ON PRIVACY-ENHANCED DISTRIBUTED ANALYTICS IN ONLINE SOCIAL NETWORKS

AIDMAR WAINAKH

Dissertation
Zur Erlangung des akademischen Grades
Doktor der Naturwissenschaften (Dr. rer. nat.)

genehmigte Dissertationsschrift in englischer Sprache
von **Aidmar Wainakh**, MSc.
aus Darmstadt, Germany
geboren in Damaskus, Syrien

Erstreferent: Prof. Dr. Max Mühlhäuser
Korreferent: Prof. Dr. Mathias Fischer

Tag der Einreichung: 7 February 2022
Tag der Prüfung: 21 March 2022

Fachgebiet Telekooperation
Fachbereich Informatik
Technische Universität Darmstadt
Hochschulkennziffer D-17
Darmstadt, 2022

February 7, 2022 — version 1.0

Aidmar Wainakh:
On Privacy-Enhanced Distributed Analytics in Online Social Networks

Darmstadt, Technische Universität Darmstadt
Jahr der Veröffentlichung der Dissertation auf TUprints: 2022
URN: urn:nbn:de:tuda-tuprints-210340
Tag der Prüfung: 21.03.2022

There is only one good in existence, which is knowledge,
and one evil, which is ignorance.

— *Prof. Dr. Mohieddin Wainakh*

Dedicated to those who have sacrificed themselves for the good of humanity

SYNOPSIS

More than half of the world's population benefits from Online Social Networks' (OSNs) services. A considerable part of these services is mainly based on applying analytics on user data to infer their preferences and enrich their experience accordingly. At the same time, user data is monetized by service providers to run their business models. Therefore, providers tend to extensively collect (personal) data about users. However, this data is oftentimes used for various purposes without informed consent of the users. Providers share this data in different forms with third parties (e.g., data brokers). Moreover, user sensitive data was repeatedly a subject of unauthorized access by malicious parties. These issues have demonstrated the insufficient commitment of providers to user privacy, and consequently, raised users' concerns. Despite the emergence of privacy regulations (e.g., GDPR and CCPA), recent studies showed that user personal data collection and sharing sensitive data are still continuously increasing.

A number of privacy-friendly OSNs have been proposed to enhance user privacy by reducing the need for central service providers. However, this improvement in privacy protection usually comes at the cost of losing social connectivity and many analytics-based services of the wide-spread OSNs. This dissertation addresses this issue by first proposing an approach to privacy-friendly OSNs that maintains established social connections. Second, approaches that allow users to collaboratively apply distributed analytics while preserving their privacy are presented. Finally, the dissertation contributes to better assessment and mitigation of the risks associated with distributed analytics. These three research directions are treated through the following six contributions.

CONCEPTUALIZING HYBRID ONLINE SOCIAL NETWORKS

We conceptualize a hybrid approach to privacy-friendly OSNs, *Hybrid Online Social Network (HOSN)*. This approach combines the benefits of using Centralized Online Social Networks (COSNs) and Decentralized Online Social Networks (DOSNs). Users can maintain their social experience in their preferred COSN while being provided with additional means to enhance their privacy. Users can seamlessly post public content or private content that is accessible only by authorized users (friends) beyond the reach of the service providers.

IMPROVING THE TRUSTWORTHINESS OF HOSNS

We conceptualize software features to address users' privacy concerns in OSNs. We prototype these features in our HOSN approach and evaluate their impact on the privacy concerns and the trustworthiness of the approach. Also, we analyze the relationships between four important aspects that influence users' behavior in OSNs: privacy concerns, trust beliefs, risk beliefs, and the willingness to use.

PRIVACY-ENHANCED ASSOCIATION RULE MINING

We present an approach to enable users to apply efficiently privacy-enhanced

association rule mining on distributed data. This approach can be employed in DOSNs and HOSNs to generate recommendations. We leverage a privacy-enhanced distributed graph sampling method to reduce the data required for the mining and lower the communication and computational overhead. Then, we apply a distributed frequent itemset mining algorithm in a privacy-friendly manner.

PRIVACY ENHANCEMENTS ON FEDERATED LEARNING

We identify several privacy-related issues in the emerging distributed machine learning technique, Federated Learning (FL). These issues are mainly due to the centralized nature of this technique. We discuss tackling these issues by applying FL in a hierarchical architecture. The benefits of this approach include a reduction in the centralization of control and the ability to place defense and verification methods more flexibly and efficiently within the hierarchy.

SYSTEMATIC ANALYSIS OF THREATS IN FEDERATED LEARNING

We conduct a critical study of the existing attacks in FL to better understand the actual risk of these attacks under real-world scenarios. First, we structure the literature in this field and show the research foci and gaps. Then, we highlight a number of issues in (1) the assumptions commonly made by researchers and (2) the evaluation practices. Finally, we discuss the implications of these issues on the applicability of the proposed attacks and recommend several remedies.

LABEL LEAKAGE FROM GRADIENTS

We identify a risk of information leakage when sharing gradients in FL. We demonstrate the severity of this risk by proposing a novel attack that extracts the user annotations that describe the data (i.e., ground-truth labels) from gradients. We show the high effectiveness of the attack under different settings such as different datasets and model architectures. We also test several defense mechanisms to mitigate this attack and conclude the effective ones.

ZUSAMMENFASSUNG

Mehr als die Hälfte der Weltbevölkerung nutzt die Dienste der sozialen Online-Netzwerke (OSNs). Ein beträchtlicher Teil dieser Dienste basiert hauptsächlich auf der Analyse von Nutzerdaten. Diese Analysen dienen dazu die Vorlieben der Nutzer zu ermitteln und ihre Erfahrungen entsprechend zu bereichern. Gleichzeitig werden die Nutzerdaten von den Dienstanbietern zu Geld gemacht, um ihre Geschäftsmodelle zu betreiben. Daher neigen die Anbieter dazu, in großem Umfang (persönliche) Daten über die Nutzer. Diese Daten werden jedoch oft für verschiedene Zwecke verwendet ohne dass zuvor die Zustimmung der Nutzer eingeholt wurde. Die Anbieter teilen diese Daten in verschiedenen Formen an Dritte (z. B. an Datenbroker). Außerdem waren sensible Nutzerdaten immer wieder Gegenstand eines unberechtigten Zugriffs durch böswillige Parteien. Diese Vorkommmnisse zeigen, dass das Engagement der der Anbieter, hinsichtlich des Datenschutzes der Nutzer und deren Bedenken, unzureichend ist. Aufkommende Datenschutzbestimmungen (z. B. GDPR und CCPA) sind hauptsächlich dazu gedacht, solche Bedenken zu zerstreuen, Jüngste Studien haben jedoch gezeigt, dass die Erhebung von personenbezogenen Daten und die Weitergabe sensibler Daten weiterhin kontinuierlich zunehmen.

Es wurde eine Reihe von datenschutzfreundlichen OSN vorgeschlagen, um die Bedenken der Nutzer zu zerstreuen, indem sie den Bedarf an zentralen Dienstanbietern verringern. Allerdings führt dieser Verbesserung des Schutzes der Privatsphäre in der Regel zu einem Verlust sozialer Konnektivität und einer Verschlechterung analytischer Dienste der weit verbreiteten OSNs. Diese Dissertation befasst sich mit diesem Problem, indem sie zunächst einen Ansatz für datenschutzfreundliche OSNs vorschlägt, der etablierte soziale Verbindungen aufrechterhält. Zusätzlich werden in dieser Arbeit Ansätze vorgestellt, die es den Nutzern ermöglichen, gemeinsam verteilte Analysen unter Wahrung ihrer Privatsphäre durchzuführen. Schließlich trägt die Dissertation dazu bei, die Risiken, die mit verteilten Analysen verbunden sind, besser einzuschätzen und zu entschärfen. Diese drei Forschungs Richtungen werden in den folgenden sechs Beiträgen behandelt.

KONZEPTUALISIERUNG HYBRIDER SOZIALER ONLINE-NETZWERKE

Wir konzipieren einen hybriden Ansatz für datenschutzfreundliche OSNs. Dieser Ansatz kombiniert die Vorteile der Verwendung von zentralisierte soziale Online-Netzwerke (COSNs) und dezentralisierte soziale Online-Netzwerke (DOSNs). Die Benutzer können ihr soziales Erlebnis in ihrem bevorzugten COSN beibehalten, während ihnen zusätzliche Mittel zur Verbesserung ihrer Privatsphäre zur Verfügung gestellt werden. Die Nutzer können nahtlos öffentliche oder private Inhalte posten, die nur von autorisierten Nutzern (Freunden) außerhalb der Reichweite der Dienstanbieter zugänglich sind.

VERBESSERUNG DER VERTRAUENSWÜRDIGKEIT VON HOSNS

Wir konzipieren Softwarefunktionen, um die Datenschutzbedenken der Benut-

zer in OSNs zu berücksichtigen. Wir prototypisieren diese Funktionen in unserem hybriden OSN-Ansatz und bewerten ihre Auswirkungen auf die Datenschutzbedenken und die Vertrauenswürdigkeit des Ansatzes. Darüber hinaus analysieren wir die Beziehungen zwischen vier wichtigen Aspekten, die das Verhalten der Nutzer in OSNs beeinflussen: Datenschutzbedenken, Vertrauensüberzeugungen, Risikoüberzeugungen und die Bereitschaft zur Nutzung.

DATENSCHUTZ-ERWEITERTE ASSOZIATIONS-REGEL-MINING

Wir stellen einen Ansatz vor, der es Nutzern ermöglicht, effizient datenschutzfreundliche Assoziations-Regel-Mining auf verteilte Daten anzuwenden. Dieser Ansatz kann in dezentralen und hybriden OSNs eingesetzt werden, um Empfehlungen zu generieren. Wir nutzen ein datenschutzfreundliches verteiltes Graphen-Sampling-Verfahren, um die für das Mining benötigten Daten zu reduzieren und den Kommunikations- und Rechenaufwand zu senken. Anschließend wenden wir einen verteilten häufige Artikelgruppe Mining Algorithmus auf eine datenschutzfreundliche Weise an.

DATENSCHUTZVERBESSERUNGEN BEIM FÖDERIERTEN LERNEN

Wir haben mehrere datenschutzbezogene Probleme bei der aufkommenden verteilten maschinellen Lerntechnik, föderiertes Lernen (FL), identifiziert. Diese Probleme sind hauptsächlich auf die zentralisierte Natur dieser Technik zurückzuführen. Wir erörtern die Lösung dieser Probleme durch Anwendung von FL in einer Hierarchiearchitektur. Zu den Vorteilen dieses Ansatzes gehören eine geringere Zentralisierung der Kontrolle und die Möglichkeit, Verteidigungs- und Überprüfungsmethoden flexibler und effizienter innerhalb der Hierarchie zu platzieren.

ANALYSE VON BEDROHUNGEN IM FÖDERIERTEN LERNEN

Wir führen eine kritische Untersuchung der bestehenden Angriffe in FL durch, um das tatsächliche Risiko dieser Angriffe in realen Szenarien besser zu verstehen. Zunächst strukturieren wir die Literatur auf diesem Gebiet und zeigen die Forschungsschwerpunkte und -lücken auf. Dann beleuchten wir eine Reihe von Themen in (1) den Annahmen, die von Forschern üblicherweise gemacht werden und (2) den Bewertungspraktiken. Abschließend diskutieren wir die Auswirkungen dieser Probleme auf die Anwendbarkeit der vorgeschlagenen Angriffe und empfehlen verschiedene Abhilfemaßnahmen.

ETIKETTENLECKAGE DURCH FARBVERLÄUFE

Wir stellen fest, dass bei der gemeinsamen Nutzung von Gradienten in FL das Risiko eines Informationsverlusts besteht. Wir demonstrieren die Schwere dieses Risikos, indem wir einen neuartigen Angriff vorschlagen, der die Nutzerkommentare, die die Daten beschreiben (d.h. die "ground-truth labels"), aus Gradienten extrahiert. Wir zeigen die Wirksamkeit des Angriffs unter verschiedenen Bedingungen, wie z.B. verschiedenen Datensätzen und Modellarchitekturen. Wir testen auch verschiedene Verteidigungsmechanismen, um diesen Angriff zu entschärfen, und kommen zu dem Schluss, dass diese effektiv sind.

