use different feature extraction methods as described next.

For a user u, we collectively refer to the set of step count feature vectors from different days by S_u, and a feature vector for a day d as $\vec{s}_u^d$. These $\vec{s}_u^d$ can either be the raw step counts or features created according to one of the methods below or their normalized versions. We omit subscript u when it is clear that we are talking about one specific user. Similarly, we omit the superscript d when the d is irrelevant.

5.4.1 Statistical Method

For each user, we first split $\vec{s}_{raw}$ into smaller, non overlapping subvectors of a defined window size w (the last subvector might be smaller). Then we calculate basic statistical values, which reflect different characteristics of a user's walking behavior: sum (how much the user walks), maximum (highest speed of the user), mean (how fast the user walked on average), median (around what speed did user use most of the time) and standard deviation (how much acceleration / deceleration did user do) on each subvector and combine all the calculated values in a single output vector $\vec{s}_{stat}$. We use different subsets of statistics and different values of w in our experiments.

Example 5.4.1. *Let* $\vec{s}_{raw} = (5, 0, 0, 2, 3, 4, 3, 0), w = 3$*,* $stat = \{sum, mean\}$*. We split* $\vec{s}_{raw}$ *into three subvectors* $(5, 0, 0)$*,* $(2, 3, 4)$ *and* $(3, 0)$*. Thus* $\vec{s}_{\{sum,mean\}} = (sum(5, 0, 0), mean(5, 0, 0), sum(2, 3, 4), mean(2, 3, 4), sum(3, 0), mean(3, 0)) = (5, 1.67, 9, 3, 3, 1.5)$*.*

5.4.2 Distributional Method

We want to capture a distribution of steps done in every 15s period. We also want to retain the information about time of a day when users walk more or less. Therefore, we split the $\vec{s}_{raw}$ into subvectors, calculate distributions on them and finally concatenate them.

Precisely, we find the maximal number of steps *max_steps* any user makes in 15s. Then, for each user, we split $\vec{s}_{raw}$ into smaller, non overlapping subvectors of a defined window size w (the last subvector might be smaller). We group all possible number of steps $0, 1, 2, \ldots,$ *max_steps* into buckets of size b. Then for each resulting vector, we count the number of occurrences of steps in each bucket. Since 0 steps is the most common event, we have a bucket containing just 0 steps, and further buckets containing b different number of steps. We combine the calculated value counts into a single output vector $\vec{s}_{dist}$.

Example 5.4.2. *For* $\vec{s}_{raw} = (5, 0, 0, 2, 3, 4, 3, 0)$*,* $w = 3$*,* $b = 3$*,* $max\_steps = 6$*. We split* $\vec{s}_{raw}$ *into three subvectors* $(5, 0, 0)$*,* $(2, 3, 4)$ *and* $(3, 0)$*. Each subvector will have three buckets, i.e.,* $\{0\}, [1, 3]$*, and* $[4, 6]$*. For the first subvector* $(5, 0, 0)$*, we have* 2 *instances in the first bucket, i.e.,* $\{0\}$*,* 0 *instances in the second bucket, i.e.,* $[1, 3]$ *and* 1 *instance in the third, i.e,* $[4, 6]$*. Analogous calculations are done for the next two subvectors. The resulting feature vector is then* $\vec{s}_{dist} = (2, 0, 1, 0, 2, 1, 1, 1, 0)$*.*

5.4.3 Autoencoder

Autoencoders are a type of neural networks consisting of an encoder and decoder. In general, the encoder compresses the input into a lower dimensional vector and the

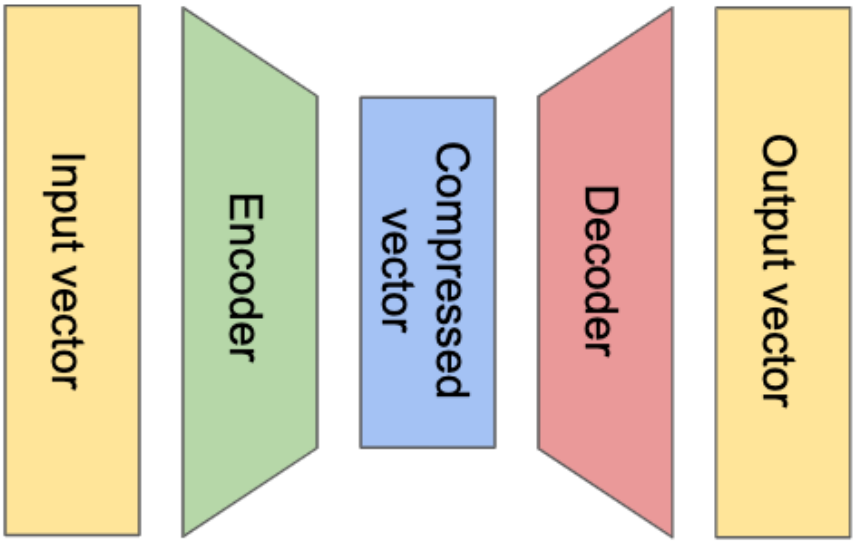


Figure 5.4: Illustration of a basic Autoencoder

decoder recreates the original input from it. The unsupervised training objective of the network is to reconstruct the original input with as little loss as possible. Because of the created "bottleneck", a data compression method is learnt by the neural network.

Once trained, the autoencoder model can be used to transform data into their compressed representations by only using the encoder part. Autoencoders are widely used for denoising, anomaly detection, feature extraction and more.

In this work, we first train an autoencoder with the normalized $\vec{s}_{raw}$ of all the users in our training set to train the model. We then encode each vector into a compressed representation $\vec{s}_{ae}$ with the trained encoder part of our network. We then use the resulting $\vec{s}_{ae}$ as feature vectors for attacks. We create two variants of autoencoders: one consisting of densely connected layers and the other one consisting of 1 dimensional Convolutional Neural Network (1D CNN) layers.

5.4.4 Normalization

We now describe the normalization techniques we use on the raw data as well as the features generated by the methods above.

5.4.4.1 Feature-wise normalization

In this method, each i-th value of a feature vector is divided by the maximum of i-th values of all feature vectors. This normalization is mostly useful for statistics and distribution features.

5.4.4.2 Vector-wise normalization

In this method, each element of an input vector is divided by the value of the largest element of this vector. For raw step count data, such processing removes information of

how fast a user can walk, but keeps information when does she walk more and when less.

5.4.4.3 Probability distribution

In this method, each element of a vector is divided by the sum of all of its elements. Such processing preserves information about the overall relative walking speed of a person, but loses information about the total amount of steps made during the concerned time period.

5.5 Attribute inference attack

It has been shown that social network user's personal attributes, e.g., gender, age, political believes, location, occupation, even if not disclosed, can be inferred from their friends [6], online behaviours [35] or from the content of their messages [21]. In this work we study whether gender, age and education can be inferred from user's step count data itself.

Using a wide range of feature vectors extracted from raw step data, we train multiple different machine learning and deep neural network models on the attribute classification task and report the results in Section 5.6.

5.5.1 Experimental Setup

Out of 1000 users in our dataset we need to filter out 3 of them due to step measurement errors. We focus on binary classification of 3 attributes: gender, age and education. For gender, we classify between male and female, where 56% of participants are female. For education, we focus on middle and high level education as explained before. Age of the participants range from 30 to 91 years old, we choose the median (55 years) and classify participants as younger than 55, or 55 and older, giving us 50% of users in each group.

We assume an adversary that has access to some amount of user's data with their attributes for training her models and unlabeled step count data of users whose attributes she wants to infer. To simulate this settings, for each attribute, we randomly choose 80% of users of each class as training data set and the remaining 20% as testing data set.

We run an extensive amounts of attacks for attribute inference. We first use a wide range of feature extraction techniques on the raw dataset, and then train and test many different classifiers on those feature vectors.

5.5.2 Feature Extraction

We will show our methods for feature extraction in a step by step manner.

1. The raw step vector of each user $\vec{s}_{raw}$ we take as a whole week $\vec{s}^{D}_{raw}$ or split it into different days, resulting in 7 $\vec{s}^{d}_{raw}$ for each user. This results in 2 feature vector types.

2. We run statistical methods (as described in 5.4.1) for all possible subsets of $\{max, mean, median, std, sum\}$ (other then $\varnothing$) for window sizes $w \in \{12, 24, 48, 60, 120, 240, 480, 720, 960, 1440, 1920, 2880, 5760\}$ on $\vec{s}^{d}_{raw}$ and for window sizes $w \in \{240, 480, 720, 960, 1440, 1920, 2880, 5760, 40320\}$ on $\vec{s}^{D}_{raw}$. This results in $(2^5 - 1) * (13 + 9) = 682$ feature vector types.

3. We run distributions as described in 5.4.2 on $\vec{s}^{D}_{raw}$ and $\vec{s}^{d}_{raw}$ for window sizes $w \in \{240, 720, 1440, 2880\}$ and for bucket sizes $b \in \{2, 4, 8\}$. This results in $2 * 4 * 3 = 24$ feature vector types.

4. We apply three different normalization methods as described in 5.4.4 to all feature vector types we made so far. This results in $(2 + 682 + 24) * 3 = 2124$ normalized feature vector types.

5. On each normalized feature vector type we train a densely connected auto encoder. Because some of them are much shorter then others we need to adjust the layer sizes to avoid compressing a shorter vector into a single value. The densely connected auto encoder consists of five densely connected layers of sizes $(l_1, l_2, l_3, l_4, l_5)$, where l_1 is equal to the length of the input feature vector,

$$l_2 = \begin{cases} min(2048, \lfloor l_1/4 \rfloor), & \text{if } l_1 > 255 \\ \lfloor l_1/2 \rfloor, & \text{otherwise,} \end{cases}$$

$$l_3 = \begin{cases} \lfloor l_2/4 \rfloor, & \text{if } l_2 > 127 \\ \lfloor l_2/2 \rfloor, & \text{otherwise,} \end{cases}$$

   $l_4 = l_2$, $l_5 = l_1$. We then take third layer and use it as a new feature vector, which results in 2124 feature vector types.

6. On each normalized feature vector type we train a convolution auto encoder. It consists of two convolution layers with 8 filters each, k_1 and k_2 kernel sizes respectively and a max pooling layer in between with the pool size of p. Then two transposed convolution layers with an unpooling layer between them with corresponding parameters follow. Because of how long it takes to train a cnn autoencoder, we use the following sets of parameters $(k_1, k_2) \in \{(6, 6), (9, 6), (9, 9), (21, 9)\}$ and $p \in \{2, 4\}$ only on weekly feature vectors $\vec{s}^{D}_{raw}$ normalized with feature-wise normalization and $k_1 = 21, k_2 = 9, p = 4$ for the rest. This results in $4 * 2 * 1 + 2121 = 2129$ feature vector types.

7. Additionally, we use a third type of data splitting from step 1, that we call actions. We take a whole week of step data and extract all periods of maximal length that have no subperiod of 8 consecutive periods with 0 steps done (2min of rest) and use each such action together with the length, and time of beginning of the action as a feature vector. We then apply statistics (all statistics together) from step 2 and distributions from step 3 with the window size equal to action length to obtain 3 (together with raw) feature vector types. The reasoning behind it is that maybe a combination of shorter activities are better suited to predict person's attributes.

In total we are testing 7085 feature vector types of either seven or one vector per user and 3 actions feature vector types.

5.5.3 Classifiers

We classify on three different binary tasks: gender (male or female), education (medium or high) and age (below 55 or above). We use the following classifiers from sklearn on each task with each feature vector type: Random Forest, Linear Regression, Support Vector Machine, Linear Support Vector Machine and 3 Densely Connected Neural Networks, all consisting of a densely connected and dropout layers. If l is the feature vector length and `d` is a 20% dropout layer, then the networks will look as follows: $(l, \frac{1}{4}l, \texttt{d}, 1), (l, \frac{1}{2}l, \texttt{d}, \frac{1}{8}l, 1), (l, \frac{1}{2}l, \texttt{d}, \frac{1}{4}l, \texttt{d}, \frac{1}{16}l, 1)$.

Now each feature vector set is split into 80% training and 20% testing data in a balanced way (e.g., if the classification is to be performed for gender, the 80 - 20 split is done on male and female separately and only then combined and shuffled together for both training and testing). For the feature vector sets, where we have 7 vectors per user, we make sure, that all the vectors are in either training or testing set for each user.

In addition for actions feature vectors an additional aggregation step is needed, because our objective is to classify an attribute of a person and not the action itself. After obtaining scores from the classifier for each action of a specific person we discard 50% of the results, that are the least sure about the attribute and then calculate both arithmetic mean and majority vote.

Due to long training time, we run CNN classifiers only on selected feature sets, namely raw daily and weekly step count with both feature-wise vector-wise normalization, all 3 actions, 4 distributions and 4 statistics (2 best performing on average and 2 with highest performance, as measured on other classifiers). Each CNN classifier looks as follows: 2 convolution layers with 16 filters each, k_1 and k_2 kernel sizes respectively, 50% dropout layer, max pooling layer with the pool size of p, fully connected layer with 100 neurons and a fully connected layer with a single perceptron. We use $(k_1, k_2) \in ((21, 9), (6, 6))$ and $p \in (2, 4, 8)$

We run 3 LSTM classifier types, regular LSTM, bidirectional LSTM and bidirectional LSTM with attention. We use a Luong-style dot-product attention layer, [68]. We only run LSTMs on feature vectors resulting from fixed window size aggregations, i.e. statistical (with window size $w \in \{60, 240, 720\}$), distributional (with window size $w \in \{60, 240, 720\}$, bucket size $b \in \{2, 4\}$, statistical activities (with all the statistics combined) and distributional activities (with bucket size $b = 2$). We split the feature vectors according to actions or windows and feed each subvector one by one to the LSTM. With an analogy to training an LSTM on a sentence of words, where each word is encoded in a fixed size vector, each window or activity is a separate word and the result of distribution or statistics function is the embedding. All three LSTMs consist of one respecive LSTM layer with 16 or 32 units, one dropout layer with 0.2 dropout rate and a fully connected layer with a single perceptron.

5.6 Attribute inference evaluation

Our goal was not to train the best possible classifier for the attribute inference task, but rather to cover a wide range of different classifiers and feature vectors. We use AUC (Area Under ROC Curve) for measuring the performance of our classifiers. As a result, we have 149433 classification results in total: 7 non-CNN classifiers for 3 different attribute types run on 7085 feature vector types, 3 actions feature vector sets with either mean or majority vote, 6 CNN classifiers for 3 different attribute types run on 12 selected feature vector types and 3 actions feature vector sets with either mean or majority vote and 6 LSTM classifiers 3 different attribute types run on 9 selected feature vector types and 2 action feature vector sets.

5.6.1 Normalization Method

The normalization methods do not have a strong influence on the results. On average the best performing is feature-wise normalization, with the exception of random forest classifier, where normalization is not needed and it would, in almost all cases, cause a small drop in performance.

5.6.2 Statistical Method

We first look at the best window size for the statistical methods. On Fig. 5.5 we can see best results for single statistics on different windows on the task of predicting age. Additionally we also plot highest and average result of all the (combined) statistics. We can see that the best results are obtained for $w = 720$ and $w = 1920$, which are 3 and 8 hours periods correspondingly. The best performing single statistic is maximal number of steps and the worst is median. Sum and mean of the steps are overlapping on the plot, since mean is just sum divided by a constant. Especially interesting is the 8 hours period since it reflects differences in people's habits during the night (from midnight to 8 a.m.), working period (from 8 a.m. to 4 p.m.) and leisure time (4 p.m. to midnight). With only the maximal number of steps done in each period we can already infer person's age with the AUC score of 0.74 with a linear regression classifier.

On Fig. 5.6 we analyze best performing subsets of statistics on the window size $w = 720$. We can see that the easiest classification task is to predict user's age, then education and the hardest one is gender. Interestingly the best performing for both gender and age prediction is a combination of maximal and median values - the best and worst performing when taken alone. In general, combinations of more then two statistic values do not perform well. When a classifier has too many features and not enough data to train on, its performance will usually be lower then with less but well chosen features.