PUBLICATIONS

1. Borchert, A., Wainakh, A., Krämer, N., Mühlhäuser, M. & Heisel, M. *Mitigating Privacy Concerns by Developing Trust-related Software Features for a Hybrid Social Media Application* in *ENASE* (2021), 269–280.

2. Borchert, A., Wainakh, A., Krämer, N., Mühlhäuser, M. & Heisel, M. *The Relevance of Privacy Concerns, Trust, and Risk for Hybrid Social Media* in *Communications in Computer and Information Science (CCIS)* (2022).

3. Wainakh, A., Grube, T., Daubert, J., Porth, C. & Mühlhäuser, M. *Tweet beyond the Cage: A Hybrid Solution for the Privacy Dilemma in Online Social Networks* in *2019 IEEE Global Communications Conference (GLOBECOM)* (2019), 1–6.

4. Wainakh, A., Grube, T., Daubert, J. & Mühlhäuser, M. *Efficient privacy-preserving recommendations based on social graphs* in *Proceedings of the 13th ACM Conference on Recommender Systems* (2019), 78–86.

5. Wainakh, A., Guinea, A. S., Grube, T. & Mühlhäuser, M. *Enhancing privacy via hierarchical federated learning* in *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)* (2020), 344–347.

6. Wainakh, A., Müßig, T., Grube, T. & Mühlhäuser, M. *Label leakage from gradients in distributed machine learning* in *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)* (2021), 1–4.

7. Wainakh, A., Strassheim, A., Grube, T., Daubert, J. & Mühlhäuser, M. *Enabling Privacy-Preserving Rule Mining in Decentralized Social Networks* in *The 16th International Conference on Availability, Reliability and Security* (2021), 1–11.

8. Wainakh, A., Ventola, F., Müßig, T., Keim, J., Cordero, C. G., Zimmer, E., Grube, T., Kersting, K. & Mühlhäuser, M. *User-Level Label Leakage from Gradients in Federated Learning* in *Proceedings on Privacy Enhancing Technologies (PETS)* [to appear] (2022).

9. Wainakh, A., Zimmer, E., Subedi, S., Keim, J., Grube, T., Karuppayah, S., Guinea, A. S. & Mühlhäuser, M. *Federated Learning Attacks Revisited: A Critical Discussion of Gaps, Assumptions, and Evaluation Setups* in *IEEE Access* [under review] (2022).

ACKNOWLEDGMENTS

This work would not have been accomplished without the support and encouragement of my advisors, friends, and family. Thank you all for accompanying me on this exciting journey.

First and foremost, I would like to express my sincere gratitude to my supervisor Prof. Dr. Max Mühlhäuser for his continuous support, encouragement, trust, and advice. Thank you for lifting me up every time I walked into your (virtual) office. I cannot remember ever ending a session with you without a smile on my face and refreshing positive energy. I learned from you, not just on scientific topics, but also on the politeness and modesty of scientists. Second, I am thankful to my second supervisor Prof. Dr. Mathias Fischer for the discussions that challenged and inspired me.

Thank you SPINers, my friends! Carlos, Tim, Jörg, Nikos, Leon, Rolf, Ephraim, Andrea, Sarah, Shankar, Manolis, Sheikh, Lamya, and Simon. You guys made this journey possible and also joyful in many of its parts. Thank you for supporting me, showing me the way, listening to my thoughts, and inspiring my work. Thanks to all the members of the TK lab for creating a warm and pleasant work atmosphere.

My family, may God bless you all! Despite the distance, your support and care were crucial for finishing this work. Thank you, my dad for guiding my way by example, my mom for granting infinite tenderness, my brother for enlightening my path with his steps forward. A special warm thanks goes to my beloved sister, who was always beside me bearing my fluctuating mood, listening to my complaints, and encouraging me. To my big family, uncles, aunts, and cousins, thank you for staying in touch and showing me how proud you are of everything I achieve.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

ACRONYMS

OSN	Online Social Network
COSN	Centralized Online Social Network
DOSN	Decentralized Online Social Network
HOSN	Hybrid Online Social Network
GDPR	General Data Protection Regulation
CCPA	California Consumer Privacy Act
API	Application Programming Interface
IPFS	InterPlanetary File System
P2P	Peer-to-Peer
TrustSoFt	Eliciting Trust-Related Software Features
SEM	Structural Equation Model
GIPC	General Information Privacy Concern
IUIPC	Internet Users' Information Privacy Concerns
ARM	Association Rule Mining
FI	Frequent Itemset
RW	Random Walk
ARW	Anonymous Random Walk
MHRW	Metropolis-Hasting Random Walk
MHARW	Metropolis-Hasting Anonymous Random Walk
NDD	Node Degree Distribution
DFP	Distributed FP-Growth
FP-tree	Frequent Pattern Tree
SMS	Systematic Mapping Study
TEE	Trusted Execution Environment
ML	Machine Learning
FL	Federated Learning

HFL	Hierarchical Federated Learning
NN	Neural Network
CNN	Convolutional Neural Network
DNN	Deconvolutional Neural Network
RNN	Recurrent Neural Network
MLP	Multilayer Perceptron
LSTM	Long Short-Term Memory
AE	Autoencoder
SGD	Stochastic Gradient Descent
LR	Logistic Regression
RF	Random Forest
DT	Decision Tree
DLG	Deep Leakage from Gradients
LLG	Label Leakage from Gradients
ASR	Attack Success Rate

Part I

PREFACE

1 INTRODUCTION

Online Social Networks (OSNs) are among the most powerful communication tools in our modern society. Millions of users create profiles, share content, and communicate with friends via OSNs on daily basis. Additionally, a huge number of businesses increasingly employ OSNs to reach customers. That is, the ever-growing user base of OSNs was tripled in the last decade to cover more than half of the world's population [56]. Along with this increase in the user base, data generated by users is massively rising.

The dominant OSNs (e.g., Facebook and Twitter) are commercial; the service providers of OSNs use the data of the users to generate revenue, mainly by realizing targeted advertisements. For that, the user data is processed and analyzed to infer additional information about users. This information is also used to enrich the user experience by customizing the services and recommending interesting content (e.g., an article to read or a friend to connect with). However, since the main revenue of the service providers come from user data, they practice bulk data collection. Beyond what is explicitly posted on OSNs, they track users across the Internet and learn their preferences and behaviors using many different technologies (e.g., third-party cookies).

1.1 PROBLEM STATEMENT

As of today, the mainstream of OSNs is realized in a centralized manner by the platform providers. We refer to these networks as Centralized Online Social Networks (COSNs). That is, all kinds of data shared by users or generated by users' interactions are controlled by a central provider and stored in their infrastructures. Unfortunately, the providers show consistently insufficient commitment to user data and privacy protection [174]. Oftentimes, user data is used without informed consent or misused in various ways [1]. The providers frequently disclose various forms of user data to third parties, e.g., data brokers. Furthermore, user data was prone to unauthorized access on many occasions, e.g., Facebook tokens hack 2018 [98], Twitter readable passwords 2018 [78], and Facebook's leak of user personal data 2021 [116]. Some parties violated the terms of use of OSNs and harvested user data for suspicious purposes, such as Cambridge Analytica 2016 [97]. The privacy of users in COSNs is constantly and seriously threatened or even violated considering the aforementioned issues. This has led to a remarkable increase in the users' privacy concerns [139].

Emerging regulations, e.g., General Data Protection Regulation (GDPR) and California Consumer Privacy Act (CCPA), came to force providers to follow privacy-friendly practices such as obtaining user consent for data collection. However, evidence shows that providers are increasingly collecting data, while obscuring their practices in significantly longer and harder-to-read privacy poli-

cies [272], which leave users no choice but to give their *uninformed* consent. From the technical perspective, several privacy-friendly OSNs were proposed to address the users' privacy concerns, including standalone Decentralized Online Social Networks (DOSNs) (e.g., Diaspora [224]), browser add-ons (e.g., FaceCloak [166]) and apps (e.g., Twitterize [51]). The better-accepted approaches are DOSNs, which are based on Peer-to-Peer (P2P) networks, where not only content (profiles, posts, likes, ...) is encrypted but a centralized authority is eliminated altogether. Unfortunately, most of DOSNs proposals suffer from major limitations, such as poor functionality and support [167], high usage complexity [224], and/or low scalability [51]. Additionally, many users hesitate to use such novel technologies due to lack of trust (a.k.a. penguin effect). More importantly, users became dependent on COSNs due to their established social connections, which can be lost when switching to new networks (a.k.a. lock-in effect). As a result, DOSNs were not able to attract a sufficient number of users to survive the competition with the dominant COSNs. For example, Diaspora, the most popular DOSN, claims 21,227 active users monthly [257] compared with 330 million for Twitter [248] and 2.89 billion for Facebook [247]. That is, there is a need for innovative approaches to address the users' privacy concerns and simultaneously overcome the social lock-in and penguin effects. Based on that, we can formulate our first research question as follows.

RESEARCH QUESTION 1

What OSN approach is better suited to improve privacy while maintaining a large number of users?

One additional drawback of DOSNs is the lower quality of services. Recommender systems are typically used in OSNs to improve the services by recommending interesting content for users. To build a recommender system, a variety of data analytics techniques (data mining and machine learning) can be applied. The application of these techniques to distributed data is called *distributed analytics*, where multiple entities process subsets of data and share collective insights. However, applying distributed analytics in DOSNs while maintaining user privacy is challenging. Some research works proposed cryptography-based solutions, where the user data is encrypted, thus, protected throughout the process. These solutions employ secure multi-party computation [256], homomorphic encryption, and other cryptography primitives [36, 133]. However, despite some improvements, the computational and communication overhead of these approaches remains high (per operation). Also, this overhead remarkably increases with the number of users in the system, which introduces scalability issues and renders these approaches impractical for large-scale applications, such as OSNs. Another category of solutions uses perturbation techniques to protect user data [57]. However, these solutions naturally incur a loss of information, consequently, decreasing the utility of the analysis results. An emerging approach for distributed machine learning, termed Federated Learning (FL) [175], is claimed to offer some privacy advantages by allowing users to train joint models while keeping their data local. However, recent literature has

shown that this approach is prone to information leakage under its default setting, which means user data needs to be protected with additional security and privacy measures. Overall, the three aforementioned solution categories, i.e., cryptography, perturbation, and FL, cannot be readily applied to large OSNs, but need furthering and probably complementary approaches. Based on that, we formulate our second overall research question.

RESEARCH QUESTION 2

How to apply distributed analytics in OSNs privately and efficiently?

Despite the privacy benefits introduced by DOSNs through distributing the user data, the data can still be prone to attacks [229]. One might even argue that the distributed nature of the data extends the attack surface. Applying analytics techniques on distributed data also entails risks [114, 187]. Mitigating these risks is therefore an important research challenge, for which two methodological approaches exist: one possibility is to engage in research on appropriate privacy-preserving techniques directly, the other approach investigates pertinent attacks on privacy first and may lead to the conceptualization of novel attacks. Following that, research can concentrate on techniques for mitigating these attacks. In our work, we adopt the latter approach and formulate the third research question as follows.

RESEARCH QUESTION 3

What are the potential attacks against distributed analytics and how can they be mitigated?

In the next section, we summarize our contributions to address these questions.

1.2 CONTRIBUTIONS & PUBLICATIONS

The contributions of this thesis are divided into three parts to tackle the three aforementioned research questions (see Figure 1.1). These parts discuss hybrid online social networks, privacy-enhanced distributed analytics, and threats to distributed analytics. Next, we present our contributions in each of these parts along with the corresponding publications. In total, the content of this thesis is based on nine papers; eight of them have been published at respected peer-reviewed conferences or journals, and one is under review.

1.2.1 *Hybrid online social networks*

This part focuses on proposing a novel approach to OSNs, where users can access their favorite COSN, thus, communicate with their friends and established communities, and yet acquire additional means for privacy control.

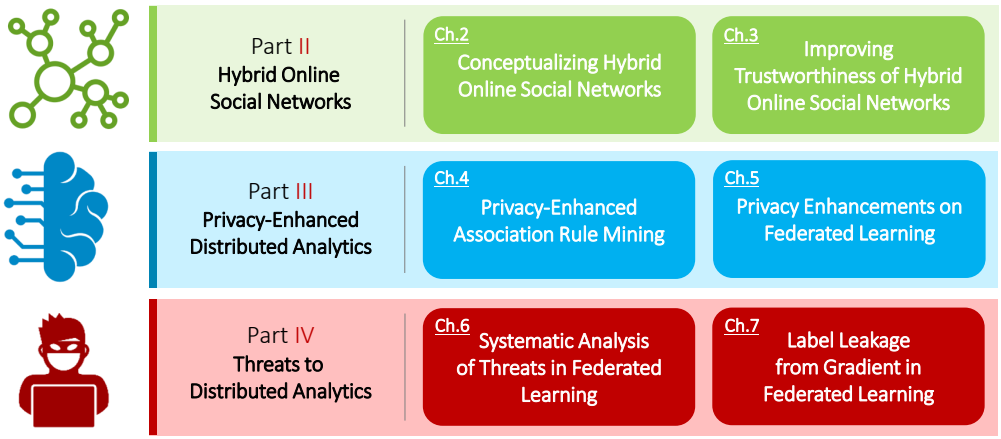


Figure 1.1: Overview of the contributions.

CONCEPTUALIZING HOSNS

Here, we propose the concept of Hybrid Online Social Networks (HOSNs), which combines properties from COSNs and DOSNs. By that users benefit from both the market penetration of COSNs and the privacy advantages of DOSNs. To prove the viability of the concept, we develop Hushtweet, a Twitter application that allows users to privately tweet and give anonymous *likes* on a private network. The data on this network is encrypted and stored in a distributed fashion. Private Tweets are then only accessible by authorized users (followers) beyond the reach of the Twitter provider. Users can seamlessly switch between the usage of Twitter and the private network within Hushtweet. This contribution was published in [273].

IMPROVING THE TRUSTWORTHINESS OF HOSNS

Understanding the user perception for privacy-friendly OSNs in general, and HOSN in particular is crucial to calibrate the development of the concept as well as Hushtweet as an app. That is, we analyze the relationships between four aspects that influence users' perception and behavior: privacy concerns, trust beliefs, risk beliefs, and the willingness to use. Also, we conceptualize and realize software features to address users' privacy concerns reported by Malhotra et al. [171]. We showcase through an extensive user study that the developed features contribute to trust improvement in Hushtweet. This contribution was published in [26, 27].

1.2.2 *Privacy-enhanced distributed analytics*

Here, we tackle the second research question. In particular, we focus on applying two techniques in a distributed and privacy-enhanced manner, namely Association Rule Mining (ARM) and neural networks through FL. We selected ARM as it is one of the simple yet effective analytics techniques, while neural networks are more sophisticated and used widely in state-of-the-art solutions in several domains.

PRIVACY-ENHANCED ASSOCIATION RULE MINING

We present an approach to enable efficient privacy-enhanced ARM on distributed data. In particular, this approach focuses on Frequent Itemset (FI) mining, as it is the most expensive phase of ARM. Our approach combines an adapted version of graph sampling and distributed FI mining techniques to (1) curtail the data required for the mining and (2) reduce the communication and computational overhead. We evaluate our approach on three real-world social networks datasets and show that users can achieve FIs with very high precision and recall rates in well-connected networks, also for very small samples. This contribution was published in [274, 278].

PRIVACY ENHANCEMENTS ON FEDERATED LEARNING

Here, we look into enhancing the privacy of the emerging distributed machine learning technique, FL. FL enables users to train a joint model collaboratively while keeping their data local. They train the model locally and share only the model updates with a central server, which aggregates the updates to obtain an updated global model. In this work, we explore the potential benefits w.r.t. privacy of applying FL in a hierarchical architecture, where the aggregation of the updates happens in multiple layers through the hierarchy. We discuss that this approach can reduce the concentration of power and control in the hands of the central server. Also, the hierarchy allows applying defenses and verification methods in a more flexible and efficient manner. This contribution was published in [276].

1.2.3 *Threats to distributed analytics*

This part is dedicated to answering the third research question. It is comprised of two major contributions as follows.

SYSTEMATIC ANALYSIS OF THREATS IN FEDERATED LEARNING

We conduct a systematic quantitative and qualitative analysis of the publications on attacks against FL. We mainly classify the attacks based on two aspects: (1) the properties of the attacks, (2) the experimental setups. The distribution of publications among the defined classes allows us to derive the foci and gaps in the research literature. We also highlight several issues in assumptions commonly made by researchers, as well as identify their implications on the applicability of the proposed attacks. Finally, we identify multiple fallacies found in the evaluation practices and discuss how these fallacies might impact the generalizability of the results. The paper [280] is based on this contribution and it is under review.

LABEL LEAKAGE FROM GRADIENTS

Here, we highlight the information leakage risk of sharing gradients in FL. We propose a novel attack termed Label Leakage from Gradients (LLG), aims at extracting ground-truth labels (i.e., annotations that describe correctly the characteristics of user data) from gradients. The attack exploits the sign and magnitude of the gradients of the last layer of a neural network to disclose the

ground-truth labels of the training data. We measure the effectiveness of the attack under a variety of setups, including different: datasets, federated training algorithms, model architectures, and model convergence statuses. Finally, we test LLG against two defense mechanisms: noisy gradients and gradient compression (pruning). Results indicate that gradient compression can render the attack ineffective while maintaining good model accuracy. A preliminary set of results of this work was published in [277]. The full work [279] was accepted at PETS 2022.

1.3 BASIC TERMS AND CONCEPTS

In this section, we present a set of important terms and concepts used throughout the rest of the thesis.

1.3.1 *Online social networks*

A *social network* refers to a social structure consisting of multiple entities such as individuals and organizations [66]. Typically, overlapping subsets of these entities have interests, activities, backgrounds, and/or friendships in common. *Online Social Networks (OSNs)* are a special form of virtual social network, where users mostly are able to create profiles, share content (text, images, and videos), and interact with others in various ways [228]. OSNs can be *user-oriented*, where the focus is on the relationships between the users. Thereby, the content is shared within a community of users who are interconnected (e.g., Facebook and Twitter). Other OSNs are *content-oriented*, i.e., the content is shared within a community based on common interests, rather than the relationships between users (e.g., Reddit and Youtube) [201].

1.3.2 *Distributed analytics*

The term *analytics*, as used in this thesis, became common with the growing availability of large amounts of digital data created by sensors (Internet of things), (online) users, and computers (log data in the widest sense). Thereby, analytics denotes data analysis concepts as well as their application to concrete datasets. Often, the goal of systematic analytics is the discovery of patterns and the elicitation of insights on what might happen in the future, mainly to support decision making [60, 204]. There are many techniques for analyzing data with different goals, capabilities, and complexities, ranging from simple statistical operations (e.g., average, histogram) to sophisticated algorithms (e.g., neural networks). In this thesis, we focus on two categories of techniques, namely data mining and machine learning. Data mining refers to the application of specific algorithms to extract patterns from data [76]. Machine learning is the study of algorithms that can produce and automatically improve models by the use of data [181]. The two categories are significantly overlapping as they often employ the same algorithms [29]. *Distributed analytics* refers to the distributed application of these algorithms by multiple entities, with each entity possessing

a subset of the data. When the processing concludes, the outcomes are aggregated to generate collective insights.

1.3.3 *Privacy*

Privacy is a sweeping concept with different definitions varying in social, legal, and computer science domains. In this thesis, we concentrate on the computer science domain, where privacy problems are tackled by developing technical solutions that achieve various privacy properties such as:

- *Confidentiality* is the property of the data being concealed from system entities (e.g., subjects) unless they have been authorized to access it, in other words, maintaining authorization restrictions on data access [226].
- *Anonymity* of a subject is the property of being not identifiable within a set of subjects, known as *anonymity set* [210]. This set comprises the subjects who may be related to a particular anonymous action. A subject performs an action anonymously if they are indistinguishable (by the adversary) from other subjects [59].
- From an adversary's perspective, *unlinkability* of two or more items (subjects or objects) means that the adversary cannot sufficiently distinguish whether or not these items are related to each other [210]. The items can be of the same nature, e.g., subject-subject, or heterogeneous, e.g., subject-object.

Further definitions of privacy properties can be found in [210] by Pfitzmann and Hansen. Our contributions in this thesis aim to address the privacy problems in two contexts: online social networks and analytics.

PRIVACY IN ONLINE SOCIAL NETWORKS. Privacy problems in OSNs reside in the user-user relationship and user-service provider relationship. The social interactions mediated by OSNs' services have disrupted the social boundaries between users, creating the need to renegotiate these boundaries. Service providers extensively collect the personal data of users to run their business model and generate revenue. Users are left without *control* or *awareness* of the process by which service providers collect and analyze their data [99]. In this thesis, we focus on empowering users w.r.t. their relationship with service providers. In particular, we aim at enabling users to actively control and protect their own data instead of relying on a (un)trusted service provider. This can be achieved through approaches that fulfill the aforementioned privacy properties, which in turn allow eliminating or minimizing the personal data disclosure. Further discussion of the properties obtained can be found in the contribution chapters, where appropriate.

PRIVACY IN ANALYTICS. Analytics typically is applied on a sample dataset to produce patterns that represent the distribution of the whole population at hand. Here, we focus on protecting the users whose data was used to apply analytics, i.e., the sample dataset. Privacy in this context is defined such that these

patterns should not reveal (1) whether a particular user was part of the sample dataset or not, i.e., the users should be *unlinkable* to the sample dataset, in other words, the sampled users are *anonymous* within the population, and (2) information about individual users beyond what can be inferred from the underlying distribution [236]. Furthermore, in distributed analytics, the exchange of data between the participating entities should be *confidential* to prevent (1) information leakage to the outside world and (2) disclosure of unintended information to other entities, i.e., to prevent entities from deriving additional information about each other beyond what is necessary for the analytics task.

1.4 OUTLINE

This thesis is structured as follows. After this introduction, the first contribution is presented in Chapter 2, where the concept of HOSN is provided. Chapter 3 focuses on improving the trustworthiness of our social network approach through developing trust-related software features.

Moving on to Chapter 4, we propose a novel approach for applying association rule mining on distributed data in DOSNs or HOSNs. Chapter 5 covers a discussion on the potential privacy advantages of applying federated learning in a hierarchical architecture.

The last two contribution chapters of the thesis are concerned with threats against federated learning. Chapter 6 provides an extensive analysis for the attacks in this field with a critical discussion about the assumptions and evaluation practices. Chapter 7 introduces a novel attack extracting the ground-truth labels of user data from gradients in federated learning. Finally, we close the thesis in Chapter 8 with a conclusion and an outlook.

With our approaches and studies in this thesis, we make a considerable step forward in helping users to take an active role in protecting their privacy in OSNs and applying analytics collaboratively on their protected data.

1.5 NOTES ON STYLE

In the thesis, we use verbatim copies of text from our publications. These verbatim are printed in gray color throughout the thesis. Tables, figures, and algorithms taken from our publications are marked with † in their caption.

Part II

HYBRID ONLINE SOCIAL NETWORKS

2

CONCEPTUALIZING HYBRID ONLINE SOCIAL NETWORKS

This chapter constitutes the first contribution of this thesis. We argued in the previous chapter for the necessity of innovative approaches to privacy-friendly OSNs that empower users to control their data. In this chapter, we introduce a hybrid solution that allows equipping users with means for data control while maintaining the social experience provided by wide-spread OSNs that users are subscribed to. Our concept can also pave the way for users to apply collaborative analytics to their data.

2.1 INTRODUCTION

Despite the recurrent data breaches in Centralized Online Social Networks (COSNs) (e.g., Cambridge Analytica incident [97]), these networks are still dominant with a huge user base. E.g., Facebook reported 2.89 billion monthly active users in 2021 [247], and Twitter had 330 million in 2019 [248]. People use these platforms not only for online self-presentation and social exchange but also to run and promote their businesses. For instance, 200 million businesses—mostly small businesses—use Facebook tools to reach customers [267]. In addition, more than 48% of Business-to-Business decision-makers use Facebook for research [189]. This enables these networks to monopolize the OSN market and impose themselves as unsubstitutable technologies in parts of our societies. Therefore, there is a need for a novel approach that considers the inevitable usage of COSNs and at the same time preserves user privacy.