For all attributes the best results are found on week long step count series. The best performing for age was a random forest classifier trained on maximal and median values on windows of size $w = 720$ (3 hours), achieving 0.78 AUC. The best result for education is achieved by the smallest densely connected neural network classifier (with only 1 hidden layer) trained on maximal value of windows of size $w = 240$ (1 hour) with feature-wise normalization, achieving 0.68 AUC. The best performance for gender was

a random forest classifier trained on maximal values of windows of size $w = 1440$ (6 hours), achieving 0.65 AUC. We can see that for age and gender the best discriminator is how fast and how fast on average are people walking, while for education it is more important how much are people walking. We are confident with prediction regarding age, but much less so when it comes to education or gender.

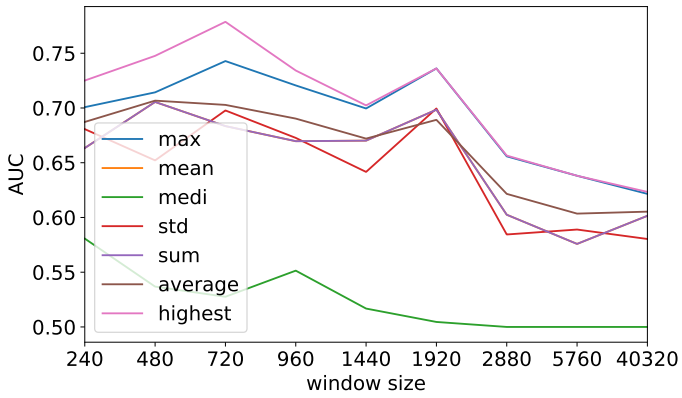


Figure 5.5: Influence of window size on age prediction

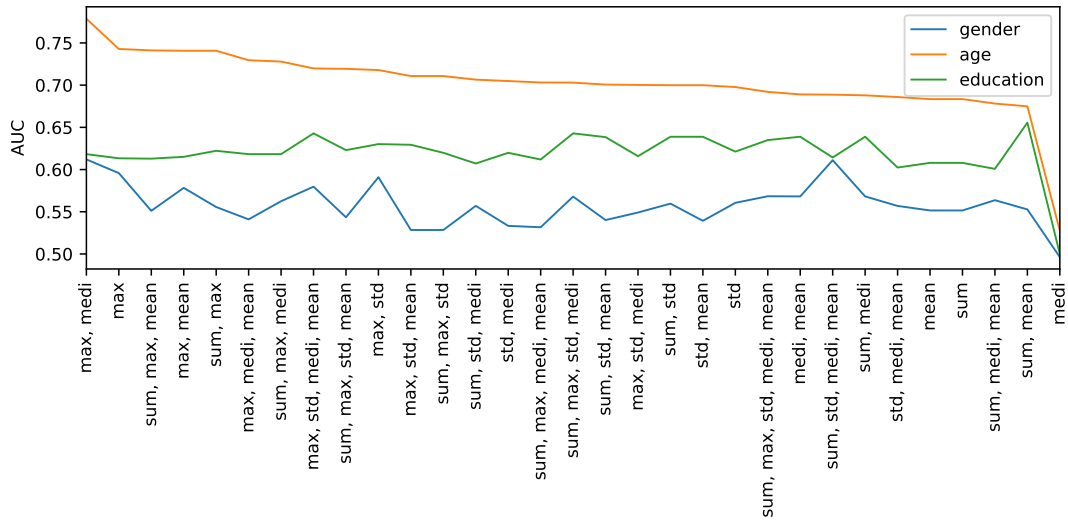


Figure 5.6: Influence of statistics subset on age predictions

5.6.3 Distributional Method

On Fig. 5.7 we can see best results for all bucket sizes, window sizes and target attributes. We can observe, that bucket size $b = 2$ has always the best (or almost best) performance, meaning more fine grained distribution has a positive impact on the results (even though it generates more features that classifier will need to learn how to process). The best

result for each attribute are as follows:

- **Age.** Random forest classifier trained on $b = 2$ and $w = 240$ (1 hour) achieves 0.73 AUC,
- **Gender.** Support vector classifier trained on $b = 2$ and $w = 1440$ (6 hours) achieves 0.67 AUC,
- **Education.** Densely connected neural network (with 2 hidden layers) trained on $b = 8$ and $w = 720$ (3 hours) achieves 0.66 AUC.

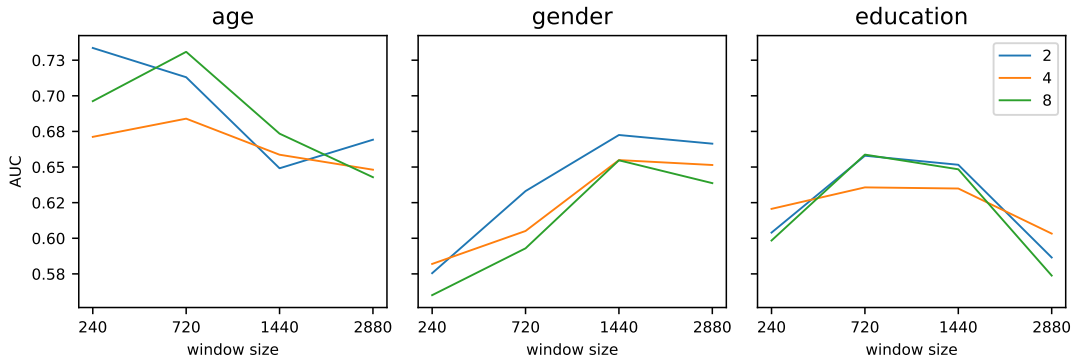


Figure 5.7: Influence of the window size for distributional method on the prediction quality

5.6.4 Autoencoders

We observe that autoencoders almost always cause a drop in AUC as compared to the feature vectors on which they were trained. The best autoencoder was a densly connected auto encoder on weekly max and mean statistics on windows of size 960 with feature-wise normalization, age classificiation logistic regression achieved 0.77 AUC value, while the same classifier with the same feature vectors but without auto encoder step achieved 0.73 AUC. The improvement, although non-negligable, is an exception rather than a trend. It might be the case, that we do not have enough data for the autoencoders to efficiently learn an intermediate representations, and training directly for the task of prediction show better results. It might, however, still be a useful technique if we have access to big number of unlabeled step count timeseries and we only know attributes of a small percentage of them.

5.6.5 Actions

On average arithmetical mean outperforms majority voting of the result scores, by a small margin. However in some of the best performing cases, mean can have a drastic improvement (e.g., for plain steps and a CNN gender classifier, after cross validation, majority vote achieves 0.53 AUC, while mean gives 0.78). The best results were obtained for gender prediction with raw actions and a random forest classifier resulting in AUC of

0.87. For age the best was a densely connected neural network classifier (with 2 hidden layers) trained on distribution of actions and achieved AUC of 0.69. For education the best was logistic regression trained on distributions of actions achieving AUC of 0.61. While with all other methods, gender was difficult to predict, actions can do it with a very high confidence. It suggests that there exists certain walking patterns characteristic for either man or woman.

5.6.6 LSTMs

Because LSTMs use feature vectors as a 2d matrix rather then a long vector we do not include them in the previous evaluations of the feature selection methods, but instead dedicate them this section. For gender and education, distributional method outperforms statistical ones, while for age the statistical gives best results. Splitting the data based on activities rather then a fixed window size causes a big drop in the AUC score for all the attributes. For both gender and age, window size $w = 240$ gives the best result. For distributional methods, the bucket size does not seem to have a big influence on the AUC score, but best results are achieved for $b = 2$. Making the LSTM bidirectional or adding attention both improve the results for gender prediction, but do not achieve higher performance on the other two attributes. It is possible that due to larger amount of trainable weights, more data would be needed to make use these additions to the basic LSTM.

5.6.7 Summary

Because of the amount of classifiers trained, we were not able to cross validate each experiment and rather we take the best performing ones (described in details in previous subsections) and run them again with 5 fold cross validation. This approach results in us reporting potentially lower best performance values then possible, since we might have missed some well performing methods due to unlucky split into training and testing data, while ensuring, that we do not report high, lucky performance scores.

In general, weekly walking patterns are much better then daily for the attribute prediction task. Daily patterns can differ drastically between weekdays and weekends and thus, looking at a person's whole week is advised.

The result for age prediction are in Fig. 5.9. It is the easiest attribute to predict and many classifiers achieve AUC greater then 0.7. Great performance of statistical feature vector generation confirms our observations, that for age inference the most important is how fast (in steps per minute) a person walks, as older people tend to walk slower. Data split into activities instead of fixed size windows does not improve the predictions, which means that no activities were found that would clearly indicate person's age. LSTMs perform very well on both statistical and distributional methods, however the best result is achieved by a CNN consisting of 2 convolution layers with 6 filters of size 16 each, a 0.5 dropout layer, max pooling layer of pool size 8, dense layer with 100 perceptrons and finally dense layer with 1 perceptron, run on a single maximum statistic on window of size 240 (maximum number of steps done every hour). The average AUC over 5 fold validated training-test split is 0.778 with standard deviation of 0.047. In Fig. 5.8 we show the Receiver Operating Characteristic curve for the best cross validation run.

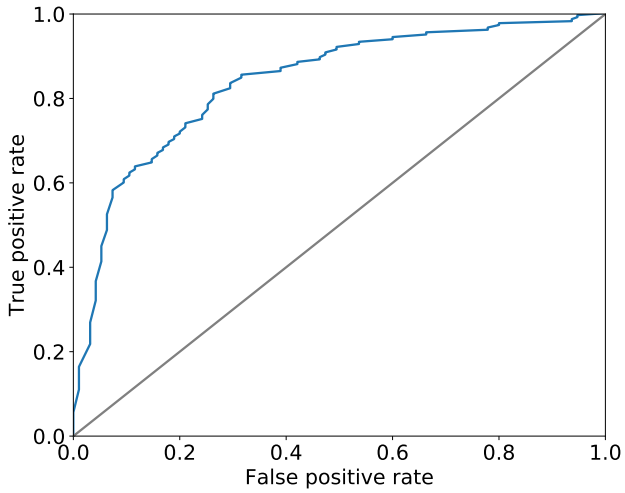


Figure 5.8: ROC curve for best performing age classifier (in blue). The gray line indicates the baseline (random guess).

The results for gender prediction are in Fig. 5.10. In case of gender prediction, simple statistics are not enough and distributional method seems superior, as it contains more information. Again LSTMs and CNN outperform simpler machine learning methods. On most classifiers we can observe that activities perform better then fixed window size split. This indicates existence of certain walking patterns characteristic for males or females. We do not have data saying what were users doing during those activities, but it would be very interesting to study, which activities expose our gender. The best result is achieved by a CNN consisting of 2 convolution layers with 6 filters of size 16 each, a 0.5 dropout layer, max pooling layer of pool size 4, dense layer with 100 perceptrons and finally dense layer with 1 perceptron run on plain activities with the prediction for a specific user being an arithmetical mean of predictions of all their activities. The average AUC over 5 fold validated training-test split is 0.780 with standard deviation of 0.024.

The results for education prediction are in Fig. 5.11. Education is by far the hardest attribute to predict and we are not able to train a reliable classifier. LSTMs seem to perform slightly better then other methods, with the best being a standard LSTM with 16 units and a 0.2 dropout layer, trained on distributions with $b = 4$ and $w = 720$ (3h). The average AUC over 5 fold validated training-test split is 0.650 with standard deviation of 0.037. The reason why we are able to perform better then a random guess is probably because of the correlation of age and education in our data. To verify this claim we use the best performing (for age) CNN classifier, train it on statistical file for the task of predicting age and use it to predict education. In such setting we managed to achieve AUC of 0.637, which confirms our hypothesis.

5.7 Linkability

Our linkability attack tries to identify whether two observations of stepcount data belong to the same individual. In our experiments we focus on daily observations,

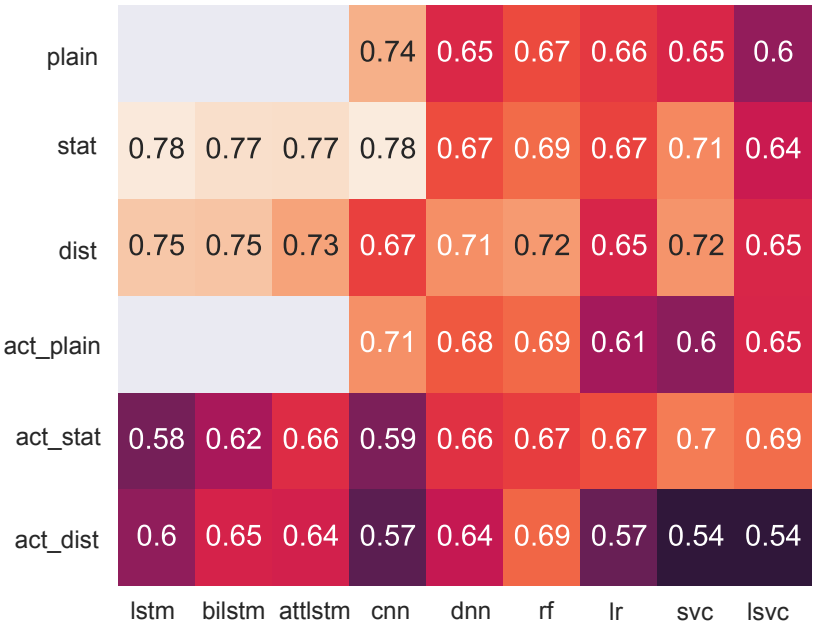


Figure 5.9: Best AUC scores for age prediction

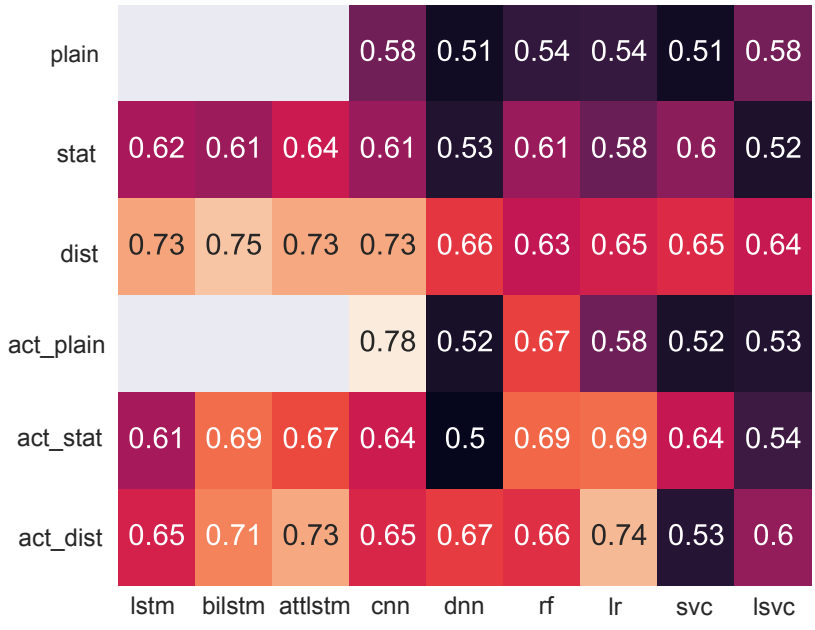


Figure 5.10: Best AUC scores for gender prediction

meaning for each user u we have 7 daily feature vectors $\vec{s}_u^d$, where $d \in D$ and $D = \{Monday, Tuesday, \ldots, Sunday\}$.

Formally, let $\vec{s}_u^{d_1}$ and $\vec{s}_v^{d_2}$, $d_1 \neq d_2$ be the step count feature vectors from two days of users u and v. The adversary's objective is to predict the probability that u and v are the same user, where one or both are anonymous.

$$f^{link}(\vec{s}_u^{d_1}, \vec{s}_v^{d_2}) = (u \stackrel{?}{=} v)$$

$d_1, d_2 \in D, d_1 \neq d_2$. The adversary can use the daily raw step counts per 15s interval $\vec{s}_{raw,u}^d$, features created by the statistical and distributional methods or their normalized versions as described in sections 5.4.1, 5.4.2 and 5.4.4.