2.1.1 *Summary of contributions*

In this chapter, we propose the concept of *Hybrid Online Social Network (HOSN)*, which combines advantages from both COSNs and DOSNs. With HOSNs, users keep using COSNs, thus, using their user base and functionality, yet with additional means for privacy control provided by additional concepts inspired from DOSNs.

As a proof of concept, we introduce *Hushtweet*, an Android application that builds on top of Twitter. Users can *tweet* and *like* publicly on Twitter as usual. Additionally, Hushtweet allows users to privately tweet and anonymously *like* on a private network. User private data is then encrypted and stored in distributed databases. Central authorities like Twitter cannot obtain control over the data. Such tweets and *likes* are only accessible by the users' followers, who also use Hushtweet. Moreover, Hushtweet is based on open-source technologies; its architecture and source code are open to the public, resulting in high

transparency for the users. The content of this chapter is based on the following publication.

PUBLICATION

Wainakh, A., Grube, T., Daubert, J., Porth, C., & Mühlhäuser, M. (2019, December). Tweet beyond the Cage: A Hybrid Solution for the Privacy Dilemma in Online Social Networks. In *Global Communications Conference (GLOBECOM) (pp. 1-6). IEEE*.

Contribution statement
In this work, I led the process of idea generation, realization, and writing. Tim Grube and Jörg Daubert contributed helpful comments and observations that inspired my conceptual work and helped to highlight and address a number of issues. Carsten Porth, Rohit Gowda, Pavel Azanov, and Pritish Kumar contributed to the implementation of the prototype application through their (Master or Bachelor) theses. Insightful discussions with Max Mühlhäuser contributed to different aspects of the work.

2.1.2 *Outline*

The remainder of this chapter is organized as follows. First, we define our threat model in Section 2.2. Then, we present related social network approaches in Section 2.3. The concept of HOSN is introduced in Section 2.4, followed by a description of the proof of concept (Hushtweet) for Twitter in Section 2.5. We discuss the requirements met by Hushtweet in Section 2.6. Finally, we conclude in Section 2.7.

2.2 THREAT MODEL

As mentioned in Chapter 1, the existence of omnipotent COSN providers leads to several privacy issues. Recent data breaches demonstrate that the providers cannot be fully trusted to adequately protect user data from external attackers. Also, the misuse of user data for commercial purposes by providers remains a major concern. In this work, we consider the threat posed by a curious COSN provider or an attacker who could penetrate the provider's system and gain access to user data. This adversary has read and write access to all data stored in the COSN system. The adversary has no global view of the network traffic. We assume that the users' devices are not compromised. The adversary's goal is to gather additional information about the individuals who use HOSN, especially about their activities on the private network, e.g., the content they share and their interaction with each other.

2.3 RELATED WORK

In this section, we present three categories of related OSN approaches: (1) Extensions built on top of COSNs for the purpose of privacy, (2) standalone DOSNs, and (3) DOSNs based on decentralized application (dApp).

2.3.1 *Privacy extensions*

Twitterize [51] allows users to share content privately via Twitter itself by encrypting the content and applying concepts for obfuscating the follower-relation. The approach focuses on anonymity and confidentiality as privacy requirements in a group communication settings. Twitterize establishes a P2P overlay "above" Twitter, which allows users to exchange encrypted messages via tweets. The message is passed from a user to another by retweeting until it reaches the targeted recipient. This causes a significant tweet overhead and may violate the terms of use. The restrictions on the Twitter Application Programming Interface (API) (e.g., only 250 direct messages/day are allowed) render the proposed protocol unpractical. FaceCloak [166] protects personal information by sending fake data to Facebook while storing the benign content encrypted on a third-party server. Authorized users can retrieve and decrypt the real content. The approach was implemented as an extension for the Firefox browser, which manipulates locally the Facebook webpages to show the real data instead of the fake ones. Yet, the approach suffers from another potentially failure-prone central server, and continuous Facebook updates make it hard to keep the extension working. Further deviations of these solutions can be found in [206]; the majority of these proposals suffer from severe limitations in performance that impede their usage.

2.3.2 *Standalone DOSNs*

There are several DOSNs proposed in the research community and market. Diaspora [224] is one of the most prominent. It is an open source federated OSN, with a focus on privacy and the users' control over their data. Mostly, groups and organizations host their own servers named *pods*, which interconnect via the Diaspora* federation protocol, with some servers being public. Even though Diaspora multiplied its user base during the Facebook privacy scandals in recent years, Diaspora is still five orders of magnitude behind Facebook. LifeSocial.KOM [93] is a framework for fully P2P-based multimedia-centric social networking. The structured P2P network stores content and protects it with a combination of encryption and an access control list. While a prototype exists, LifeSocial.KOM was never released for public use.

2.3.3 *dApp-based DOSNs*

Decentralized apps run on P2P networks, avoiding the pitfalls of centralization. AKASHA.org uses a combination of blockchain technology (Ethereum Rinkeby)

and a distributed file sharing system, InterPlanetary File System (IPFS). Many of the typical OSN functionalities are available here. AKASHA is switching to the Ethereum *main* blockchain, which will require users to spend tokens (money) to use the network. Peepeth [207] is a micro-blogging platform that promises not to repurpose any user data. Like AKASHA, Peepeth uses the Ethereum blockchain and IPFS. Unlike AKASHA, Peepeth pays the transaction fees (tokens) for its users with capital collected from a crowdfunding campaign. However, as blockchain transactions are signed by the users themselves, a browser with a wallet extension is required. Furthermore, the platform only updates once per hour to lower the transaction fees by executing transactions in batches.

In summary, the aforementioned solutions fall short in the real-world market due to additional costs or lack of usability and features. Peepeth has only reached 4,055 users so far and AKASHA does not disclose statistics about their user base. Only Diaspora is doing reasonably well with 584,197 users [257].

2.4 HOSN CONCEPT

In this section, we first introduce the idea of HOSN. Then, we define its functional and non-functional requirements. We discuss the major challenges to realize HOSNs. Finally, we present our conceptual layout.

2.4.1 *Combining two worlds: COSN and DOSN*

OSNs conduce means of digital life to their users, mostly with a focus on communication. Oftentimes, service providers offer their OSN's functionality without (monetary) costs to their users—they rely on realizing profits from their users' data, e.g., by realizing targeted advertisements. Big COSNs have already attracted a huge number of users. As such, those COSNs dominate the market so that emerging OSNs oftentimes are not able to survive the competition (e.g., Google+). Lacking alternatives, the usage of the *dominant* COSNs becomes inevitable. While being inevitable, COSNs show consistently insufficient commitment to the privacy of their users [143, 144, 264]. Their privacy-friendly alternatives [93, 166, 224], i.e., DOSNs, focus on avoiding the intrusion of their users' privacy; however, these systems are not well-adopted by the users due to several reasons (see Section 2.3).

As mentioned before, the core idea of HOSNs is to combine advantages from COSNs and DOSNs. That combination enables users to leverage both the market penetration of COSNs and the privacy features of DOSNs. HOSN contains a COSN-like part (underlay COSN) and a DOSN-like part (private network) that is established "above" (see Figure 2.1). Using the COSN-like part as base network in HOSN allows users to continue their online social activities smoothly without negative side effects. As such, they do not need to create new profiles nor refind and connect with their friends, which are time-consuming processes and even depend on the availability of the users' contacts and willingness. Thus, continuing to rely on a COSN-like part helps to preserve the users' experience of functionality and social connectivity, and enables a gradual transition to-

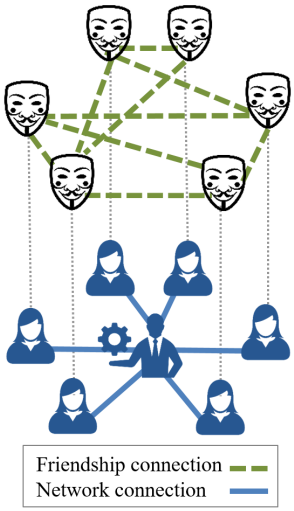


Figure 2.1: Illustration of the HOSN concept, where a decentralized overlay network is established on top of a central network[†].

wards the usage of DOSNs. HOSN provides additionally the ability for users to preserve their privacy by establishing a private network as a complement to the COSN-like part. Using this network, users can communicate without revealing their data to the provider of the COSN-like part. Compared to the individual solutions, i.e., COSNs *or* DOSNs, users can choose between the COSN-like part and DOSN-like part depending on their current interaction and context.

2.4.2 *Functional requirements*

We define the essential functional requirements as follows.

- *COSN-related functionality*: Users can use all the functionality of the underlay COSN app, e.g., managing their own profiles, viewing contact lists, publishing, viewing, and interacting (*like*) with posts. This ensures that HOSN provides all the functionality with which users are familiar and that no additional clients/apps are needed.
- *DOSN-related functionality*:
  - Users can use the communication functionality (posts) in the DOSN-like part to communicate with their friends beyond the reach of the COSN-like part provider.
  - Only authorized users (friends) can view the content of the private posts on the DOSN-like part.
  - Users can *like* a post anonymously on the DOSN-like part.
- Users can freely decide (*control*) whether to interact through the COSN-like part or the DOSN-like part.
- The underlay COSN's provider can obtain privacy-protected (statistical) information from the DOSN-like part about the private posts and *likes*. This requirement is crucial to maintain the possibility to generate "value" out of the social network—targeted advertisements—and thus develop and improve the network.

2.4.3 *Non-functional requirements*

Non-functional requirements, such as privacy and usability, outline the main difference between the HOSN concept and DOSNs/COSNs. In the following, we explain those requirements.

- *Privacy*: HOSNs enable the fulfillment of three privacy-related requirements:
  - *Confidentiality*: Sensitive communication should be concealed such that only authorized users can access its content.
  - *Anonymity*: Neither unauthorized users nor the provider of the underlay COSN are able to link sensitive communication of HOSN to its sender and receivers.
  - *Awareness*: Intuitive interfaces should enable users to understand the privacy implications of their actions and make informed decisions to post to either part of HOSN.
- *Transparency*: The technologies used in HOSN should be announced and users should be able to know how the functionalities of HOSN are realized.
- *Usability*: Setting up and using HOSN should be as simple as possible. Weak usability is a major failure reason for many privacy-preserving/secure applications [111, 290]. All users, especially technically inexperienced users, should be able to use HOSN with minimal effort. There should be no limitations for users who are willing to use the COSN-like part regularly through HOSN. The user interfaces should deliver the same experience as in the underlay COSN with additional privacy control elements.
- *Cost*: As most of the COSNs offer their services free of monetary-charge, HOSN should be available for users for free, e.g., induced by hosting own servers and fees of anonymization services.
- *Compliance*: It is essential that HOSN adheres to the underlay COSN terms of use and policies, otherwise, legal use is not possible.
- *Scalability*: The network should be able to handle a huge number of users, ultimately, the number of users in the underlay COSN. In addition, enough resources (e.g., storage means) should be mobilized to deal with the growing content.
- *Extensibility*: The functionality of HOSNs should be constantly developed responding to users' needs.

2.4.4 *Major challenges*

To realize the DOSN-like part of HOSN, we need to address three technical challenges: storage, access control, and connectivity.

STORAGE. In COSNs, the service providers take care of providing storage for data emerging from users and their communication. Thus, HOSN needs to provide means of storage in the private network part of their service—three different realizations can be used:

- *Centralized Server*: Using a central storage server, the users' data will be stored centrally by one entity in the private network. This solution requires (1) sufficient resources, (2) the server to be trusted by the users, and (3) the need for offering a business model for the storage provider. This solution would essentially reproduce the centralization of control problem of the original COSN with just a "second layer". Additionally, the server will be a potential performance bottleneck and single point of failure. A very limited number of proposals of privacy-friendly OSNs follow this approach, e.g., FaceCloak [167], where neither scalability (thus resources) nor the business model of the storage provider are considered.
- *Hybrid P2P Network*: Using a hybrid P2P approach, powerful users (often called *super peers*) assume the role of storage providers. While the distribution of responsibilities eases the establishment of trust, resource allocation and costs (which underlie the necessity of a business model) are still challenging issues. Several privacy-friendly OSN solutions are based on this approach, e.g., Vis-a-Vis [230] and Polaris [292].
- *P2P Network*: Involving all users in the provision of storage, the costs are minimized as every user has to contribute only a little storage. Also, by further increasing the number of users providing storage, the trust challenge can be addressed under the assumption of the existence of minor benignity. While earlier mentioned challenges are addressed, the classical P2P network challenges emerge: peer discovery, global Internet-based data exchange, and data availability need to be achieved. Multiple privacy-friendly OSNs have adopted this approach, e.g., Safebook [47], LifeSocial.KOM [93], and PeerSoN [28].

ACCESS CONTROL. In COSNs, the provider can implement simple means of access control by realizing *policies*, i.e., users can define groups of users that are able to access a piece of information. HOSN needs to provide similar means of access control. However, as there is no central entity enforcing access policies, access control is a challenge. As pointed out by Paul et al. [206], access control can be realized in DOSNs by access policies, encryption schemes, or a combination of both. To replace the service provider as an access control-enforcing entity, a quorum of "special" users can be enabled to manage access control lists and ensure compliance. These users can be elected by others considering (among other factors) their trustworthiness. The social graph can be leveraged as a base of trust. However, this approach puts a significant load onto these special users resulting in a higher failure probability. A higher failure probability of this access control-enabling users may impede the scalability of HOSN. Another approach is basing access control on data encryption. By sharing keys only with an authorized group of users, users can control the access to their information.

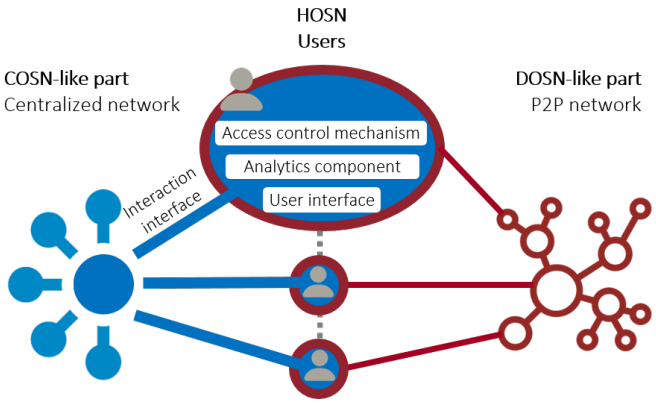


Figure 2.2: Illustration of the conceptual layout of HOSN.

CONNECTIVITY. In COSNs, the service itself provides means of connectivity between users using the central entity. When the central entity is to be avoided for privacy reasons, the DOSN-related functionality must still fulfill the well-known two aspects of connectivity: (1) reachability, i.e., users need to be able to reach each other by finding a path and (2) discovery and addressing, i.e., users need to be able to send and receive messages of interest.

In HOSN, connectivity in the DOSN-like part is more challenging. Two possibilities are addressing the challenge of connectivity differently:

- *Indirect connections*, involving the COSN-like part. Using indirect connections, users protect their privacy by applying necessary obfuscation schemes (e.g., encryption) such that only authorized users are able to read the actual information. The message dissemination itself is established using the COSN-like part; the provider does not learn private information through the obfuscation. Partial unlinkability against the provider of the COSN-like part is only probabilistically achieved as the provider may be able to derive the receiving end of the communication. A variant of this approach is used in Twitterize [51] (see Section 2.3).
- *Direct connections*, bypassing the COSN-like part. Using direct connections, users can combine encryption and anonymous communication techniques to protect their privacy—communication is established by using the DOSN-like part only, the provider of the COSN-like part is not aware of private communication. However, this bypassing causes the dissemination of communication to be challenging w.r.t. user discovery and key exchange.

2.4.5 *HOSN conceptual layout*

To fulfill the aforemetioned requirements, we design the conceptual layout of HOSN to contain four components: a COSN-like part, a DOSN-like part, an analytics component, and a user interface. Next, we describe these components.

COSN-LIKE PART. This part consists of an underlay COSN and an interaction interface that connects the underlay COSN with HOSN. To offer the COSN's functionality through HOSN, HOSN needs adequate means of interaction with the underlay COSN. This interface can be a API or a crawler of the underlay COSN web pages. Upon choosing the underlay COSN, it is essential to consider the effectiveness of such an interface, which will allow users to smoothly use the functionality of the underlay COSN.

DOSN-LIKE PART. In the DOSN-like part, we need to address the three aforementioned challenges, namely storage, access control, and connectivity. For storage, we adopted a P2P network to avoid the control of any central authority over user data. By that, all the HOSN's users equally share responsibilities and privileges for data storage. Yet, it is important to keep in mind the limited resources of the users' devices when it comes to resource allocation.

To realize an access control mechanism, we use end-to-end encryption. The sender of a post encrypts the post and only the authorized receivers can decrypt it. By that, no third parties have access to the content of the private posts. Users encrypt and decrypt the posts locally on their devices. To ensure efficient encryption, we employ a combination of symmetric and asymmetric (public-private keys) encryption schemes (see Section 2.5.4). Upon the usage of HOSN, each user needs to generate a pair of public and private encryption keys. For key exchange, we can leverage the COSN-like part as a communication anchor or use a key server. In both cases, the social graph in the COSN-like part is considered the base of trust.

Encrypted posts are submitted to the P2P network by the sender, where they are stored and retrieved later by the receivers. With this posting and retrieving mechanism, users are interconnected through the P2P network with direct connections that bypass the COSN-like part. The social graph in the COSN-like part remains the base of these connections.

ANALYTICS COMPONENT. As mentioned before, the provider of COSN-like part needs information on users' activity to realize targeted advertisements and run their business model. This information should be collected in a privacy-preserving manner. One essential requirement here is that the information should emerge by aggregating data of multiple users in a way that does not reveal information about individuals. This component is responsible for collecting this information, which can be of different forms. It can be statistical information, e.g., average, minimum, maximum, sum, and count. It can contain the results of data mining algorithms applied on users' data. Machine learning models also can be trained on users' data to be used for different purposes, e.g., to build a recommender system.

USER INTERFACE. The user interface merges the services from both the COSN-like and DOSN-like parts, such that users are enabled to smoothly switch between the different services. The users view a combination of posts fetched from both the COSN-like and DOSN-like parts. The posts on the COSN-like part

are retrieved directly through the interaction interface. For the DOSN-like part, the posts are fetched and decrypted by the receivers from the P2P network.

2.5 PROOF OF CONCEPT

We realized a proof of the HOSN concept, we call it *Hushtweet*. This PoC builds on top of Twitter as an underlay COSN and consists of four main components: Twitter services, P2P data storage, statistics dashboard, and user interface (see Figure 2.3). In the following, we introduce these components and the design decisions underlying the used technologies.

2.5.1 *Twitter services*

- **API.** To allow users to interact with the underlay COSN through HOSN, crawling or APIs can be used. Crawling the web pages of the underlay COSN is hindered by the rapid changes to the pages structure and the fake requests detection mechanisms, which prevent the crawlers from adding data. Additionally, some COSNs, e.g., Facebook, completely prohibit the use of automated crawling methods. On the other hand, most of COSNs provide different kinds of APIs, however, these interfaces vary from COSN to another w.r.t. the available functions and restrictions, e.g., on the number of requests per time interval. Since the APIs are more sustainable and comply with the terms of use of COSNs, we decided to use them as an interaction interface.

- **Twitter.** To select the underlay COSN, we mainly considered two factors: (1) the significance of the COSN and (2) the available functionality in its APIs. Facebook and Twitter are obvious candidates as they are currently among the most dominant COSNs [73, 266]. However, Facebook restricted its API recently after several data breaches (e.g., Cambridge Analytica [96]), which renders it very poor and limited. For example, it is not possible to give a *like* for a post through this API. Unlike Facebook, Twitter provides a variety of rich APIs [268], e.g., standard, premium, and enterprise APIs; the latter two are paid and for high-frequency or mass data access. The APIs are based on HTTP and JSON, and there are several libraries to simplify their usage, e.g., Twit[1] and TwitterKit[2] for Android. Considering the previous discussion, we decided to use Twitter, via its standard API, as an underlay COSN for Hushtweet. Twitter offers through its standard API the widely common functionality of OSNs, such as managing profiles, sharing content (tweeting), retweeting, and liking tweets. The relationship among users is "follow"; when user A follows user B, A is a *follower* of B. When both users A and B follow each other, A and B are *friends*.

1 https://github.com/ttezel/twit
2 https://github.com/twitter/twitter-kit-android

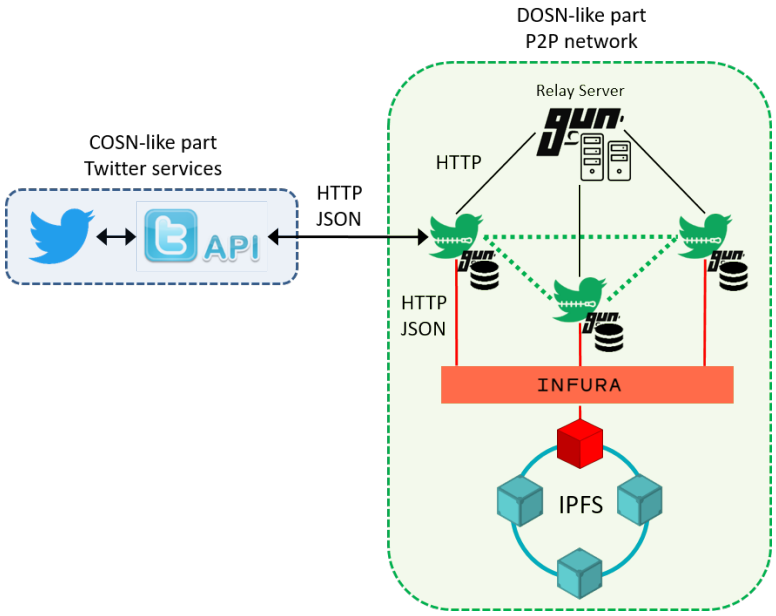


Figure 2.3: Illustration of the software architecture of Hushtweet. The green bird icons refer to Hushtweet users, while the blue bird refers to Twitter. All Hushtweet users are connected through the API with Twitter. For simplicity, only one connection is shown in the figure.

2.5.2 *P2P data storage*

Hushtweet should allow the users to exchange private data avoiding any centralized control. As mentioned before, this is achieved through the DOSN-like part. Next, we present the technologies we used to realize this part.

- **IPFS & GUN.** As discussed in the previous section, we use a P2P network as the basis for the DOSN-like part. There are several technologies that can be used by Hushtweet users to establish a P2P network, such as WebRTC, Yjs, and Hive2Hive. For our application scenario, these technologies exhibit various limitations. For instance, WebRTC and Yjs require servers, and the Hive2Hive library is outdated. In addition, the fact that more than 80% of users access social networks via their smartphones [117, 246] adds more challenges; in particular, the limited battery power of smartphones and their frequent mobility impede constructing a reliable P2P network based solely on Hushtweet users.

  As an alternative, using a P2P network with existing storage and computation capabilities is proposed. For this purpose, three technologies are discussed, namely blockchain, IPFS, and GUN.

  A blockchain is an open distributed ledger. Using blockchains requires the users to spend tokens ("coins"). Obtaining tokens is either costly or computationally expensive. Blockchains with free tokens, e.g., Rinkeby at Ethereum, are only operated on an experimental basis.

InterPlanetary File System (IPFS)[3] is an immutable distributed file sharing system [17]. The goal of IPFS is to connect all computing devices to the same file system. The basis for this is a P2P network where the data is distributed and stored. Unlike other structured P2P networks, IPFS is meant to replace HTTP, which makes it suitable for sharing web content. A cryptographic hash function generates a hash address (content identifier) for each file, which can be used to find the file on the P2P network using a distributed hash table. The maximum size of a file (IPFS object) is 256 KiB; bigger files are split up into multiple objects. While connecting to IPFS requires a rather heavy-weight JavaScript client, services like Infura[4] provide smartphone/app-compatible access to IPFS. Infura is a cloud hosting service that can be used for IPFS free of charge. With such a service not every user has to contribute storage and bandwidth.

GUN[5] is a decentralized database written in JavaScript. It is further developed by the community as an open-source project. Only a browser is required to hold the P2P database shares. The data is stored in a graph structure in the local storage of the browser. Peers synchronize the data among each other over the Internet. For connecting peers, at least one relay server is required.

To realize the storage means of the DOSN-like part, we chose a combination of IPFS and GUN. The private user data, in particular, the private tweets and *likes*, are stored as files in IPFS. To facilitate access to these files, their hash addresses (identifiers) are stored in the GUN database.