The attack function f^{link} can be instantiated in three different ways as follows:

5.7.1 Similarity based Attack

This attack is unsupervised and is the simplest one. It involves calculating a distance metric between two samples $\vec{s}_u^{d_1}$ and $\vec{s}_v^{d_2}$ and comparing it to a threshold t. If the distance is smaller than t, then we predict that the samples came from the same user. We use the euclidean distance and the cosine distance. Other distance metrics have similar or lower performance and are therefore not included. For the euclidean distance, the attack function is instantiated as:

$$f_{eucl}^{link}(\vec{s}_u^{d_1}, \vec{s}_v^{d_1}) = \sqrt{\sum_{i=1}^{n}(s_{u,i}^{d_1} - s_{v,i}^{d_2})^2} < t_{eucl}$$

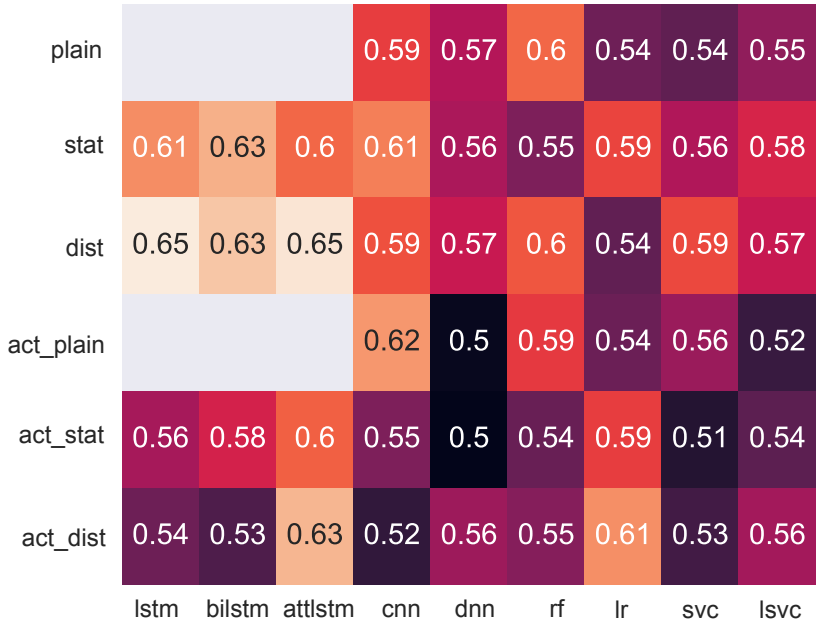


Figure 5.11: Best AUC scores for education prediction

Similarly, for the cosine distance, the attack function is instantiated as:

$$f_{cos}^{link}(\vec{s}_u^{d_1}, \vec{s}_v^{d_2}) = 1 - \frac{\vec{s}_u^{d_1}\,\vec{s}_v^{d_2\,T}}{||\vec{s}_u||||\vec{s}_v||} < t_{cos}$$

where $\cdot^T$ denotes vector transposition and $||\cdot||$ denotes the L_2 norm of the vector. t_{cos} and t_{eucl} denotes thresholds for each attack.

Experimenting with different values of thresholds t_{cos} and t_{eucl} gives us a range of false and true positive values. Plotting the false-positive rate on the x-axis and the true-positive rate on the y-axis produces the ROC (Receiver Operating Characteristic) curve. We use the area under this curve, referred to as AUC, to analyze the success of the attack. Unlike other metrics, AUC summarizes the performance in a straightforward manner: 0.5 is as bad as random guessing and 1.0 indicates perfect prediction. We assume the attacker knows the best threshold $t_{\cos}$ or t_{eucl} and therefore get an upper bound for the privacy threat resulting from the unsupervised attack. We denote these attacks by `Euclidean` and `Cosine`.

5.7.2 Random Forest Classifier based Attack

This attack uses random forest classifier, which is a state-of-the-art supervised machine learning approach. We first apply a vector distance metric, namely, L1 distance between the pair of samples. We then use the resulting distance vectors as features to fit random forest classifiers. As above, we use the AUC between the positive class probability of the random forest and the true class labels to evaluate the performance of the attack. L2, element-wise arithmetic mean and Hadamard distance have similar or lower performance and are therefore not included. We denote this attack by `RF_standard`.

5.7.3 Siamese Neural Network based Attack

One Shot Learning has become the state of the art solution when large amounts of labelled training data is not available for standard classification with deep neural networks. This attack uses One Shot Learning with Siamese Neural Networks [14]. Siamese networks are a special type of neural network architecture that contain two identical sub-networks that have the same configuration, the same parameters and weights, and mirrored updates during training. Instead of learning to classify its inputs, this model learns to differentiate between two inputs and finds a similarity or relationship between them. Specifically, our Siamese Network Model leverages the semantic similarities between pairs of step count vectors of the same user and the differences between pairs of step count vectors of different users. Just one sample of the true identity of a user in training may be sufficient to predict or recognize this user in the future.

Figure 5.12 shows an illustration of a basic Siamese Neural Network.

We use L1 distance to combine the output of the shared networks, followed by a fully connected layer with sigmoid activation function. The hypothesis is that two inputs from different users will produce different feature embeddings at the inner layers, and inputs from the same user will result in similar feature embeddings. Hence the

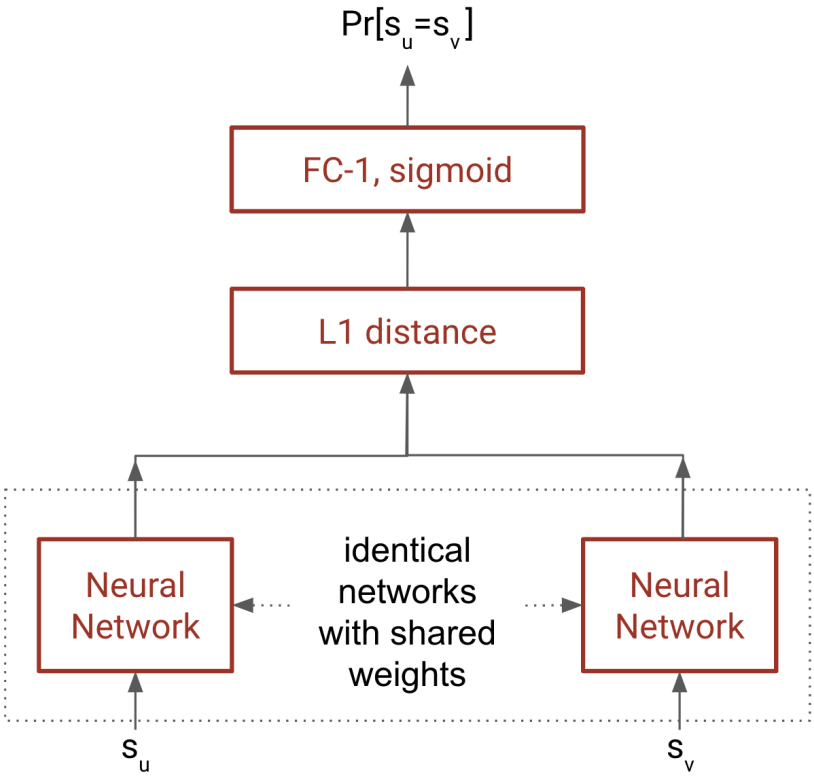


Figure 5.12: Illustration of a basic Siamese Network

element-wise absolute difference between the two feature embeddings must also be very different between the two cases. Therefore the score generated by the output sigmoid layer must also be different in both cases.

This model is similar to having an auto encoder and a distinguishing classifier, but instead we learn encodings useful for this exact purpose of our task. We found that using auto encoders for this task does not produce representations resulting in good predictions and thus we exclude those experiments in the paper.

We instantiate the two shared subnetworks with different state of the art neural network layers to get five different attack variants as follows:

5.7.3.1 Dense layers

In the first variant, we use two densely connected layers for the shared subnetworks. The first dense layer contains half the number of neurons as the size of the input. The second dense layer further contains half the number of neurons as the first one. We denote this attack by `Dense_siamese`.

5.7.3.2 LSTM layers

In this variant, the shared subnetworks consist of an LSTM layer with 8 units, and dropout of 0.2. We denote this attack by `LSTM_siamese`.

5.7.3.3 Bidirectional LSTM layers

In this variant, the shared subnetworks consist of a Bidirectional LSTM layer with 8 units, and dropout of 0.2. We denote this attack by `biLSTM_siamese`.

5.7.3.4 Attention layers

In this variant, the shared subnetworks consist of a Luong-style dot-product attention layer, [68] applied to the output of the Bidirectional LSTM layer in `biLSTM_siamese`. We denote this attack by `Attention_siamese`.

5.7.3.5 1D CNN layers

In this variant, the shared subnetworks consist of two 1D CNN layers, followed by a max pooling layer of size 8. Each CNN layer had a filter size of 16 and a kernel size of 6. The max pooling layer is followed by a flatten layer, which is followed by a densely connected layer of 100 neurons. The result of using a pooling layer and creating down sampled or pooled feature maps is a summarized version of the features detected in the input. They are useful as small changes in the location of the feature in the input detected by the convolutional layer will result in a pooled feature map with the feature in the same location. We denote this attack by `CNN_siamese`.

As before, we calculate the AUC between the true class labels and the result of the sigmoid function to evaluate the success of the attack.

5.8 Linkability - Evaluation

We now present the experimental evaluation of our linkability attacks, namely our unsupervised attacks `Euclidean` and `Cosine`, our random forest attack `RF_standard` and our three Siamese network attacks `CNN_siamese`, `LSTM_siamese` and `Dense_siamese` on our dataset.

5.8.1 Experimental Setup

We evaluate our linkability attacks on various stepcount features $\vec{s}_u^d$ which include raw step counts $\vec{s}_{raw,u}^d$ as well as $\vec{s}_{stat,u}^d$, $stat \in \{sum, max, median, mean\}$ and distributions $\vec{s}_{dist,u}^d$, over a period of 24 hours on a day $d \in D$. We also evaluate the attacks on the normalized (feature-wise normalization) versions of each feature as described in 5.4.4.1 and denote the results by the suffix `_norm`. For example we denote the results of the attack `Euclidean` on the normalized features by `Euclidean_norm`. We remove features with variance less than 0.001 before fitting into each model.

We create all possible pairs of daily step counts $\vec{s}_u^d$ out of 7 days of data for each user to obtain positive class samples (vectors representing two different days of the

same user). Thus for each user u we have $\binom{7}{2}$, i.e. 21 positive samples in the form $\{\vec{s}_u^{d_1}, \vec{s}_u^{d_2}\}$, $d_1, d_2 \in D$, $d_1 \neq d_2$. For negative samples, we randomly pick an equal number of pairs of daily step counts from different users in the form $\{\vec{s}_u^{d_1}, \vec{s}_v^{d_2}\}$, $d_1, d_2 \in D$, $u \neq v$.

We perform a 5-fold cross validation to evaluate our attacks. The positive and negative samples are equally divided into both sets. We do not optimize any parameters for the supervised attacks. The results of our supervised attacks are therefore a lower bound on the privacy risk arising out of such an adversary model.

Following the above strategies and removing symmetric user pairs, we get 20937 samples in each class and 41874 samples in total. We train on 33500 samples, test on 8374 samples in each iteration.

5.8.2 Results

Figure 5.13 shows the results of our attacks on the features $\vec{s}_{dist}$. The y-axis indicates the AUC (mean and the standard deviation via an error bar over all cross validation folds). The x-axis indicates different bucket size b and window size w combinations for the distributions in the input file. We notice that the unsupervised attacks `Euclidean` and `Cosine` and the standard random forest classifier `RF_standard` have very low performance. However our `Dense_siamese` and `CNN_siamese` classifiers hugely outperform them and achieves AUC higher than 0.75 for most of the inputs. The RNN based classifiers have only a slightly lower performance; the `LSTM_siamese` is outperformed by the `biLSTM_siamese`, which is further outperformed by the `Attention_siamese` as expected. For big bucket and window sizes, we have lower number of features. Therefore the `Dense_siamese` attack does not have a significant advantage over the simpler attacks. We notice that its performance is almost the same as the unsupervised attacks for $b\_w = 8\_2880$.

Figures 5.14-5.15 shows the results of our attacks on the features $\vec{s}_{max}$, $\vec{s}_{mean}$, $\vec{s}_{medi}$ and $\vec{s}_{sum}$. The x-axis shows increasing window size w over which the statistic is calculated. We observe that our neural network based siamese attacks outperform the `RF_standard`, `Euclidean` and `Cosine` for smaller window sizes. However as window size increases, the performance of the siamese attacks drops quite fast. We also observe that `CNN_siamese` and `biLSTM_siamese` always achieved the top AUCs for each of the 5 feature extraction methods, closely followed by `LSTM_siamese`. The `Attention_siamese` struggles for small window sizes (larger number of timesteps), however as the number of timesteps increase, it outperforms `biLSTM_siamese` and `LSTM_siamese`. This shows that the success of attention mechanism is limited to sequences of smaller length.

We further compare the five different feature extraction methods with each other to see which captured the most information for user linkability. Figure 5.18 shows the AUCs (mean and standard deviation over all cross validation folds) achieved by the three best performing inputs of each feature extraction method. We see that features extracted by the distribution method $\vec{s}_{dist}$ outperform the rest and as expected $\vec{s}_{medi}$, i.e. the median statistic shows the poorest performances. The highest average AUC, achieved by $\vec{s}_{dist}$ with $b\_w = 4\_720$, is 0.78 using `Dense_siamese`.

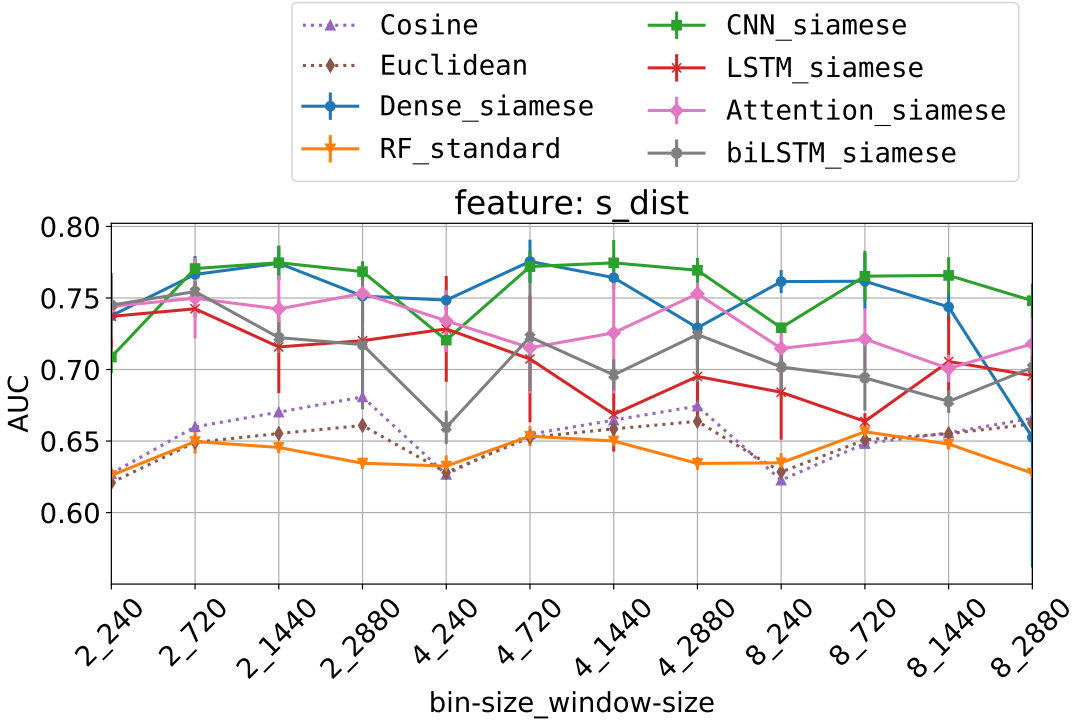


Figure 5.13: Performance of all attacks on distribution feature $\vec{s}_{dist}$. The x-axis indicates the bucket size bfollowed by the window size w.

5.9 Conclusion

We perform the first systematic analysis of privacy risks arising from physical activity data. Via an extensive evaluation on a real life dataset, we demonstrate that indeed step count data poses significant threat to various aspects of users' privacy.

With a relatively small dataset of 1000 users only and using simple machine learning classifiers, we find that an adversary having access to actimetry (step count) data of different users can perform a linkability attack with a high confidence. Attribute inference is harder, but still possible. Among all three attributes, prediction of age is the easiest, while education cannot be predicted reliably. For gender inference specifically, we obtain promising results by averaging predictions from classifying shorter walking periods (activities). Some of these short patterns strongly indicate users' gender but not education or age. Exploring such smaller patterns for different attributes would be an interesting direction for future work. Note that our goal is not to quantify the worst case privacy risk of a specific attack but to explore a wide range of different feature extraction methods and classifiers in a spectrum of privacy attacks.