- **Access control.** An access control mechanism was implemented based on the OpenPGP standard[6], which is a hybrid encryption scheme, i.e., it is a combination of public-key and symmetric-key cryptosystems. This hybrid encryption scheme is particularly efficient in group communication, where an encrypted tweet is sent simultaneously to multiple receivers through IPFS (see Section 2.5.4). An OpenPGP server is used to store and exchange the public keys of the users. Alternatively, users can exchange their public keys through Twitter directly.

2.5.3 *Statistics dashboard*

In addition to the app, we developed a dashboard connected to the GUN database, where trending topics (hashtags) are listed with the number of tweets for each. This dashboard is the first step towards an analytics component, which needs further development. In Chapters 4 and 5 of this thesis, we elaborate on applying analytics algorithms, namely association rule mining and neural networks in a distributed privacy-enhanced manner, which can be leveraged to enrich this component in Hushtweet.

3 https://ipfs.io/
4 https://infura.io/
5 https://gun.eco, https://github.com/amark/gun
6 https://www.openpgp.org/about/standard

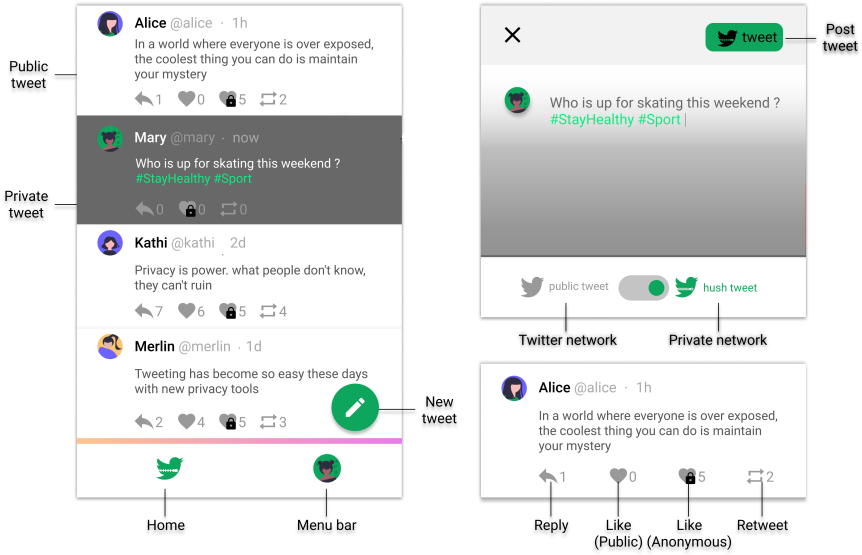


Figure 2.4: Hushtweet timeline and tweet actions control.

2.5.4 *User interface*

We implemented Hushtweet as an Android app using the Ionic framework. The interfaces of this app enable users to interact simultaneously with the services of both Twitter and the P2P network. Figure 2.4 shows some of these interfaces, namely the timeline and tweeting interfaces. In the following, we describe the workflow of two main operations in the app: tweeting and retrieving tweets.

WORKFLOW. Tweeting to Twitter and retrieving the tweets are straightforward processes using the Twitter API. The process of tweeting to the P2P network is illustrated in Figure 2.5 and described as follows.

1 The user chooses whether to tweet to Twitter or privately to the P2P network.

2 The user writes their tweet and submits.

3 The app creates a JSON file, which contains the tweet and additional metadata (e.g., user ID and timestamp).

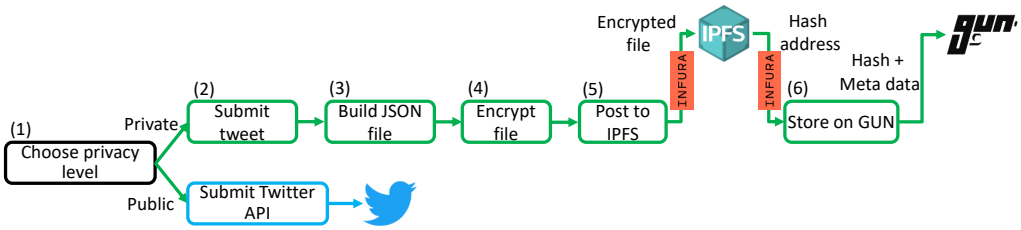


Figure 2.5: Tweeting workflow diagram.

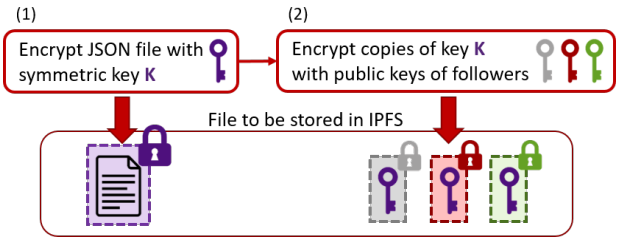


Figure 2.6: Encryption scheme in Hushtweet.

4 The JSON file is symmetrically encrypted; multiple copies of the symmetric key are generated and encrypted with the public keys of the followers, as shown in Figure 2.6.

5 The app posts the encrypted file along with the encrypted copies of the symmetric key to IPFS via Infura.

6 IPFS returns a hash address which is stored on GUN with the user ID hashed by *bcrypt* (a slow-hashing algorithm) with salt and timestamp. The salt is encrypted by the followers public keys and stored in GUN. In addition, the hashtags found in the tweets are stored separately in GUN with timestamps. These hashtags are used later to provide statistical information to Twitter about the trending topics in the P2P network.

On the home timeline, the user can see the tweets of the users they follow chronologically ordered. The tweets are a combination of public ones fetched from Twitter and private ones fetched from the P2P network. In the following, we describe the steps with which the home timeline is produced (see Figure 2.7).

1 Hushtweet obtains public tweets from Twitter via the streaming API.

2 For the P2P network, Hushtweet first collects the list of IDs of followed users via the Twitter API.

3 For each user ID, the hash addresses of new private tweets are fetched from GUN.

4 For each hash address, the corresponding file is loaded from IPFS via Infura.

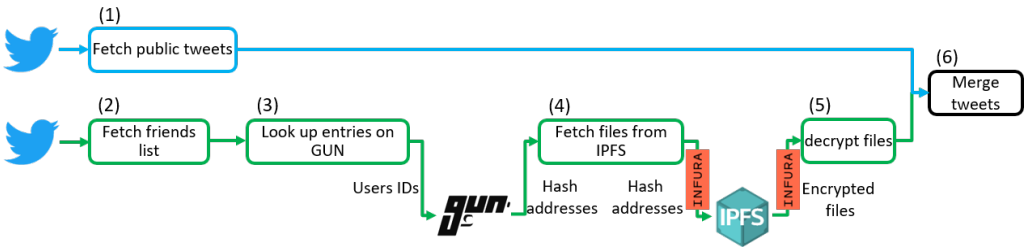


Figure 2.7: Retrieving tweets workflow diagram.

5 For each retrieved file, the app decrypts the symmetric key with the user's private key, then, decrypts the JOSN data with the symmetric key.

6 Finally, the tweet is extracted and displayed in the timeline accordingly.

2.6 DISCUSSION

In this section, we discuss our hybrid solution w.r.t. the privacy and user experience requirements.

2.6.1 *Privacy*

As mentioned in Section 2.2, we consider the curious provider as an adversary; in the case of Hushtweet, it is the Twitter provider. Next, we discuss a number of privacy properties in the light of our threat model.

CONFIDENTIALITY. To preserve authorized access to user data, Hushtweet leverages end-to-end encryption. More precisely, a hybrid encryption scheme (symmetric and asymmetric) is used to secure the private data stored on the distributed network, namely IPFS. By that, we ensure that within the DOSN-like part (1) only authorized users can read private tweets, (2) only authorized users can link private tweets to the author, and (3) the provider cannot read private tweets or anonymous *likes*.

ANONYMITY. In the DOSN-like part of Hushtweet, users communicate with their followers through private tweets. A private tweet posted by a user is fetched and decrypted by their followers—one or more users—and thus group communication scenario is applied. The Twitter provider is aware of the social graph, i.e., the "follow" relationships linking senders and receivers of a tweet. Consequently, if the sender anonymity is compromised by the provider, the receiver anonymity is intuitively violated. Therefore, we focus here on discussing the sender anonymity.

Using the Twitter API requires registering the Hushtweet app and identifying it by a unique token. Thus, the users of Hushtweet are identifiable by the provider. As a result, the maximum anonymity set that can be achieved is the total number of Hushtweet users. Furthermore, Hushtweet users send the token every time they interact with the API. This allows the provider to identify at least a subset of the users who are online at a given time. However, the activity on the DOSN-like part is completely separated from the Twitter API. Assuming that the Twitter provider has no global view of the Internet (i.e., they cannot monitor the network traffic generated by users), submitting a private tweet is unobservable by the passive provider.

Nevertheless, the provider might actively try to detect the senders of private tweets by monitoring the publicly accessible technologies IPFS and GUN. According to Henningsen et al [110], IPFS contains 309,404 nodes. Monitoring the behavior of these nodes to infer information about the sender in Hushtweet is

very challenging for the provider. Surveying IPFS to retrieve private tweets and detect their senders is also infeasible since Hushtweet submits encrypted files with no link to the sender. Moreover, the provider could try to query the GUN database to infer the sender of a private tweet. However, the records on GUN contain the sender ID hashed with salt, which is accessible only by the followers of the sender. The GUN records also contain the timestamps of the private tweets. Combining this information with the active online users of Hushtweet can lead to a reduction in the size of the anonymity set. One mitigation for this issue can be the introduction of a delay in the submission of private tweets.

AWARENESS. The interfaces of Hushtweet were designed to enable the users to easily distinguish between the functionality of Twitter and the private P2P network. Thus, users can be aware of the privacy implications of their actions. We elaborate more in detail on software features to emphasize this property in Chapter 3.

2.6.2 *User experience*

LOCK-IN AND NETWORK EFFECTS. Popular OSNs benefit from the so-called *lock-in effect*. A large number of users have already joined these networks and established their social connections. That is, they are dependent on the networks to maintain their social experience. This leads to better user acquisition and retention, and that in turn increases the *network effect*. This effect refers to the fact that the value of the network (i.e., the positive benefits that users receive) grows with more users joining. In contrast, the lack of success of DOSNs can be attributed to the low user base and, thus, a low incentive for new users to join—the lack of a network effect. Our HOSN ensures that the network effect of the COSN is preserved, i.e., Hustweet delivers the functionality and user experience—including the huge user base and the positive network effect—of Twitter.

PENGUIN EFFECT. Users often refrain from using a new technology as they cannot be sure about its safety and whether it works as expected; instead, users wait for *others* to join first. A big advantage of our approach in this regard is that there is only one "discrete" step necessary for users to use Hushtweet, which is the move to a new client app. This is considered a relatively small effect since all functionality of their favorite COSN (Twitter) remains the same at first and no additional setup is required. With that, users can gradually get used to the additional functionality of Hushtweet; they can start tweeting only very sensitive content to the private network until they feel comfortable.

USABILITY. The user interfaces of Hushtweet contain the main functionality of Twitter, such as viewing the timeline, tweeting, retweeting, and liking tweets as shown in Figure 2.4. The additional privacy control elements are added considering simplicity and clarity for users. Users can choose on a tweet level whether to make it private. They also can anonymously *like* tweets. We

endeavored to minimize the technical knowledge required form users to use Hushtweet. Even though Hushtweet employs new components such as IPFS and GUN, these components are transparent for users.

2.6.3 *Limitations*

Although the Hushtweet app demonstrates the viability of the HOSN concept, some limitations still remain: (1) Twitter enforces a 300 tweets/hour limit via the standard API, usually enforced in 15 min intervals. This prevents excessive tweeting through Hushtweet. In contrast, the regular Twitter app does not have this restriction. (2) Using the Infura cloud service to connect with IPFS takes a considerable overhead off the users' devices. However, Infura is a centralized service and can be a single point of failure. (3) GUN requires a relay server, which is another potential single point of failure regarding the DOSN-like part availability. While the same applies to Twitter itself, Twitter has proven its robustness against failures and attacks. (4) Data exchange via GUN requires the users to be online. The high churn of the users' devices (smartphones) impedes a permanent connection.