Actimetry data is not currently regarded as privacy sensitive, but big players, having access to an even bigger dataset could deanonymize users and infer their hidden attributes much easier. This highlights the need for further research on privacy preserving ways of collecting such data.

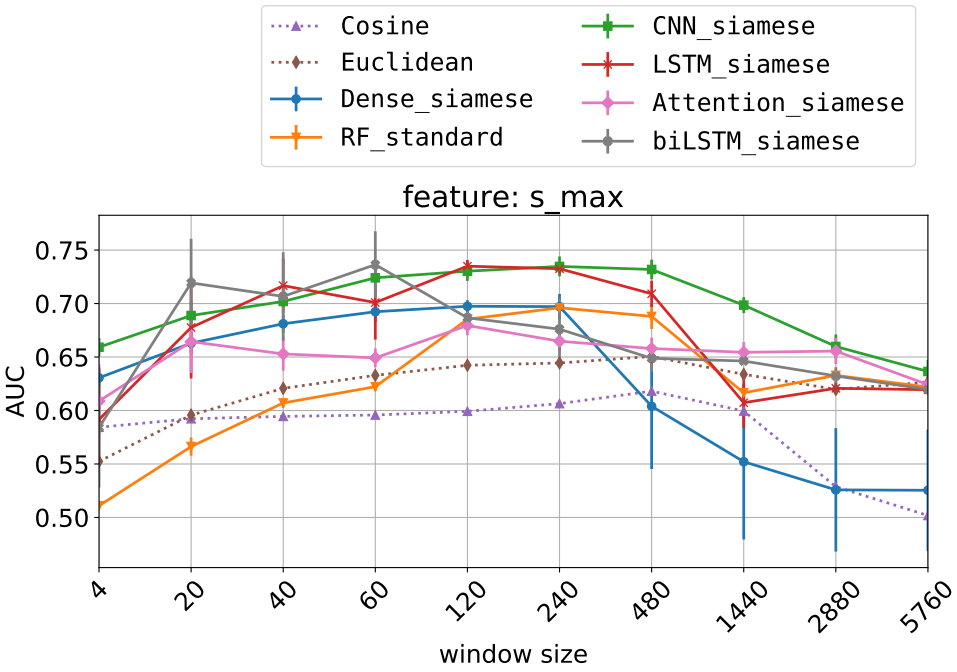


Figure 5.14: Performance of all attacks on the statistical feature $\vec{s}_{max}$

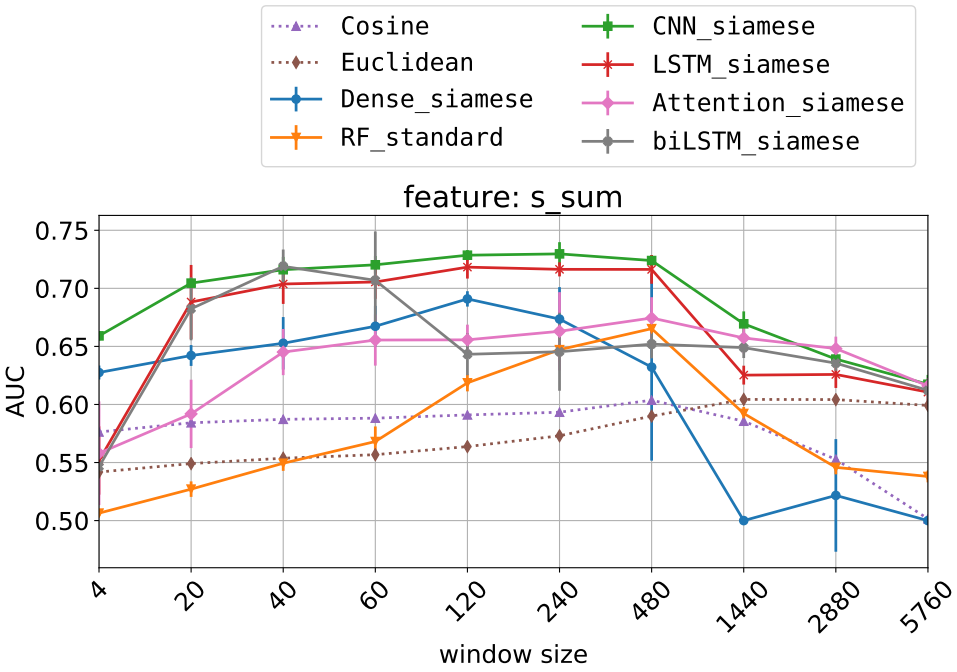


Figure 5.15: Performance of all attacks on statistical feature $\vec{s}_{sum}$

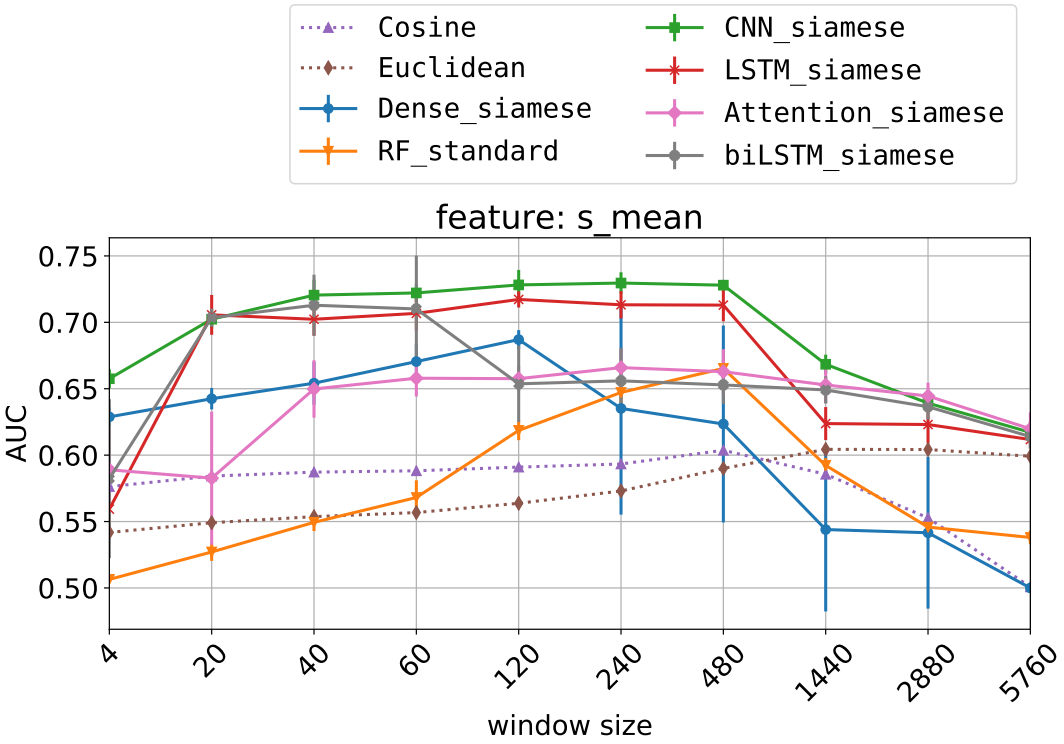


Figure 5.16: Performance of all attacks on statistical feature $\vec{s}_{mean}$

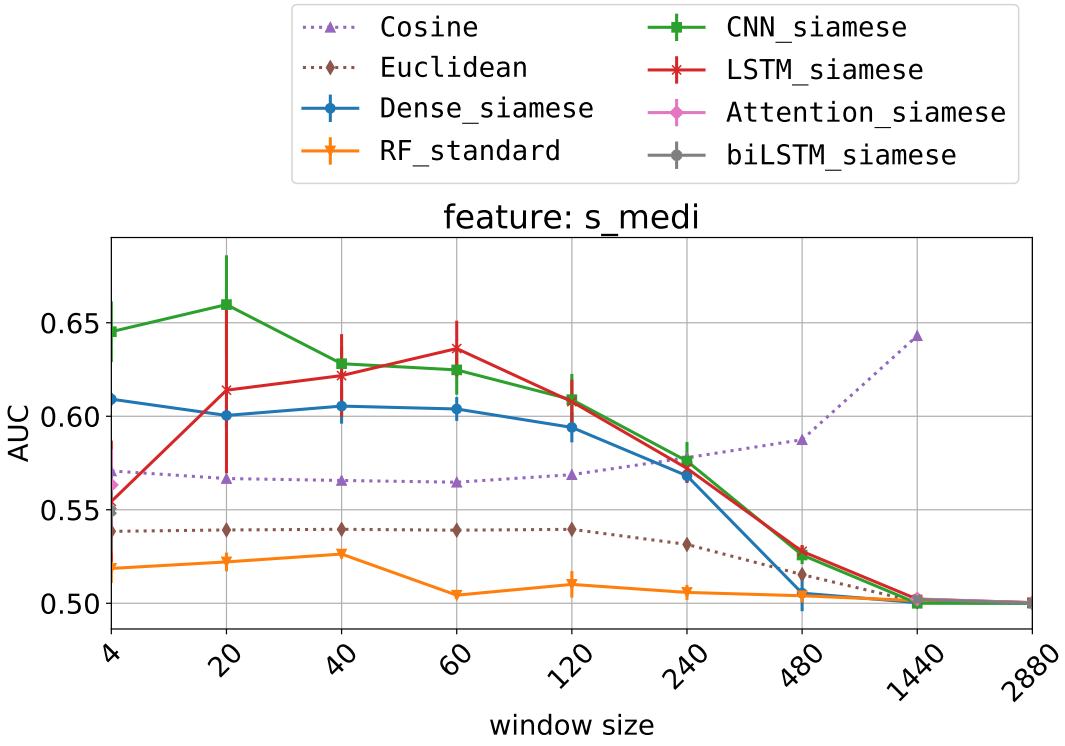


Figure 5.17: Performance of all attacks on statistical feature $\vec{s}_{medi}$

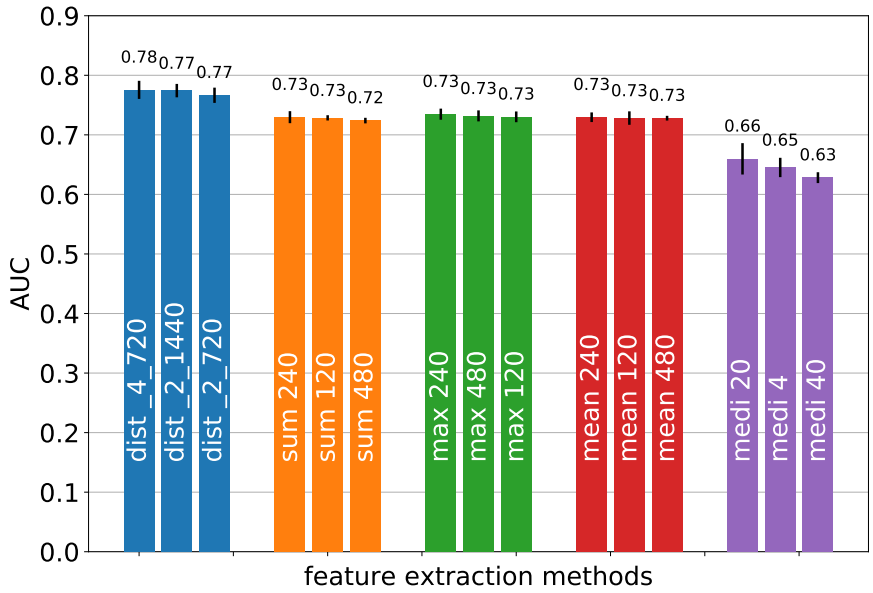


Figure 5.18: Top 3 AUCs achieved by input features from each feature extraction method (with the `Dense_siamese` classifier).

6
Fairwalk

Towards Fair Graph Embedding

6.1 Motivation

The rapid assimilation of online social networks (OSNs) into people's daily lives has resulted in a massive wealth of user generated data. Researchers have created various tools to mine this rich and diverse data. One of the most prominent tools in this domain is graph embedding, which maps each user into a lower dimension vector that reflects the user's structural information within the network, such as her neighborhood, communities she belongs to, her popularity, etc. Embedding vectors have been used in various tasks, such as friendship recommendation and personal attribute prediction.

Most network embedding methods rely on deep neural networks and are often treated as a black box. The resulting vectors may have captured undesired sensitive information, that reinforces the bias already existing in the network. When used for more advanced tasks, e.g., friendship recommendation, they may result in unanticipated fairness issues. Indeed, different groups based on ethnicity, gender, levels of urbanization, or wealth are usually unequally represented in the social network graph. Analyzing bias issues inherent in such embedding based algorithms is therefore especially important but has not received attention from academia so far.

In this paper, we take the first step towards quantifying and addressing fairness issues of graph embedding methods. In particular, we concentrate on one of the most prominent methods in this field, namely node2vec [38], and investigate its fairness with respect to friendship recommendation. As pointed in [105], a recommendation system biased towards a majority in an OSN can prevent minorities from rising in the network (becoming influencers with high reach). This is further confirmed by studies on strongly homophilic networks and their negative implications on minorities [58].

Anti-discrimination laws in various countries prohibit unfair treatment based on certain traits such as race, religion, gender (sensitive attributes). The Title VII of the Civil Rights Act of 1964 is one of the most noteworthy examples. However, these laws typically are too abstract for computation, hence there exist a variety of interpretation and the resulting proposals for mathematical formulations. In this paper, we focus on disparate impact also known as group fairness [9]. Disparate impact occurs when the decision of an algorithm benefits or hurts (a) certain sensitive feature group(s) more frequently than other group(s).

6.2 Contributions

First, we conduct the first-of-its-kind study of algorithmic fairness in the setting of graph embedding methods. We specifically analyze group fairness (disparate impact) of the state-of-the-art graph embedding: node2vec.

Second, we extend *statistical parity*, a well-known measure of disparate impact to measure fairness for groups based on sensitive attributes of pairs of users. We further propose a novel notion of *Equality of Representation* to measure fairness in friendship recommendation systems.

Third, we apply node2vec for friendship recommendation in real world OSN datasets as a case study. Using the fairness metrics above, we find biases in the recommendations caused by unfair graph embeddings.

Fourth, to mitigate the aforementioned biases, we propose a novel fairness-aware graph embedding algorithm *Fairwalk*, that extends node2vec. Evaluations on our OSN demonstrate the effectiveness of *Fairwalk* on both datasets w.r.t. all fairness metrics. Compared to node2vec, *Fairwalk* increases fairness in each case by a large margin. We also compare the utility of *Fairwalk* and node2vec.

6.3 Fairness Metrics

In this section, we present the notations used throughout the paper, followed by definitions of the fairness metrics. We focus on the notion of disparate impact. Firstly, we leverage the most well-known measure of disparate impact, namely *statistical parity*. Secondly, since our work is aimed at improving the representation of under-represented groups, we propose our own notion of *Equality of Representation* which comes in two variants - the user and the network level.

6.3.1 Notations

We define an OSN as an undirected, unweighted graph $\mathcal{G} = (U, E)$, where the set of vertices U represents the set of users and the set of edges $E \subseteq \{\{u, v\} \colon u \in U, v \in U, u \neq v\}$ represents the set of friendships. Note that since $\mathcal{G}$ is undirected $(u, v) \in E$ implies $(v, u) \in E$. We define neighbors of a node u as $\omega(u) = \{v \colon (u, v) \in E\}$.

We denote a sensitive attribute by $\mathcal{S}$, and gender as $\mathcal{S} = g$ and race as $\mathcal{S} = r$. We represent the set of all possible values of $\mathcal{S}$ by $\mathcal{Z}^{\mathcal{S}}$ and denote a specific value by $z^{\mathcal{S}} \in \mathcal{Z}^{\mathcal{S}}$. The function $\zeta^{\mathcal{S}} : U \to \mathcal{Z}^{\mathcal{S}}$ maps users to their attribute values. For example, for an Asian male u, $\zeta^{g}(u) =$ "`male`" and $\zeta^{r}(u) =$ "`asian`". We denote neighbors of u with an attribute value z as $\omega_z(u) = \{v \colon v \in \omega(u) \wedge \zeta(v) = z\}$. Note that $\bigcup_{z \in \mathcal{Z}} \omega_z(u) = \omega(u)$. We define a set of users recommended to user u as $\rho \colon U \to 2^U$ and set of users with a specific attribute value z recommended to u as $\rho_z(u) = \{v \colon v \in \rho(u) \wedge \zeta(v) = z\}$.