2.7 CONCLUSION

COSNs dominate the social network market with a large user base despite their shortcomings w.r.t. user privacy protection. More privacy-friendly OSNs–mainly DOSNs—fail to attract a sufficient number of users either due to lack of usability or the lock-in and penguin effects.

In this chapter, we proposed the concept of *Hybrid Online Social Network (HOSN)*, drawing from a combination of the properties of COSNs and DOSNs. HOSN enables users to control their data by providing them the ability to store their sensitive content on a P2P network, and at the same time, enjoy the regular experience of their favorite COSN. In addition, HOSN contains an analytics component, which paves the way for applying analytics to user data in a privacy-friendly manner. We demonstrated the concept's viability via *Hushtweet*, an Android app that builds on top of Twitter and a distributed file sharing system (IPFS) and a distributed database (GUN). A primitive realization of the analytics component was also provided. Our novel concept showed that it is possible to overcome the lock-in effect of COSNs while enhancing user privacy control through confidentiality and anonymity. This concept also made it more feasible for users to conduct effective analytics (i.e., based on a sufficiently large user population).

Considering the requirements met by HOSN, we argue that the content of this chapter can provide a potential answer to RQ.1: *What OSN approach is better suited to improve privacy while maintaining a large number of users?* However, attracting users relates to many human factors. Two important factors are addressing their *privacy concerns* and gaining their *trust*. The next chapter further investigates (among others) these two aspects of HOSN.

3

IMPROVING THE TRUSTWORTHINESS OF HYBRID ONLINE SOCIAL NETWORKS

The previous chapter introduced the Hybrid Online Social Network (HOSN) concept and Hushtweet as a prototype. This concept came to bridge the gap between centralized and decentralized social networks while focusing on empowering users with means of privacy control. Although the technologies used in implementing Hushtweet fulfill several privacy requirements, gaining user trust is yet another challenge. In this chapter, we address this challenge by conceptualizing software features that are designed to mitigate specific privacy concerns, and consequently, improve the trustworthiness of our application.

3.1 INTRODUCTION

Despite the extensive efforts and advanced technologies used to build privacy-friendly OSNs (e.g., Diaspora [224], Twitterize [51]), they usually fall short of clearly conveying their privacy practices and thus gaining the users' trust. This is due to various reasons, such as introducing novel, and sometimes complex concepts and technologies [273]; users usually hesitate to use a novel technology, as they are not sure that it is safe and works as expected (a.k.a. penguin effect) [42]. Furthermore, since some of these network applications are still under development, their user interfaces are not mature yet, hence they provide a poor user experience. To overcome the above issues and gain user trust, it is important to design user interfaces that provide user-friendly functionalities and insightful explanations of the technologies used to address privacy concerns.

Malhotra et al. [171] found that reducing a set of predefined privacy concerns had a positive effect on users' trust in online companies. This finding can contribute to a potential solution to the lack of trust in privacy-friendly OSNs. Although these networks enhance user privacy through various features, they may not cover all the relevant privacy concerns mentioned by Malhotra et al., which are well established in the research community (see [67, 75, 193]). Thus, conceptualizing software features that are carefully designed to address the relevant privacy concerns could lead to improved trust in OSNs. However, the study of Malhotra et al. was conducted in 2004 [171], even before the existence of modern OSNs, e.g., Facebook and Twitter. The huge number of new technologies introduced in the last decade has shaped society and different user groups more than ever before [61]. Hence, it is likely that various user groups also differ in their cognitions regarding privacy in OSNs. Investigating this aspect is essential for building user-centered OSNs, i.e., tailored to the needs of different user groups. For the aforementioned reasons, it is important to revisit the conclusions of [171] for the new context.

3.1.1 *Summary of contributions*

In this chapter, we conceptualized trust-related features to address predefined users' privacy concerns [171] by applying the software engineering method Eliciting Trust-Related Software Features (TrustSoFt) [25]. These features mainly focus on the graphical user interface. Then, we conducted an extensive user study (with more than 2300 participants) through which:

1 We analyzed, in the context of OSNs, the relationships between the privacy concerns, trust beliefs, *risk beliefs*, and the *willingness to use*. Furthermore, we investigated how these relationships differ for various user groups w.r.t. demographic and privacy-related variables (see Section 3.2.1).

2 We evaluated the impact of the elicited software features on the privacy concerns and the trustworthiness of a use case privacy-friendly OSN application, namely Hushtweet (see Chapter 2).

The content of this chapter is based on the two following publications.

PUBLICATIONS

- Borchert, A., **Wainakh, A.**, Krämer, N., Mühlhäuser, M., & Heisel, M. (2021, April). Mitigating Privacy Concerns by Developing Trust-related Software Features for a Hybrid Social Media Application. In *the 16th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE) (pp. 269-280)*.
- Borchert, A., **Wainakh, A.**, Krämer, N., Mühlhäuser, M., & Heisel, M. (2022). Mitigating Privacy Concerns by Developing Trust-related Software Features for a Hybrid Social Media Application. In *Communications in Computer and Information Science (CCIS). Springer.*

Contribution statement
These papers are the result of a collaborative work, where Angela Borchert and I were the joint first authors and equally contributed the main concepts of the work. I designed the use case application, conceptualized and incorporated the trust-related software features, and developed the user study procedures. Angela Borchert prepared the user study materials, conducted the data collection, analysis, and interpretation. The Master student Girish Sivadanam developed a mockup under my supervision. Nicole Krämer and Maritta Heisel contributed with helpful comments and revised the manuscript. Max Mühlhäuser advised me.

3.1.2 *Outline*

We proceed with the chapter as follows. We start with providing a background on privacy concerns and the TrustSoFt method in Section 3.2. In Section 3.3, we

introduce our hypotheses regarding the relationships between the study's constructs. We apply TrustSoFt and report our procedure in Section 3.4. Then, we present the user study for examining our hypotheses and evaluating the features in Section 3.5. The results are reported in Section 3.6, followed by an analysis and discussion in Section 3.7. Finally, we conclude in Section 3.8.

3.2 BACKGROUND

In this section, we introduce (1) the six privacy concerns defined in [171, 240] and well adopted in the research community (see [67, 75, 193]) and (2) the TrustSoFt method to elicit trust-related software features [25].

3.2.1 *Privacy concerns*

In the last few years, people have been increasingly concerned about their privacy [139]. Malhotra et al. [171] have identified the most prominent privacy concerns as follows.

- *Lacking awareness:* This refers to the degree to which an individual is aware of the privacy practices taken by the organization.
- *Collection:* Users are concerned about the amount of their personal data possessed by the organization. They weigh the cost of disclosing personal data against the benefit of the received services.
- *Insufficient control:* This encompasses whether individuals are able to decide on certain procedures concerning their personal data, e.g., approving, modifying, or opting out.
- *Errors:* This concern stems from the apprehension that the organization might make an insufficient effort to minimize the errors in personal data.
- *Improper access:* This concern focuses on the availability of personal data to people who are not authorized to view or process it.
- *Unauthorized secondary use:* Here, users are concerned that personal data is collected for one purpose but is used for another.

Malhotra et al. [171] examined the relation of *privacy concerns* with trusting beliefs, risk beliefs, and *behavioral intention to disclose personal information*. *Trusting beliefs* indicate the degree to which people believe that an organization is reliable to protect their personal information [94]. *Risk beliefs* are defined as the expectation of a potential loss of personal information when released to an organization [63]. The context of the study was e-commerce. They showed that the greater the privacy concerns, the less people trust online companies, and the greater the perceived risk of data disclosure is. Furthermore, *trusting beliefs* have a positive impact on the behavioral intention to disclose information, while *risk beliefs* affect it negatively. *Trusting beliefs* and *risk beliefs* are also negatively related.

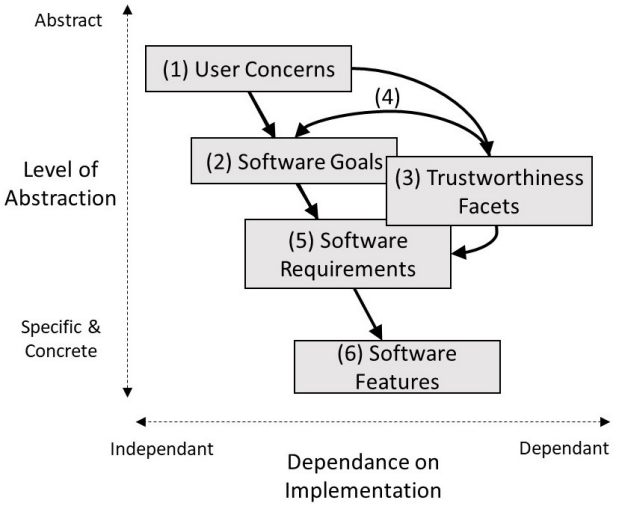


Figure 3.1: Overview of the TrustSoFt workflow, which contains six steps [24]†.

Additionally, Malhotra et al. [171] analyzed the effects of demographic and issue-related variables on the examined constructs. The demographic variables included *gender*, *age*, and *education*. The other variables were *Internet use* in hours per day, the amount of *media exposure* concerning reports of privacy violations, frequency of experienced *privacy invasion*, and occurrence of *ID misrepresentation* in percentage—meaning how often people provide false identification information when asked by a marketer. They found negative correlations between *age* and *behavioral intention*, *education* and trusting beliefs, *Internet use* and trusting beliefs, *ID misrepresentation* and *behavioral intention*, and *media exposure* and trusting beliefs.

3.2.2 *Trust-related software features*

TrustSoFt is a step-wise iterative method to elicit trust-related software features for user-centered OSN applications [25]. TrustSoFt can help to improve the trust of users in (1) the application, (2) the service provider, and (3) other social network users [24]. According to Borchert et al. [24], trust is established when users evaluate whether these parties possess so-called *trustworthiness facets*. Trustworthiness facets describe traits by which the trustworthiness of these parties is assessed. These are, for example, ability, integrity, privacy (in the sense of users' control over their data), reputation, or performance [24, 173, 183]. Applications developed with TrustSoFt shall support users in their trustworthiness assessment. It is assumed that the better a trustworthiness assessment can be carried out, the more likely it is to reduce risks associated with the application use.

TrustSoFt has six major steps as depicted in Figure 3.1: (1) The users' concerns are identified. (2) For each concern, software goals need to be determined. (3) Trustworthiness facets must be specified by considering what quality involved parties should possess so that a concern is reduced. (4) Trustworthiness facets are then related to a software goal. (5) Afterwards, to achieve the software goals and to address the related facets, the software requirements are

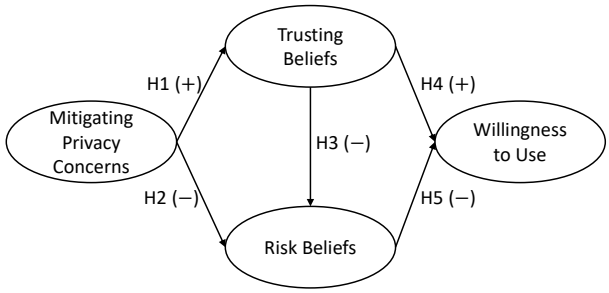


Figure 3.2: Overview of Hypotheses H1-H5†.

defined. (6) Lastly, software features describe in what way a requirement can be implemented. Usually, features are specific front- or backend elements.

3.3 HYPOTHESES

Understanding the relationships between *privacy concerns*, *trusting beliefs*, and *risk beliefs* is crucial to inform the development of privacy-friendly OSN applications. In this work, we revisit these relationships in the light of the work of Malhotra et al. [171]. As we propose software features to address privacy concerns, we focus on analyzing how mitigating the concerns impacts trusting and *risk beliefs*. Furthermore, we investigate how these constructs relate to the *willingness to use*, which is also an essential ingredient to develop acceptable applications. This results in the model shown in Figure 3.2 and the following hypotheses.

- H1: Mitigating privacy concerns (MP) has a positive effect on *trusting beliefs* (T).
- H2: Mitigating privacy concerns (MP) has a negative effect on *risk beliefs* (R).
- H3: *Trusting beliefs* (T) have a negative effect on *risk beliefs* (R).
- H4: *Trusting beliefs* (T) have a positive effect on the *willingness to use* (W).
- H5: *Risk beliefs* (R) have a negative effect on the *willingness to use* (W).