We partition user pairs $\{u, v\} \in U \times U$ into groups $G^{\mathcal{S}}_{ij}$ based on the attribute values of both u and v, specifically, for $i, j \in \mathcal{Z}^{\mathcal{S}}$, $G^{\mathcal{S}}_{ij} = \{\{u, v\} : \zeta(u) = i \wedge \zeta(v) = j \wedge u, v \in U\}$. We collectively refer to the set of all groups based on a sensitive attribute $\mathcal{S}$ as $\mathcal{G}^{\mathcal{S}}$.

6.3.2 Statistical Parity

Statistical Parity also known as *Demographic Parity* or *Independence* is the statistical equivalent of the legal doctrine of disparate impact [30]. *Statistical Parity* (typically defined in terms of two groups) requires the acceptance rates of the candidates from both groups to be equal. This allows us to measure recommendation fairness independent of the ground truth (existing friendships) which in our case is itself biased. Given a partitioning of user pairs based on an attribute $\mathcal{S}$, into two groups $G^{\mathcal{S}}_{ab}$ and $G^{\mathcal{S}}_{cd}$, let $P(G^{\mathcal{S}}_{ij})$ denote the acceptance rate for group $G^{\mathcal{S}}_{ij}$,

$$P(G^{\mathcal{S}}_{ij}) = \left|\{\{u, v\} : v \in \rho(u) \wedge \{u, v\} \in G^{\mathcal{S}}_{ij}\}\right| / \left|G^{\mathcal{S}}_{ij}\right|$$

The bias, or statistical parity, w.r.t. $G^{\mathcal{S}}_{ab}$ and $G^{\mathcal{S}}_{cd}$ is then the difference between $P(G^{\mathcal{S}}_{ab})$ and $P(G^{\mathcal{S}}_{cd})$.

We extend the above definition to account for multiple groups that we have for friendship recommendation. Since our groups consider the attribute values of both users in a pair, we have at least 4 groups when we consider a binary attribute like gender and in general $|\mathcal{Z}|^2$ groups when we consider any n-ary attribute like race. To capture the differences between multiple groups, we calculate the variance between the acceptance (recommendation) rates of each group in $\mathcal{G}^{\mathcal{S}}$.

$$\text{bias}^{\texttt{SI}}(\mathcal{G}^{\mathcal{S}}) = \text{Var}(\{P(G_{ij}^{\mathcal{S}})\} : G_{ij}^{\mathcal{S}} \in \mathcal{G}^{\mathcal{S}}) \tag{6.1}$$

6.3.3 Equality of Representation

Modern online recommendation systems suffer from problems of echo chambers or the information bubble effect [31, 97]. Usually users get recommendations based on their interests, which isolates them from other contradicting interests or viewpoints and reinforces the existing viewpoints. This also manifests in friendship recommendations in OSNs, where users who join a network into some well-established community, rarely see anything or anybody outside of it, although it might be interesting for them. As a direct consequence minority communities get isolated and are not seen by others. Our work aims to improve the representation of such under-represented groups in the OSN graph embeddings. To promote recommendations where all groups are equally represented, we define bias by *Equality of Representation* in two variants: $\text{bias}^{\texttt{ERg}}$ at the network level and $\text{bias}^{\texttt{ERu}}$ at the user level.

Network level. At the network level, we measure bias between different groups $G_{ij}^{\mathcal{S}}$, among all recommendations given in the network. Denoting the number of recommendations from a group $G_{ij}^{\mathcal{S}}$ as $N(G_{ij}^{\mathcal{S}}) = |\{\{u, v\} : v \in \rho(u) \wedge \{u, v\} \in G_{ij}^{\mathcal{S}}\}|$,

$$\text{bias}^{\texttt{ERg}}(\mathcal{G}^{\mathcal{S}}) = \text{Var}(\{N(G_{ij}^{\mathcal{S}})\} : G_{ij}^{\mathcal{S}} \in \mathcal{G}^{\mathcal{S}}) \tag{6.2}$$

User level. At the user level, among the recommendations $\rho(u)$ given to each user u, we measure the fraction of users having attribute value z and denote it as $z\text{-share}(u) = \frac{|\rho_z(u)|}{|\rho(u)|}$. Specifically, for a given attribute value z, bias is measured as the difference between a *fair fraction* (where each attribute value has an equal share) and the average z-share over all users.

$$\text{bias}^{\texttt{ERu}}(z) = \frac{1}{|\mathcal{Z}^{\mathcal{S}}|} - \frac{\sum_{u \in U} z\text{-share}(u)}{|U|} \tag{6.3}$$

This definition allows for measuring the bias for each sensitive attribute value independently.

6.4 Friendship Recommendation with node2vec

In this section, we first review the graph embedding algorithm node2vec, then describe a node2vec based recommendation system and finally analyze its fairness.

6.4.1 Graph Embedding - node2vec

A graph embedding is a mapping from nodes in a graph to a vector space $f : U \to \mathbb{R}^d$ where d is a hyperparameter capturing the number of dimensions of the vector space. Resulting vectors can be used for multiple machine learning tasks, e.g., link prediction for friend recommendation. Node2vec [38] first uses a random walk over a graph to generate walk traces and then extracts features based on learned traces.

Random walk. Given a graph $\mathcal{G}$, for each node $u \in U$, node2vec performs a single random walk `walk_num` number of times. The result is a list of traces, where each trace is a list of nodes' ids resulting from single random walks. Single random walk is a standard random walk over graphs without weights, i.e., at each step the next node is chosen uniformly at random among all neighbors of the current node. Both `walk_num` and `walk_len` are hyperparameters of node2vec.

Feature learning. Next, node2vec uses the generated traces to train a neural network and learn embedding vectors. Let us define a *network neighborhood* $N_s(u)$ as the set of nodes preceding and succeeding node u in the generated walk traces. The skip-gram architecture is adapted to the network traces with a goal to maximize the log-probability of observing a *network neighborhood* $N_s(u)$ based on the feature vector of node u. The corresponding objective function is defined as:

$$\arg\max_{f} \prod_{u \in U} \prod_{u' \in N_s(u)} P(u'|f(u)) \tag{6.4}$$

Here, the conditional probability $P(u'|f(u))$ is modeled as a softmax function.

6.4.2 Recommendation System

We take a supervised learning approach towards the recommendation task. For each user, we use the graph embedding as described above in 6.4.1 to learn embeddings. We further define a feature vector $\vec{x}_{\{u,v\}}$ for a user pair $\{u, v\}$ as the Hadamard distance* between embeddings of u and v. Given two vectors $f(u)$ and $f(v)$, the Hadamard operator $\boxdot$ is defined as: $[f(u) \boxdot f(v)]_i = [f(u)]_i[f(v)]_i$. We train a random forest classifier to learn associations between the feature vectors of user pairs $\vec{x}_{\{u,v\}}$ and presence or absence of friendships between them.

The trained model is then used to predict class probabilities for unseen candidate pairs using their corresponding feature vectors $\vec{x}_{\{u,v\}}$. We use the positive class probability as the recommendation score. A friend v is recommended to a user u, if the recommendation score for the pair $\{u, v\} \notin E$ is within the top $k\%$ of all the scores received by all candidate pairs. We denote the resulting friendship recommendations given to a user u by $\rho(u)$.

*While other binary operators (e.g., average, L1, L2 distance) could be used, we chose Hadamard distance, as it performed the best in original node2vec paper [38]

6.4.3 Fairness of node2vec

We now evaluate the fairness of node2vec-based friendship recommendation using *Equality of Representation* at user level, bias^{ERu} (Eq. (6.3)). In a fair recommendation system, we would expect the bias to be lower than the initial bias. To calculate the initial bias in the network, we modify Eq. (6.3) by using the neighborhood function ω instead of recommendation ρ. We do a rigorous bias evaluation using all metrics in Section 6.6.

We distinguish between two genders $\mathcal{Z}^g = \{z_0^g, z_1^g\}$ and three races: $\mathcal{Z}^r = \{z_0^r, z_1^r, z_2^r\}$ and the protected (minority) groups are: z_1^g, z_0^r, z_2^r (we describe our dataset in details in Section 6.6.1). Figure 6.1 shows the distribution of z-share values for all users. The distribution of z-share indeed follows the original distribution in the network, and so does the bias. This demonstrates that node2vec based recommendations mirrors the gap between minorities and majorities.

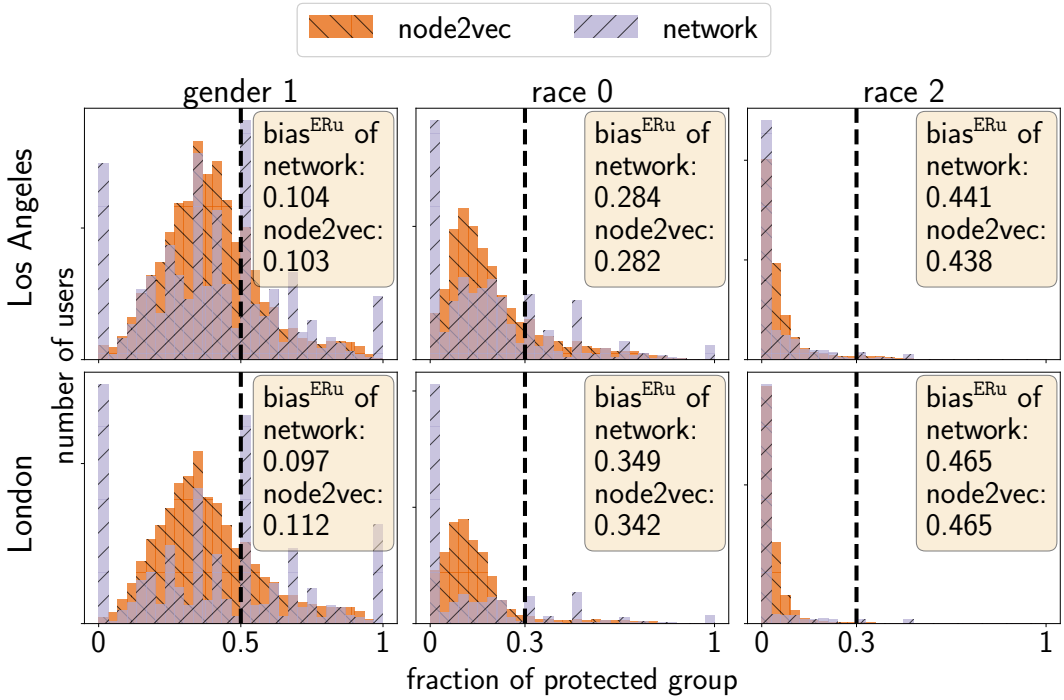


Figure 6.1: z-share distributions of node2vec and original network. The vertical line shows the *fair fraction* (0.5 and 0.3)

6.5 Fairwalk

Since graph embeddings aim to find the best structural representation of nodes, the algorithm also unintentionally learns some information about people's sensitive attributes and relations between them. Recommendations based on this further reinforces differences between minorities and majorities. As a countermeasure to this problem, we

Algorithm 6.1 Fair random walk trace generation

procedure RAND_WALK($U, \omega,$ `walk_num`, `walk_len`)
  traces ← empty_list
  for all $u \in U$ **do**
    for $i \leftarrow 0,$ `walk_num` **do**
      trace ← empty_list
      $u_1 \leftarrow u$
      for $j \leftarrow 0,$ `walk_len` **do**
        trace.append(u_1)
        $\mathcal{Z}_u \leftarrow \{z : z \in \mathcal{Z} \wedge |\omega_z(u_1)| > 0\}$
        $z_1 \xleftarrow{R} \mathcal{Z}_u$
        $v \xleftarrow{R} \omega_{z_1}(u_1)$
        $u_1 \leftarrow v$
      end for
      traces.append(trace)
    end for
  end for
  return traces
end procedure

propose *Fairwalk*, a modified version of random walk, which results in a more diverse *network neighborhood* representation thereby producing less biased graph embedding.

6.5.1 Random Walk

We are modifying the random walk procedure from original node2vec. Instead of randomly selecting a node to jump to from amongst all neighbors, we now partition neighbors into groups based on their sensitive attribute values and give each group the same probability of being chosen regardless of their sizes. Then a random node from the chosen group is selected for the jump. The modified random walk procedure is shown in Algorithm 6.1. By $\xleftarrow{R} \mathbb{A}$ we denote drawing an element of set $\mathbb{A}$ uniformly at random.

6.5.2 Resulting Traces

While detailed evaluation of the resulting modified embedding is in Section 6.6, we can already discuss differences in the walk traces themselves. From Figure 6.2 we can see that minorities appear more often in the fair random walk. Not only does it result in higher *network neighborhood* diversity with respect to sensitive attribute, but also minorities appear more frequently in the traces. This provides more *network neighborhood* data for minorities which enables the neural network give more importance to obtain their best vector representation while optimizing the overall objective function.

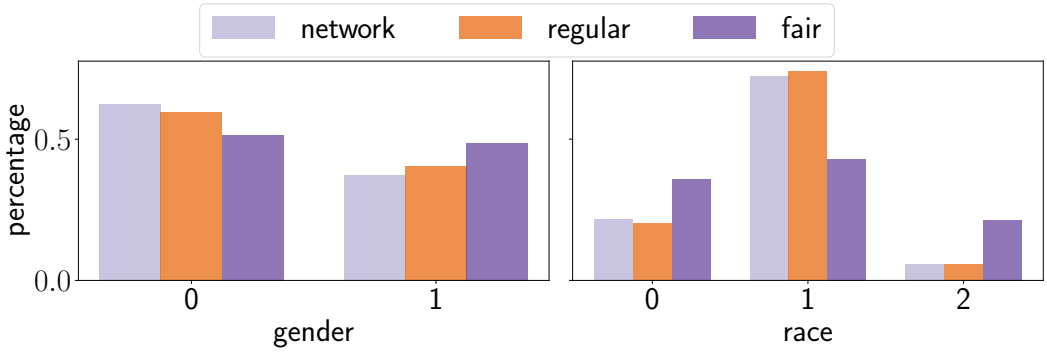


Figure 6.2: Ratio of each gender and race in the original network and regular and fair random walk traces in Los Angeles dataset

	Gender groups			
$i-j$ for G_{ij}	0-0	0-1	1-0	1-1
LA	37.78	21.00	21.99	19.21
London	38.96	18.19	20.74	22.10

Table 6.1: Percentage of existing friendships in each gender based group in our original dataset

6.6 Evaluation

In this section, we first describe our dataset, followed by the set-up of the machine learning model for our experiments. We then present the evaluation results for our *Fairwalk* using the fairness metrics defined in Section 6.3 and finally show the recommendation utility in terms of precision and recall.

6.6.1 Dataset

We use Instagram data collected from two of the biggest cities in different English speaking countries, namely London and Los Angeles (LA). The data was collected in 2016 using the Instagram API. An edge exists between two users if they mutually follow each other (e.g., [20]). We concentrate on two sensitive attributes, gender and race,

	Race groups								
$i-j$ for G_{ij}	0-0	0-1	1-0	1-1	0-2	1-2	2-0	2-1	2-2
LA	5.97	12.55	12.65	57.92	1.17	3.87	1.16	3.74	0.96
London	3.55	9.52	9.83	70.31	0.47	2.57	0.56	2.76	0.41

Table 6.2: Percentage of existing friendships in each race based group in our original dataset

	LA	London
No. users	82,607	53,902
No. social links	482,305	165,184
gender 0	62.6%	62.3%
gender 1	37.4%	37.7%
race 0	21.9%	15.9%
race 1	72.2%	80.7%
race 2	5.9%	3.4%

Table 6.3: Statistics of both datasets.

which we extract by querying Face++ with users' profile pictures. Tools such as Face++ have limitations [15]. To this end, we choose users for whom the attribute with highest confidence is at least 20 percentage points higher then the second best. We distinguish between female or male gender, and denote them as 0 or 1, i.e., $\mathcal{Z}^g = \{0, 1\}$, and between African, Caucasian or Asian race and denote them as 0, 1 or 2, i.e., $\mathcal{Z}^r = \{0, 1, 2\}$. Table 6.3 shows the sizes of datasets and the proportions of different genders and races. We thus have 4 pairwise groups for gender namely $G_{z_0,z_0}, G_{z_0,z_1}, G_{z_1,z_0}, G_{z_1,z_1}$ and 9 pairwise groups for race namely $G_{z_i,z_j}, i, j \in \{0, 1, 2\}$. Table 6.1 and 6.2 shows the proportions of friendships in different groups.

6.6.2 Experimental Setup

We iterate our experiments 5 times. To this end we divide our dataset into 5 equal slices. We train a random forest with 100 trees using 4 out of 5 slices, leaving out a different slice each time. We use Hadamard distance between the feature embeddings of each user pair as input features to the random forest. Graph embeddings are trained 5 times, each time with the same 4 slices as used for training. We use the following hyper-parameters (following [38]), i.e., length of each walk: `walk_len` = 80, number of walks starting from each node in the graph: `walk_num` = 20, number of dimensions of the resulting vector space: $d = 128$. For the recommendation candidates, of each node u we rank all non-friend nodes by the cosine similarity of their embeddings with the embedding of u and select the top 100. This approach gives us a candidate set likely to be recommended. For scalability reasons (for bigger datasets), non-friend pairs can be sampled randomly. For the quantity of top recommendations, we experiment with a variety of values for k and do not observe any significant differences. Therefore we show results below only for $k = 20$ i.e., when the top 20% of all candidates are selected for recommendation. We denote node2vec based recommendations by "regular" and *Fairwalk* based recommendation by "fair".

6.6.3 Statistical Imparity

Figure 6.3 shows the acceptance rates $P(G_{ij}^{\mathcal{S}})$(fraction of recommended users pairs out of all possible pairs in each group) for both the regular and the *Fairwalk*. We observe that, *Fairwalk* increases the probability of the under represented groups being recommended

		LA		London	
		gender	race	gender	race
ERg	regular	$1.3e^{10}$	$2.5e^{7}$	$6.5e^{9}$	$2.4e^{7}$
	fair	$0.8e^{10}$	$1.9e^{7}$	$4.8e^{9}$	$1.9e^{7}$
fSI	regular	$4.7e^{-9}$	$1.4e^{-12}$	$1.1e^{-8}$	$7.1e^{-11}$
	fair	$1.7e^{-9}$	$0.4e^{-12}$	$0.2e^{-8}$	$2.8e^{-11}$

Table 6.4: bias^{SI} and bias^{ERg} for both cities (lower, the better)

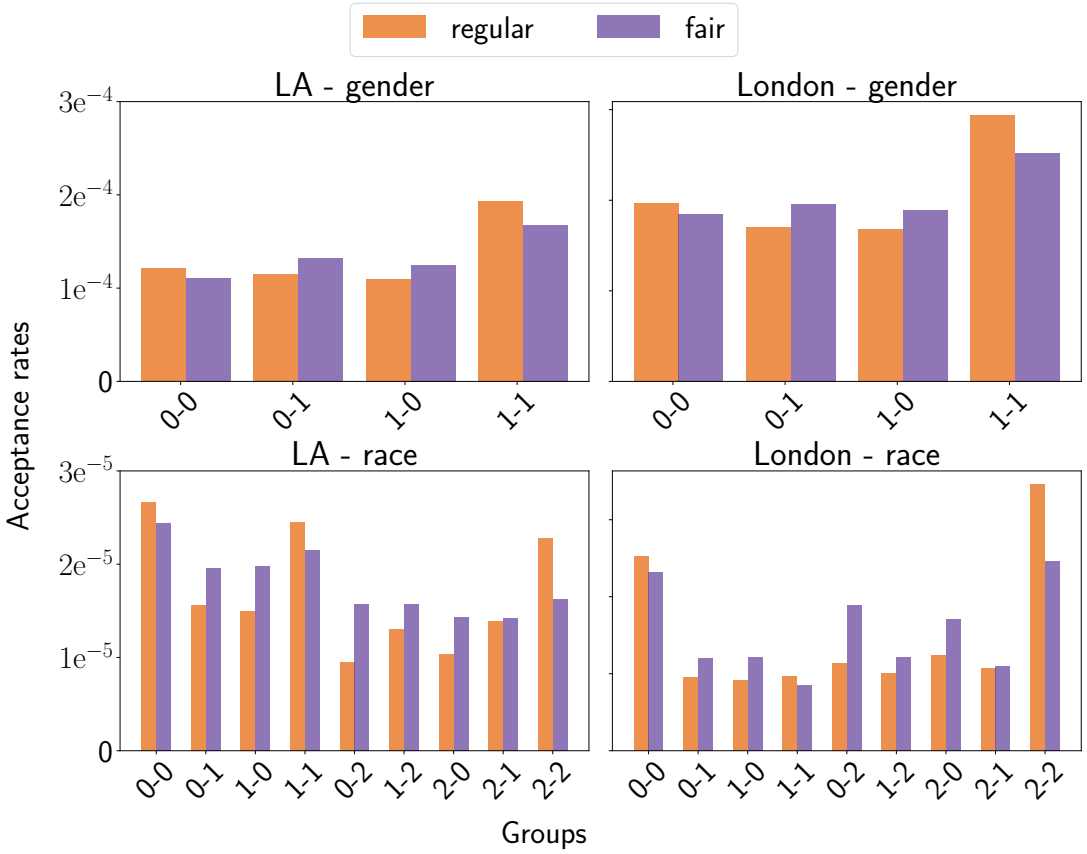


Figure 6.3: Fraction of recommended users pairs out of all possible pairs in each group. The x-axes marks the Type-2 groups G_{z_i,z_j} with the corresponding $i - j$

to a very large extent. Additionally, it is noteworthy to see that the probabilities of same gender and same race friendship recommendations are always reduced by *Fairwalk* and the probabilities of diverse friendships are always increased.

Table 6.4 shows bias^{SI} compared to node2vec for both cities. *Fairwalk* reduces $\text{bias}^{\text{SI}}(\mathcal{G}^g)$ by 61% and $\text{bias}^{\text{SI}}(\mathcal{G}^r)$ by 68% for LA. For London, *Fairwalk* reduces $\text{bias}^{\text{SI}}(\mathcal{G}^g)$ by 91% and $\text{bias}^{\text{SI}}(\mathcal{G}^r)$ by 61% compared to node2vec.

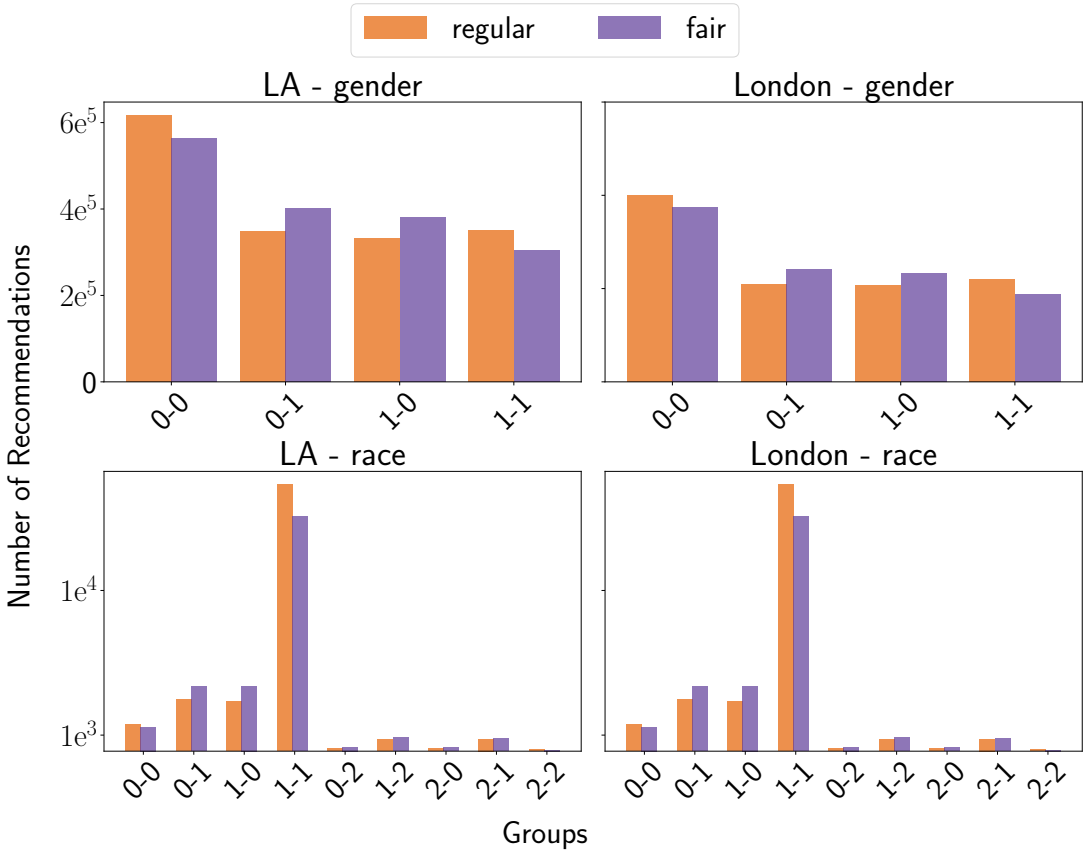


Figure 6.4: Number of recommended users pairs from each group. The x-axes marks groups G_{ij} with the corresponding $i - j$

6.6.4 Equality of Representation

Group Level. Figure 6.4 highlights the differences between the representations of different groups among the top recommendations. We see that as compared to node2vec, *Fairwalk* decreases the share for the over-represented groups, e.g., $\mathcal{G}^r_{11}$ and $\mathcal{G}^g_{00}$ and increases the share for most underrepresented groups which can be clearly seen for $\mathcal{G}^r_{01}$, $\mathcal{G}^r_{10}$, $\mathcal{G}^r_{02}$, $\mathcal{G}^r_{12}$ and $\mathcal{G}^g_{10}$, $\mathcal{G}^g_{01}$. For $\mathcal{G}^g_{11}$ and $\mathcal{G}^r_{00}$, *Fairwalk* could not increase the representation anymore since they are already very strongly connected as seen in Fig 6.3. Table 6.4 shows bias^{ERg} for both cities. Compared to node2vec, *Fairwalk* reduces $\text{bias}^{\text{ERg}}(\mathcal{G}^g)$ by 36% and $\text{bias}^{\text{ERg}}(\mathcal{G}^r)$ by 23% for LA and $\text{bias}^{\text{ERg}}(\mathcal{G}^g)$ by 26% and $\text{bias}^{\text{ERg}}(\mathcal{G}^r)$ by 21% for London.

User Level. On the distribution plots on Figures 6.5 we can see that *Fairwalk* reduces the gap between different groups, by leaning towards the *fair fraction* (where each attribute value has an equal share). We skipped the distribution for gender 1 since it's symmetrical to gender 0. The exact values of bias^{ERu} for different groups for the original network (initial bias) for node2vec and *Fairwalk* can be found in Table 6.5. We see that

		gender		race		
		0	1	0	1	2
LA	network	0.104	0.104	0.117	0.392	0.275
	node2vec	0.103	0.103	0.115	0.387	0.272
	fairwalk	0.068	0.068	0.054	0.288	0.234
London	network	0.097	0.097	0.183	0.481	0.298
	node2vec	0.112	0.112	0.176	0.474	0.298
	fairwalk	0.095	0.095	0.135	0.417	0.282

Table 6.5: Bias by *Equality of Representation* at user level for both genders and all three races (lower, the better).

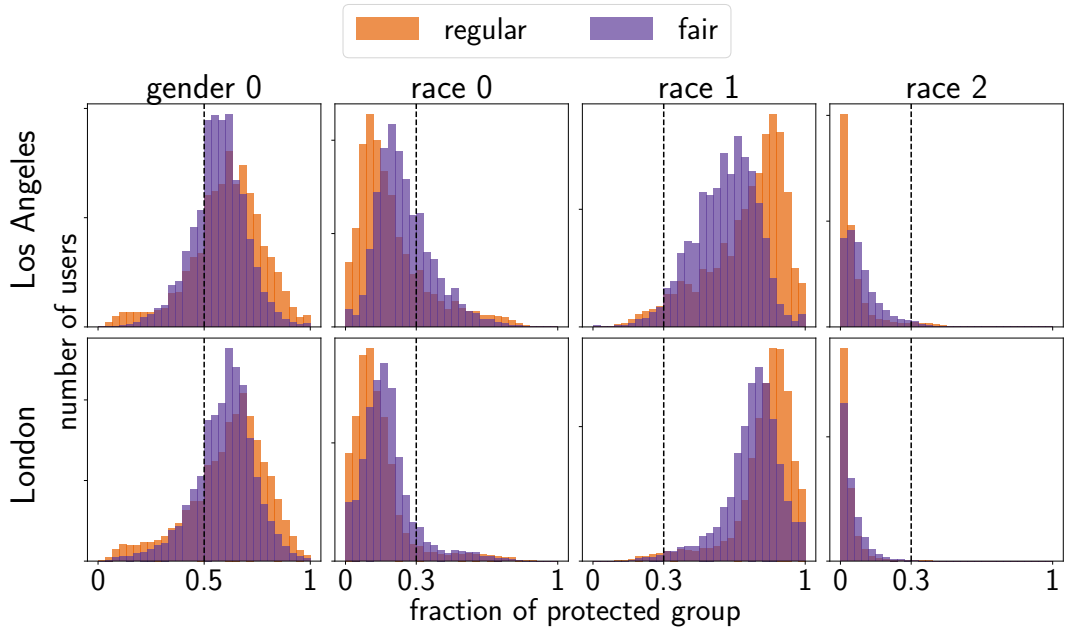


Figure 6.5: z-share distributions of node2vec and *Fairwalk*. The vertical line shows the *fair fraction*.

Fairwalk decreased the bias in all cases, for LA the average improvement is 32% and for London 14%. The best result is for race 0 in LA, with an improvement of 53%.

6.6.5 Precision and Recall

We also evaluate whether our recommendations capture any friendship edges that would otherwise have been formed naturally without users following any recommendations at all. To this end, we use the 20% of friendship edges unused during training for each iteration as described in Sec. 6.6.2 as ground truth. We calculate precision and recall for *Fairwalk*, and compare with the regular node2vec recommendations.

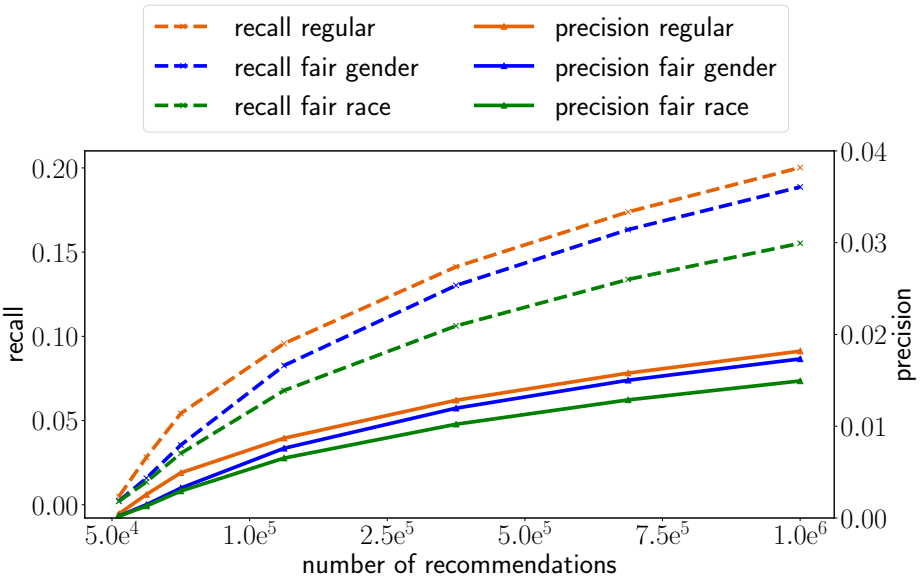


Figure 6.6: Precision and recall for different number of recommendations.

Figure 6.6 shows precision and recall for different number of recommendations for LA. Unsurprisingly, we see that precision and recall are always lower for *Fairwalk* as compared to node2vec. This indicates that we deviate more from the original biased growth of the network compared to regular recommendations thereby not amplifying the initial bias in the network. We also see that for race, *Fairwalk* deviates the most since it tries to balance more groups compared to gender, and given that race 1 alone makes up a huge proportion of our OSN dataset. Nevertheless, we capture a large number of true positives. Results for London follow the same trend.

6.7 Conclusion and Future Work

We study fairness issues in a state-of-the-art graph embedding method node2vec. Using the metrics of *Statistical Parity* and *Equality of Representations* we find bias in node2vec. We propose a fairness-aware *Fairwalk* and demonstrate its effectiveness in mitigating the aforementioned biases by a large scale evaluation on real world OSN datasets for friendship recommendation.

While *Fairwalk* already improves *Statistical Parity* and *Equality of Representations* to a large extent, there exist multiple other notions of fairness in the community dependent on the task at hand. Our *Fairwalk* can be tuned with parameters to fulfill any such fairness notion, for example maintaining the real-world ratio between groups. One could also iteratively perform further recommendation to achieve a higher balance or parity. Using a weighted ensemble-like approach that allow optimizing for a combination of sensitive attributes (e.g Asian females) can account for cross-attribute biases in the network. This can also be done following recent work on rich subgroup fairness [59].

7
Conclusion

7.1 Conclusion

In the past decade, the field of information and communication technologies has made remarkable achievements. Tools such as Online Social Networks and fitness trackers have become a part-and-parcel of people's daily lives, serving a myriad of purposes. Online social networks are used not only for maintaining relationships with people far and near, but also as a platform for self expression, advertising and exchanging information and news at the touch of a finger (to name a few). Fitness trackers are increasingly used by people to monitor the quality of their lives such as physical activity, sleep, heart rate etc. Additionally, smartphones, smart watches and other wearables also measure physical activity of their users regardless of whether the users are actively conscious of their health or not. This results in users generating a plethora of data continuously every minute. Therefore, the implications of these data on the privacy of the users has become an extremely pressing matter. These data are increasingly used to further train machine learning algorithms, with an aim to automate several prediction, classification and recommendation tasks which otherwise take countless days to be done by humans. These tasks are in turn at the core of many functionalities offered by OSNs and fitness trackers. The data that is generated, collected and used for training however is not always perfect in every respect and suffer from various kinds of biases. Therefore algorithms are often quite sensitive to bias that is prevalent in the training data and sometimes produces undesirable and unfair consequences. in this dissertation, we have systematically and extensively analyzed privacy and fairness issues relating to data generated by users of online social networks and activity trackers.

The first work presents the first scientific analysis of privacy issues induced by hashtag sharing behaviour of users. We concentrate in particular on location, which has been recognized as one of the key privacy concerns in the modern era. We empirically evaluate our attack based on a comprehensive dataset of 239,000 Instagram posts with corresponding locations and hashtags from three of the largest English-speaking cities (New York, Los Angeles and London). By relying on a random forest model, we show that we can infer a user's precise location with accuracy of 70% to 76%, depending on the city. We also compare our classification method with other popular algorithms such as support vector machine and gradient boosting machine, and show that our classifier outperforms them by at least 7%. Furthermore, we empirically identify the number of hashtags maximizing the attack success to be seven. Additionally, we evaluate our attack on a global level without prior knowledge of the city, and demonstrate that it can still reach an accuracy of more than 70%. Finally, by considering two types of attackers, one with prior knowledge on the targeted users' hashtags and locations, and the other without, we show that the accuracy drops by around 20% for the latter attacker. This demonstrates that the adversary can enhance his model by learning per-user associations between hashtags and locations. To remedy this situation, we introduce a system called Tagvisor that systematically suggests alternative hashtags if the user-selected ones constitute a threat to location privacy. Tagvisor realizes this by means of three conceptually different obfuscation techniques: hiding (a subset of) hashtags, replacing hashtags by semantically similar hashtags, and generalizing hashtags with higher-level semantic categories (e.g., Starbucks into coffee shop). It also uses a

semantics-based metric for measuring the consequent utility loss. Our findings show that the higher number of original hashtags, the better the utility for similar levels of privacy. Moreover, obfuscating as little as two hashtags already provides a near-optimal trade-off between privacy and utility in our dataset. The latter finding notably demonstrates the practical feasibility of our privacy-preserving system given the computational capabilities of current mobile devices.

In the second work, we analysed OSN privacy by jointly exploiting long term multimodal information. We focussed in particular on inference of social relationships. We introduce three novel monomodal friendship inference attacks based on hashtags, images, and text, respectively. To the best of our knowledge, this is the first work that uses text and image modalities to conduct friendship inference in social networks. Our hashtag attack massively outperforms the only prior work on friendship inference from hashtags by 7%. We use self-engineered statistical features, such as the entropy and frequency of usage patterns for our hashtag and text based attacks. For images, we rely on a state-of-the-art ResNet model to extract information from images. Experiments show that our hashtag and text attacks achieve strong performance with AUCs above 0.86 and 0.83 respectively. Our image attack, while being the weakest, still performs well with AUCs up to 0.637. Further, we design a multimodal attack that combines five components of posts namely images, hashtags, captions, geo-locations and published friendships, to infer social relations. Extensive experiments show that this attack outperforms all monomodal attacks with AUCs above 0.9. We also show that the multimodal attack exploits post components that compliment each other and the success of most attacks increases significantly when combined with additional components. We further demonstrate the robustness of the multimodal attack against information hiding, and show that even after deleting 50% of the users' posts, the drop in the attack performance is relatively low. Potential future works would be to analyse data collected from other OSNs or other cities all around the world and considering the date or time component when two users shared certain posts to make the attacker stronger.

In the third work, we performed the first systematic study of privacy risks arising from step count data. In particular we focus on attribute inference for gender, age and education and temporal linkability of users. With a relatively small dataset of 1000 users only and using simple machine learning classifiers, we find that an adversary having access to actimetry (step count) data of different users can perform a linkability attack with a high confidence. Attribute inference is harder, but still possible. Among all three attributes, prediction of age is the easiest, while education cannot be predicted reliably. For gender inference specifically, we obtain promising results by averaging predictions from classifying shorter walking periods (activities). Some of these short patterns strongly indicate users' gender but not education or age. Exploring such smaller patterns for different attributes would be an interesting direction for future work. Actimetry data is not currently regarded as privacy sensitive, but big players, having access to an even bigger dataset could deanonymize users and infer their hidden attributes much easier. We would like to bring the attention of policymakers to the sensitivity of pedometer data and propose that users should be made aware of the risks of sharing their step count data. This highlights the need for further research on privacy preserving ways of collecting such data.

In the final work we conduct the first-of-its-kind study of algorithmic fairness in the setting of graph embedding methods. We specifically analyze group fairness (disparate impact) of the state-of-the-art graph embedding: node2vec. We extend *statistical parity*, a well-known measure of disparate impact to measure fairness for groups based on sensitive attributes of pairs of users. We further propose a novel notion of *Equality of Representation* to measure fairness in friendship recommendation systems. We apply node2vec for friendship recommendation in real world OSN datasets as a case study. Using the fairness metrics above, we find biases in the recommendations caused by unfair graph embeddings. To mitigate the aforementioned biases, we propose a novel fairness-aware graph embedding algorithm *Fairwalk*, that extends node2vec. Evaluations on our OSN demonstrate the effectiveness of *Fairwalk* on both datasets w.r.t. all fairness metrics. Compared to node2vec, *Fairwalk* increases fairness in each case by a large margin. While *Fairwalk* already improves *Statistical Parity* and *Equality of Representations* to a large extent, there exist multiple other notions of fairness in the community dependent on the task at hand. Interesting directions for future work include accounting for cross-attribute biases and parameterizing *Fairwalk* to fulfill any alternative fairness notion, for example maintaining the real-world ratio between groups.

Bibliography

Author's Papers for this Thesis

[P1] Zhang, Y., Humbert, M., Rahman, T., Li, C.-T., Pang, J., and Backes, M. Tagvisor: a privacy advisor for sharing hashtags. In: *Proceedings of the 2018 World Wide Web Conference (WWW)*. 2018, 287–296.

[P2] Rahman, T., Fritz, M., Backes, M., and Zhang, Y. Everything about you: a multimodal approach towards friendship inference in online social networks. In: *arXiv:2003.00996*. 2018.

[P3] Surma, B., Rahman, T., Backes, M., and Zhang, Y. You are how you walk: quantifying privacy risks in step count data. In: *arXiv:2308.04933*. 2021.

[P4] Rahman, T., Surma, B., Backes, M., and Zhang, Y. Fairwalk: towards fair graph embedding. In: *Proceedings of the 2019 International Joint Conferences on Artifical Intelligence (IJCAI)*. 2019.

Other references

[1] Acquisti, A. and Gross, R. Predicting social security numbers from public data (2009).

[2] Adamic, L. A. and Adar, E. Friends and neighbors on the web. *Social networks* 25, 3 (2003), 211–230.

[3] Agarwal, P., Vaithiyanathan, R., Sharma, S., and Shroff, G. Catching the Long-Tail: Extracting Local News Events from Twitter. In: *Proceedings of the 6th International Conference on Weblogs and Social Media (ICWSM)*. The AAAI Press, 2012, 379–382.

[4] Ağır, B., Huguenin, K., Hengartner, U., and Hubaux, J.-P. On the Privacy Implications of Location Semantics. *Proceedings on Privacy Enhancing Technologies* 2016, 4 (2016), 165–183.

[5] Aguiar, B., Silva, J., Rocha, T., Carneiro, S., and Sousa, I. Monitoring physical activity and energy expenditure with smartphones. In: *2014 IEEE-EMBS International Conference on Biomedical and Health Informatics, BHI 2014*. June 2014, 664–667.

[6] Al Zamal, F., Liu, W., and Ruths, D. Homophily and latent attribute inference: inferring latent attributes of twitter users from neighbors. In: *Sixth International AAAI Conference on Weblogs and Social Media*. 2012.

[7] Althoff, T., Hicks, J. L., King, A. C., Delp, S. L., Leskovec, J., et al. Large-scale physical activity data reveal worldwide activity inequality. *Nature* 547, 7663 (2017), 336–339.

[8] Backes, M., Humbert, M., Pang, J., and Zhang, Y. walk2friends: Inferring Social Links from Mobility Profiles. In: *Proceedings of the 24th ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2017, 1943–1957.

[9] Barocas, S. and Selbst, A. D. Big data's disparate impact. *Cal. L. Rev.* 104 (2016), 671.

[10] Bilogrevic, I., Huguenin, K., Mihaila, S., Shokri, R., and Hubaux, J.-P. Predicting Users' Motivations behind Location Check-ins and Utility Implications of Privacy Protection Mechanisms. In: *Proceedings of the 22nd Network and Distributed System Security Symposium (NDSS)*. 2015.

[11] Bittel, A., Elazzazi, A., and Bittel, D. Accuracy and precision of an accelerometer-based smartphone app designed to monitor and record angular movement over time. 22 (Oct. 2015).

[12] Bojinov, H., Michalevsky, Y., Nakibly, G., and Boneh, D. Mobile device identification via sensor fingerprinting. *CoRR* abs/1408.1416 (2014). arXiv: 1408.1416.

[13] Bostrom, H. Estimating Class Probabilities in Random Forests. In: *Proceedings of the 6th International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2007, 211–216.

[14] Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., and Shah, R. Signature verification using a" siamese" time delay neural network. In: *Advances in neural information processing systems*. 1994, 737–744.

[15] Buolamwini, J. and Gebru, T. Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification. In: *Proc. FAT**. 2018, 77–91.

[16] Calders, T. and Verwer, S. Three naive bayes approaches for discrimination-free classification. *Data Mining and Knowledge Discovery* 21, 2 (2010), 277–292.

[17] Chandra, S., Khan, L., and Muhaya, F. B. Estimating Twitter User Location Using Social Interactions - A Content Based Approach. In: *Proceedings of the 3rd International Conference on Privacy, Security, Risk and Trust (PASSAT)*. IEEE, 2011, 838–843.

[18] Cheng, Z., Caverlee, J., and Lee, K. 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 (CIKM)*. ACM, 2010, 759–768.

[19] Cho, E., Myers, S. A., and Leskovec, J. Friendship and mobility: user movement in location-based social networks. In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2011, 1082–1090.

[20] Cho, E., Myers, S. A., and Leskovec, J. Friendship and Mobility: User Movement in Location-based Social Networks. In: *Proceedings of the 17th ACM Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 2011, 1082–1090.

[21] Ciot, M., Sonderegger, M., and Ruths, D. Gender inference of twitter users in non-english contexts. In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 2013, 1136–1145.

[22] Crandall, D. J., Backstrom, L., Cosley, D., Suri, S., Huttenlocher, D., and Kleinberg, J. Inferring social ties from geographic coincidences. 107, 52 (2010), 22436–22441.

[23] Crandall, D. J., Backstrom, L., Cosley, D., Suri, S., Huttenlocher, D., and Kleinberg, J. Inferring Social Ties from Geographic Coincidences. *Proceedings of the National Academy of Sciences* 107, 52 (2010), 22436–22441.

[24] Dalvi, N., Kumar, R., and Pang, B. Object Matching in Tweets with Spatial Models. In: *Proceedings of the 5th ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 2012, 43–52.

[25] Davarci, E., Soysal, B., Erguler, I., Aydin, S. O., Dincer, O., and Anarim, E. Age group detection using smartphone motion sensors. In: *2017 25th European Signal Processing Conference (EUSIPCO)*. 2017, 2201–2205.

[26] Dey, R., Jelveh, Z., and Ross, K. Facebook users have become much more private: a large-scale study. In: *Proc. 2012 IEEE International Conference on Pervasive Computing and Communications Workshops*. IEEE, 2012, 346–352.

[27] Dey, S., Roy, N., Xu, W., Choudhury, R., and Nelakuditi, S. Accelprint: imperfections of accelerometers make smartphones trackable. In: Jan. 2014.

[28] Dwork, C., Hardt, M., Pitassi, T., Reingold, O., and Zemel, R. Fairness Through Awareness. In: *Proc. ITCS*. 2012, 214–226.

[29] Eagle, N., Pentland, A. S., and Lazer, D. Inferring friendship network structure by using mobile phone data. 106, 36 (2009), 15274–15278.

[30] Feldman, M., Friedler, S. A., Moeller, J., Scheidegger, C., and Venkatasubramanian, S. Certifying and Removing Disparate Impact. In: *Proc. KDD*. 2015, 259–268.

[31] Flaxman, S., Goel, S., and Rao, J. M. Filter bubbles, echo chambers, and online news consumption. *Public Opinion Quarterly* 80, S1 (2016), 298–320.

[32] Getoor, L. and Diehl, C. P. Link mining: a survey. 7, 2 (Dec. 2005).

[33] Goga, O., Loiseau, P., Sommer, R., Teixeira, R., and Gummadi, K. P. On the reliability of profile matching across large online social networks. In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2015, 1799–1808.

[34] Gong, N. Z. and Liu, B. You are who you know and how you behave: attribute inference attacks via users' social friends and behaviors. In: *25th USENIX Security Symposium (USENIX Security 16)*. 2016, 979–995.

[35] Gong, N. Z. and Liu, B. Attribute inference attacks in online social networks. *ACM Trans. Priv. Secur.* 21, 1 (Jan. 2018).

[36] Goyal, P. and Ferrara, E. Graph embedding techniques, applications, and performance: a survey. *Knowledge-Based Systems* 151 (2018), 78–94.

[37] Gross, R. and Acquisti, A. Information revelation and privacy in online social networks. In: *Proceedings of the 2005 ACM workshop on Privacy in the electronic society*. ACM. 2005, 71–80.

[38] Grover, A. and Leskovec, J. node2vec: Scalable Feature Learning for Networks. In: *Proc. KDD*. 2016, 855–864.

[39] Guidoux, R., Duclos, M., Fleury, G., Lacomme, P., Lamaudière, N., Manenq, P.-H., Paris, L., Ren, L., and Rousset, S. A smartphone-driven methodology for estimating physical activities and energy expenditure in free living conditions. 52 (July 2014).

[40] Han, K., Lee, S., Jang, J. Y., Jung, Y., and Lee, D. Teens Are from Mars, Adults Are from Venus: Analyzing and Predicting Age Groups with Behavioral Characteristics in Instagram. In: *Proceedings of the 8th ACM Conference on Web Science (WebSci)*. ACM, 2016, 35–44.

[41] Han, K., Lee, S., Jang, J. Y., Jung, Y., and Lee, D. Teens are from mars, adults are from venus: analyzing and predicting age groups with behavioral characteristics in Instagram. In: *Proceedings of the 8th ACM Conference on Web Science (WebSci)*. ACM, 2016, 35–44.

[42] Hardt, M., Price, E., and Srebro, N. Equality of Opportunity in Supervised Learning. In: *Proc. NIPS*. 2016, 3315–3323.

[43] Hargreaves, E., Agosti, C., Menasché, D., Neglia, G., Reiffers-Masson, A., and Altman, E. Fairness in online social network timelines: measurements, models and mechanism design. *Performance Evaluation* 129 (2019), 15–39.

[44] Hassan, W. U., Hussain, S., and Bates, A. Analysis of privacy protections in fitness tracking social networks-or-you can run, but can you hide? In: *27th {USENIX} Security Symposium ({USENIX} Security 18)*. 2018, 497–512.

[45] He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition (2016), 770–778.

[46] Highfield, T. and Leaver, T. A Methodology for Mapping Instagram Hashtags. *First Monday* 20, 1 (2014).

[47] Highfield, T. and Leaver, T. A methodology for mapping Instagram hashtags. *First Monday* 20, 1 (2014).

[48] *How the NSA is tracking people right now*. `https://www.washingtonpost.com/apps/g/page/world/how-the-nsa-is-tracking-people-right-now/634/`. 2017.

[49] Humbert, M., Studer, T., Grossglauser, M., and Hubaux, J.-P. Nowhere to Hide: Navigating around Privacy in Online Social Networks. In: *Proceedings of the 18th European Symposium on Research in Computer Security (ESORICS)*. Springer, 2013, 682–699.

[50] Ilia, P., Polakis, I., Athanasopoulos, E., Maggi, F., and Ioannidis, S. Face/off: preventing privacy leakage from photos in social networks. In: *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*. CCS '15. 2015, 781–792.

[51] *iPhone 7 Series is Faster Than Any MacBook Air Ever Made*. https://www.macrumors.com/2016/09/15/iphone-7-faster-than-macbook-air/. 2016.

[52] Jia, J. and Gong, N. Z. Attriguard: a practical defense against attribute inference attacks via adversarial machine learning. In: *27th {USENIX} Security Symposium ({USENIX} Security 18)*. 2018, 513–529.

[53] Jiang, Y., Tang, W., Gao, N., Xiang, J., Zha, D., and Li, X. Your pedometer tells you: attribute inference via daily walking step count. In: *2019 IEEE SmartWorld, Ubiquitous Intelligence Computing, Advanced Trusted Computing, Scalable Computing Communications, Cloud Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*. 2019, 834–842.

[54] Jiang, Y., Tang, W., Gao, N., Xiang, J., Tu, C., and Li, M. Multiple demographic attributes prediction in mobile and sensor devices. In: *Advances in Knowledge Discovery and Data Mining - 24th Pacific-Asia Conference, PAKDD 2020, Singapore, May 11-14, 2020, Proceedings, Part I*. Ed. by Lauw, H. W., Wong, R. C.-W., Ntoulas, A., Lim, E.-P., Ng, S.-K., and Pan, S. J. Vol. 12084. Lecture Notes in Computer Science. Springer, 2020, 857–868.

[55] Jurgens, D., Finethy, T., McCorriston, J., Xu, Y. T., and Ruths, D. Geolocation Prediction in Twitter Using Social Networks: A Critical Analysis and Review of Current Practice. In: *Proceedings of the 9th International Conference on Weblogs and Social Media (ICWSM)*. The AAAI Press, 2015, 188–197.

[56] Kamiran, F., Calders, T., and Pechenizkiy, M. Discrimination Aware Decision Tree Learning. In: *Proc. ICDM*. 2010, 869–874.

[57] Kamishima, T., Akaho, S., Asoh, H., and Sakuma, J. Fairness-Aware Classifier with Prejudice Remover Regularizer. In: *Proc. ECML/PKDD*. 2012, 35–50.

[58] Karimi, F., Génois, M., Wagner, C., Singer, P., and Strohmaier, M. Homophily Influences Ranking of Minorities in Social Networks (2018).

[59] Kearns, M., Neel, S., Roth, A., and Wu, Z. S. An Empirical Study of Rich Subgroup Fairness for Machine Learning. In: *Proc. FAT**. 2019, 100–109.

[60] Kenter, T. and Rijke, M. de. Short Text Similarity with Word Embeddings. In: *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (CIKM)*. ACM, 2015, 1411–1420.

[61] KIM, Y. S., Choi, K. S., and Natali, F. Extending the network: the influence of offline friendship on twitter network. In: AIS. 2016.

[62] Kleinberg, J., Mullainathan, S., and Raghavan, M. Inherent trade-offs in the fair determination of risk scores. In: *LIPIcs*. Vol. 67. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2017.

[63] Lamba, H., Bharadhwaj, V., Vachher, M., Agarwal, D., Arora, M., Sachdeva, N., and Kumaraguru, P. From Camera to Deathbed: Understanding Dangerous Selfies on Social Media. In: *Proceedings of the 11th International Conference on Web and Social Media (ICWSM)*. The AAAI Press, 2017, 576–579.

[64] Li, W., Serdyukov, P., Vries, A. P. de, Eickhoff, C., and Larson, M. The Where in the Tweet. In: *Proceedings of the 20th ACM International Conference on Information and Knowledge Management (CIKM)*. ACM, 2011, 2473–2476.

[65] Liben-Nowell, D. and Kleinberg, J. The link-prediction problem for social networks. 58, 7 (2007), 1019–1031.

[66] Liu, S., Wang, S., Zhu, F., Zhang, J., and Krishnan, R. Hydra: large-scale social identity linkage via heterogeneous behavior modeling. In: *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. ACM. 2014, 51–62.

[67] Lü, L. and Zhou, T. Link prediction in complex networks: a survey. 390, 6 (2011), 1150–1170.

[68] Luong, M.-T., Pham, H., and Manning, C. D. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).

[69] Ma, L., Sun, Q., Georgoulis, S., Van Gool, L., Schiele, B., and Fritz, M. Disentangled person image generation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, 99–108.

[70] Manikonda, L., Hu, Y., and Kambhampati, S. Analyzing User Activities, Demographics, Social Network Structure and User-Generated Content on Instagram. *CoRR* abs/1410.8099 (2014).

[71] Mejova, Y., Abbar, S., and Haddadi, H. Fetishizing Food in Digital Age: #foodporn Around the World. In: *Proceedings of the 10th International Conference on Weblogs and Social Media (ICWSM)*. The AAAI Press, 2016, 250–258.

[72] Meteriz, Ü., Yıldıran, N. F., and Mohaisen, A. You can run, but you cannot hide: using elevation profiles to breach location privacy through trajectory prediction. *arXiv preprint arXiv:1910.09041* (2019).

[73] Mikolov, T., Chen, K., Corrado, G., and Dean, J. Efficient Estimation of Word Representations in Vector Space. In: *Proceedings of the 1st International Conference on Learning Representations (ICLR)*. 2013.

[74] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. Distributed Representations of Words and Phrases and their Compositionally. In: *Proceedings of the 27th Annual Conference on Neural Information Processing Systems (NIPS)*. NIPS, 2013, 3111–3119.

[75] Montjoye, Y.-A. D., Hidalgo, C. A., Verleysen, M., and Blondel, V. D. Unique in the Crowd: The Privacy Bounds of Human Mobility. 3 (2013), 1376.

[76] Narayanan, A., Shi, E., and Rubinstein, B. I. P. Link prediction by de-anonymization: how we won the kaggle social network challenge (2011).

[77] Narayanan, A. and Shmatikov, V. De-anonymizing social networks. In: *Security and Privacy, 2009 30th IEEE Symposium on*. IEEE. 2009, 173–187.

[78] Nguyen, K. A., Akram, R. N., Markantonakis, K., Luo, Z., and Watkins, C. Location tracking using smartphone accelerometer and magnetometer traces. In: *Proceedings of the 14th International Conference on Availability, Reliability and Security*. 2019, 1–9.

[79] Ni, M., Zhang, Y., Han, W., and Pang, J. An Empirical Study on User Access Control in Online Social Networks. In: *Proceedings of the 21st ACM on Symposium on Access Control Models and Technologies (SACMAT)*. ACM, 2016, 13–23.

[80] Olteanu, A.-M., Huguenin, K., Shokri, R., Humbert, M., and Hubaux, J.-P. Quantifying Interdependent Privacy Risks with Location Data. 16, 3 (2017), 829–842.

[81] Olteanu, A.-M., Huguenin, K., Shokri, R., Humbert, M., and Hubaux, J.-P. Quantifying interdependent privacy risks with location data. 16, 3 (2017), 829–842.

[82] Orekondy, T., Fritz, M., and Schiele, B. Connecting pixels to privacy and utility: automatic redaction of private information in images. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[83] Oshiro, T. M., Perez, P. S., and Baranauskas, J. A. How Many Trees in a Random Forest? In: *Proceedings of the 8th International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM)*. Springer, 2012, 154–168.

[84] Oya, S., Troncoso, C., and Pérez-González, F. Back to the Drawing Board: Revisiting the Design of Optimal Location Privacy-preserving Mechanisms. In: *Proceedings of the 24th ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2017, 1943–1957.

[85] Pang, J. and Zhang, Y. Location Prediction: Communities Speak Louder than Friends. In: *Proceedings of the 3rd ACM Conference on Online Social Networks (COSN)*. ACM, 2015, 161–171.

[86] Pang, J. and Zhang, Y. DeepCity: A Feature Learning Framework for Mining Location Check-Ins. In: *Proceedings of the 11th International Conference on Web and Social Media (ICWSM)*. The AAAI Press, 2017, 652–655.

[87] Pang, J. and Zhang, Y. Quantifying Location Sociality. In: *Proceedings of the 28th ACM Conference on Hypertext and Social Media (HT)*. ACM, 2017, 145–154.

[88] Pecli, A., Cavalcanti, M. C., and Goldschmidt, R. Automatic feature selection for supervised learning in link prediction applications: a comparative study. *Knowledge and Information Systems* 56, 1 (July 2018), 85–121.

[89] Pei, L., Guinness, R., Chen, R., Liu, J., Kuusniemi, H., Chen, Y., Chen, L., and Kaistinen, J. Human behavior cognition using smartphone sensors. *Sensors* 13 (Feb. 2013), 1402–1424.

[90] Perez, B., Musolesi, M., and Stringhini, G. You are your metadata: identification and obfuscation of social media users using metadata information. *arXiv preprint arXiv:1803.10133* (2018).

[91] Perozzi, B., Al-Rfou, R., and Skiena, S. DeepWalk: Online Learning of Social Representations. In: *Proc. KDD*. 2014, 701–710.

[92] Pham, H., Shahabi, C., and Liu, Y. Ebm: an entropy-based model to infer social strength from spatiotemporal data. In: *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. ACM. 2013, 265–276.

[93] Pierson, E., Althoff, T., and Leskovec, J. Modeling individual cyclic variation in human behavior. In: *Proceedings of the 2018 World Wide Web Conference*. 2018, 107–116.

[94] Preotiuc-Pietro, D., Ye, L., Hopkins, D., and Ungar, L. H. Beyond binary labels: political ideology prediction of twitter users. In: *ACL*. 2017.

[95] Pyrgelis, A., Troncoso, C., and Cristofaro, E. D. What Does The Crowd Say About You? Evaluating Aggregation-based Location Privacy. 2017, 4 (2017), 76–96.

[96] Pyrgelis, A., Troncoso, C., and Cristofaro, E. D. Knock Knock, Who's There? Membership Inference on Aggregate Location Data. In: *Proceedings of the 25th Network and Distributed System Security Symposium (NDSS)*. 2018.

[97] Quattrociocchi, W., Scala, A., and Sunstein, C. R. Echo chambers on facebook. *Available at SSRN 2795110* (2016).

[98] Romei, A. and Ruggieri, S. A multidisciplinary survey on discrimination analysis. *The Knowledge Engineering Review* 29, 5 (2014), 582–638.

[99] Scellato, S., Noulas, A., and Mascolo, C. Exploiting Place Features in Link Prediction on Location-based Social Networks. In: *Proceedings of the 17th ACM Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 2011, 1046–1054.

[100] Scellato, S., Noulas, A., and Mascolo, C. Exploiting place features in link prediction on location-based social networks. In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2011, 1046–1054.

[101] Shameli, A., Althoff, T., Saberi, A., and Leskovec, J. How gamification affects physical activity: large-scale analysis of walking challenges in a mobile application. In: *Proceedings of the 26th International Conference on World Wide Web Companion*. 2017, 455–463.

[102] Shokri, R., Theodorakopoulos, G., Boudec, J.-Y. L., and Hubaux, J.-P. Quantifying Location Privacy. In: *Proceedings of the 32nd IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2011, 247–262.

[103] Shoshitaishvili, Y., Kruegel, C., and Vigna, G. Portrait of a privacy invasion. 2015, 1 (2015), 41–60.

[104] Souza, F., Las Casas, D. de, Flores, V., Youn, S., Cha, M., Quercia, D., and Almeida, V. Dawn of the Selfie Era: The Whos, Wheres, and Hows of Selfies on Instagram. In: *Proceedings of the 3rd ACM Conference on Online Social Networks (COSN)*. ACM, 2015, 221–231.

[105] Stoica, A.-A., Riederer, C., and Chaintreau, A. Algorithmic Glass Ceiling in Social Networks: The effects of social recommendations on network diversity. In: *Proc. WWW*. 2018, 923–932.

[106] Sun, Q., Tewari, A., Xu, W., Fritz, M., Theobalt, C., and Schiele, B. A hybrid model for identity obfuscation by face replacement. *arXiv preprint arXiv:1804.04779* (2018).

[107] Tan, C., Card, D., and Smith, N. A. Friendships, rivalries, and trysts: characterizing relations between ideas in texts. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2017, 773–783.

[108] Tang, J., Qu, M., and Mei, Q. PTE: Predictive Text Embedding through Large-scale Heterogeneous Text Networks. In: *Proc. KDD*. 2015, 1165–1174.

[109] Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q. LINE: Large-scale Information Network Embedding. In: *Proc. WWW*. 2015, 1067–1077.

[110] Vitak, J., Liao, Y., Kumar, P., Zimmer, M., and Kritikos, K. Privacy attitudes and data valuation among fitness tracker users. In: *International Conference on Information*. Springer. 2018, 229–239.

[111] Wang, H., Li, Z., and Lee, W.-C. Pgt: measuring mobility relationship using personal, global and temporal factors. In: *Data Mining (ICDM), 2014 IEEE International Conference on*. IEEE. 2014, 570–579.

[112] Wang, P., Xu, B., Wu, Y., and Zhou, X. Link prediction in social networks: the state-of-the-art. 58, 1 (2015), 1–38.

[113] Yang, K. and Stoyanovich, J. Measuring fairness in ranked outputs. In: *SSDBM*. 2017.

[114] Yu, L., Hermann, K. M., Blunsom, P., and Pulman, S. Deep Learning for Answer Sentence Selection. *CoRR* abs/1412.1632 (2014).

[115] Zafar, M. B., Valera, I., Rodriguez, M. G., and Gummadi, K. P. Fairness Beyond Disparate Treatment & Disparate Impact: Learning Classification without Disparate Mistreatment. In: *Proc. WWW*. 2017, 1171–1180.

[116] Zehlike, M., Bonchi, F., Castillo, C., Hajian, S., Megahed, M., and Baeza-Yates, R. FA*IR: A Fair Top-k Ranking Algorithm. In: *Proc. CIKM*. 2017, 1569–1578.

[117] Zhang, Y. Language in our time: an empirical analysis of hashtags. In: *The World Wide Web Conference*. ACM. 2019, 2378–2389.

[118] Zhang, Y., Ni, M., Han, W., and Pang, J. Does #like4like Indeed Provoke More Likes? In: *Proceedings of the 2017 International Conference on Web Intelligence (WI)*. ACM, 2017, 179–186.

[119] Zhang, Y. and Pang, J. Distance and Friendship: A Distance-based Model for Link Prediction in Social Networks. In: *Proceedings of the 17th Asia-Pacific Web Conference (APWeb)*. Springer, 2015, 55–66.

[120] Zhong, Y., Yuan, N. J., Zhong, W., Zhang, F., and Xie, X. You Are Where You Go: Inferring Demographic Attributes from Location Check-ins. In: *Proceedings of the 8th ACM Conference on Web Search and Data Mining (WSDM)*. ACM, 2015, 295–304.

[121] Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., and Torralba, A. Places: a 10 million image database for scene recognition (2017).

[122] Zhou, F., Wu, B., Yang, Y., Trajcevski, G., Zhang, K., and Zhong, T. Vec2link: unifying heterogeneous data for social link prediction. In: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM. 2018, 1843–1846.