3.4 APPLYING TRUSTSOFT

We applied the TrustSoFt method to elicit software features that mitigate privacy concerns in the HOSN application, Hushtweet. Our procedure is explained below and illustrated through the concern *errors*, as an example.

IDENTIFY USER CONCERNS. Considering former research [171, 240], we focus on privacy concerns (see Section 3.2.1). We elicited features for each concern separately. As a first step, we revisited the definition of each concern (according to [171, 240]) and made ourselves aware of their identifiable characteristics and descriptive keywords. For the *errors* concern, the keywords are *errors in personal data*, *deliberate and accidental errors*, and *minimizing problems*.

SET SOFTWARE GOALS. Based on the concern definition, we derived a set of software goals that mitigate this concern, thus improving the overall satisfaction of the end-users. For instance, to address the *errors* concern, we need to ensure that the data stored by Hushtweet is accurate and error-free. Therefore, we identified *data accuracy* as a goal.

SPECIFY TRUSTWORTHINESS FACETS. In order to support users in their trustworthiness assessment, we specified a number of trustworthiness facets, which are then allocated to goals. We distinguished who exactly is involved in the concern as stakeholders. Then, we consulted the literature to determine what characteristics are desired by these stakeholders to avoid or reduce the concern. For the *errors* concern, we identified four facets for the Hushtweet application: *data integrity*, *data reliability*, *data validity*, and *failure tolerance* [183]. We assigned the facets to the goal of *data accuracy*.

SET TRUSTWORTHINESS REQUIREMENTS. Next, we defined software requirements by describing what the system should do to achieve the software goals and meet the selected facets. Oftentimes, one requirement might address multiple facets simultaneously. For example, we defined the requirement: *Verifying the correctness of the data*, to meet the facets *data integrity*, *data reliability*, and *data validity*.

DERIVE SOFTWARE FEATURES. Lastly, we specified how to realize the requirements through a set of software features. For the evaluation in the later user study, we focused on features for the user interface of Hushtweet rather than the backend system. We elicited two features to realize the aforementioned requirement: (1) An alert message on tweeting privately says: "Data is correctly and safely stored". (2) Two questions in the FAQ section: "How does Hushtweet ensure the correctness and integrity of my data?" and "Does Hushtweet modify my data?".

Applying TrustSoFt for Hushtweet resulted in a long list of software features. In Appendix A.1, we list an extract of the identified features, which were implemented in the Hushtweet mockup for the user study.

3.5 USER STUDY

To test our hypotheses (see Section 3.3) and study the relationships between the defined constructs, we conducted an extensive online user study. The structure of the study is explained below.

3.5.1 *Experimental design*

MOCKUPS. We developed eight mockups of the Hushtweet application using the online design tool Figma[1]. First, we created a basic mockup that contains only the basic functionalities of Hushtweet (see Chapter 2). Then, six mockups

1 https://www.figma.com

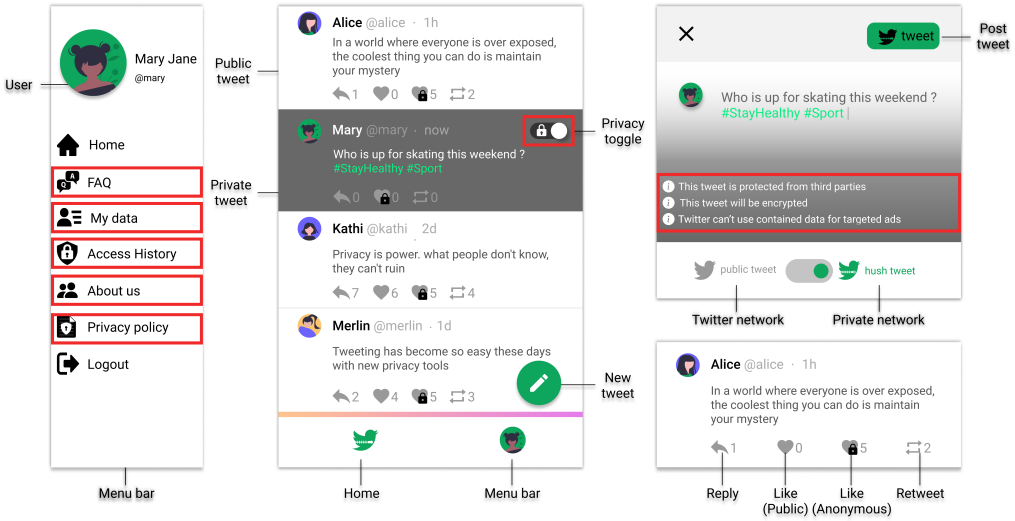


Figure 3.3: Overview of the Hushtweet mockup for the Full-featured group. The red frames highlight included software features†.

were developed and each one was extended by three software features to address one particular privacy concern. We carefully selected the implemented features to cover all trustworthiness facets identified while applying TrustSoFt. Finally, we built a full-featured mockup that includes all the elicited features, which are described in detail in Appendix A.1. An overview of this mockup is shown in Figure 3.3.

PARTICIPANTS. The online user study followed a between-group design with nine experimental groups of participants, who were recruited through Amazon Mechanical Turk[2]. The first group was termed Concept, which was introduced only to explanatory materials on the concept and basic functionality of Hushtweet. Each of the remaining eight groups additionally interacted with one of the aforementioned mockups. More details on the procedure follow in Section 3.5.2.

SCALES. We used six questionnaires listed in Table 3.1; we mainly adopted the scales found in the work of Malhotra et al. [171], namely General Information Privacy Concern (GIPC) [240], Internet Users' Information Privacy Concerns (IUIPC), trusting and *risk beliefs* [125]. We also included questions on demographic (*gender*, *age*, and *education*) and privacy-related variables, namely (1) *ID misrepresentation*: how often individuals provide falsified personal identifiable information online, and (2) *privacy invasion*: how often the privacy of subjects has been invaded in the past. Moreover, we developed an eight-questions scale to measure the *willingness to use* Hushtweet. For each questionnaire, we used a 7-point Likert scale (1="strongly disagree" to 7="strongly agree").

The questionnaires were adapted in the wording to the Hushtweet context. As an example, we replaced words like "online companies" and "computer databases" with "Hushtweet" and "distributed databases".

2 https://www.mturk.com

Scale	Subscale
General Information Privacy Concern (GIPC) [171, 240]	-
Internet Users' Information Privacy Concerns (IUIPC) [171]	Collection, insufficient control, lacking awareness, errors, improper access, unauthorized secondary use
Trusting beliefs [125, 171]	-
Risk beliefs [125, 171]	-
Willingness to use	-

Table 3.1: Overview of the used scales.

The IUIPC scale was used for two purposes: first, to measure the privacy concerns of the Concept group. Second, for the rest of the groups, the scales were used to measure to what extent the Hushtweet mockup addresses the privacy concerns. For that, the scales were modified by omitting the expectational modal verb "should". All the used questionnaires can be found in Appendix A.2.

3.5.2 *Procedure*

The procedure of the study is illustrated in Figure 3.4 and described as follows.

1 *Introduction and general concerns*: The participants were briefed about the context of the study. Also, they completed the GIPC questionnaire.

2 *Hushtweet description*: They were introduced to the concept and basic functionalities of Hushtweet.

3 *Comprehension test*: We checked the participants' comprehension of the functionality of Hushtweet with six questions. The purpose of this check was to include only the participants who understood the concept of Hushtweet for the follow-up analysis. This helped to avoid potentially misleading results based on a misunderstanding of the concept.

4 *Interaction*: Each experimental group—except the Concept group—was given a distinct task to perform using a Hushtweet mockup. The task contained hints about the features added to address the corresponding privacy concern in the particular group. Overall, the tasks of the different groups were kept comparable in terms of complexity and required number of clicks. Each participant had a minimum of five minutes to interact with the mockup. To keep results comparable, the actual interaction time of each participant was considered in the subsequent analysis; the participants with anomalous timings were excluded from the final results.

5 *Questionnaires*: All groups received the remaining scales in the following order: IUIPC, *trusting beliefs* scale, the scales for *risk beliefs* and the *willingness to use*. Finally, the participants were asked demographic and privacy-related questions described in Section 3.5.1.

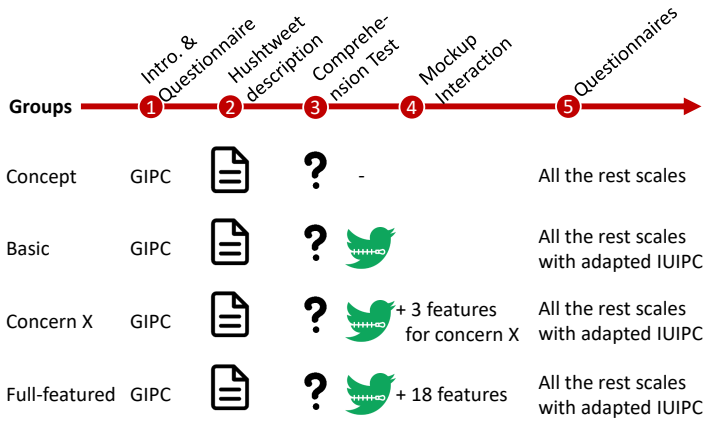


Figure 3.4: Overview of the study procedure.

3.6 STUDY RESULTS

In this section, we report details on our results w.r.t. (1) the population of the participants, (2) descriptive analysis of the constructs and the relevance of privacy concerns, (3) our hypotheses, (4) the impact of user variables on the hypotheses, and (5) the impact of the developed features on the privacy concerns.

3.6.1 *Population*

We recruited only experienced Amazon Mechanical Turk users to ensure high-quality data. Participants were only allowed to take part in one of the experimental groups. Each group contained between 250 and 300 participants. For further analysis, we considered participants, who absolved the Hushtweet comprehension test with more than 50% correct answers. Furthermore, only complete data sets were analyzed. This reduced the various populations by 7% to 19% across different groups. The final population of each experimental group is shown in Table 3.2 along with demographic information.

Group	Filtered Population	Male%	Female%	Age (M)	Education% $\geqslant$ Bachelor
Concept	245	61.2	38.4	35.4	52.7
Basic	205	68.3	31.2	33.6	84.5
C1: Lacking awareness	222	63.1	36.0	33.5	67.1
C2: Collection	223	63.2	36.3	35.6	66.9
C3: Insufficient control	223	58.3	39.5	37.6	70.8
C4: Error	211	58.7	39.8	32.6	87.7
C5: Improper access	202	58.4	41.6	35.9	72.8
C6: Unauthorized secondary use	216	64.8	35.2	33.8	83.8
Full-featured	233	63.9	35.2	35.6	68.3

Table 3.2: Overview of the experimental groups and their demographics[†].

Construct	M	SD	Subconstruct	M	SD
GIPC	4.89	0.93	-	-	-
IUIPC	5.73	0.74	Unauthorized secondary use	6.26	0.93
			Lacking awareness	6.16	0.84
			Improper access	5.89	1.03
			Insufficient control	5.87	0.86
			Errors	5.14	1.30
			Collection	5.04	1.17
Trusting beliefs	5.14	1.08	-	-	-
Rrisk beliefs	3.58	0.94	-	-	-

Table 3.3: Evaluation of the constructs in the Concept group.

The average (across groups) gender distribution of 62.3% males and 32.8% females resembles a similar distribution to the actual population of Twitter users in 2021 with 68.5% males and 31.5% females [195]. The age of our participants varies from 18 to 73 years. More than 44% of them are in the 25-34 range. This is relatively close to the age distribution of Twitter users, 38% of whom are in this range [194]. In terms of education level, 33% of Twitter users have a Bachelor's degree or higher [197], while this percentage is higher among our participants, with an average of 72.7%.

3.6.2 *Descriptive analysis of the studied constructs*

Here, we investigate the relevance of users' privacy concerns in Hushtweet. For that, we conducted a descriptive analysis for the results of the Concept group, as this group did not interact with any mockup and focused only on the principles of Hushtweet. The mean (M) and standard deviation (SD) of the studied constructs are shown in Table 3.3. The general privacy concerns GIPC and privacy concerns IUIPC are found to be moderately related ($r = 0.561$, $p < 0.001$). The participants rated the importance of the individual concerns in the following order (from highest to lowest): (1) *unauthorized secondary use*, (2) *lacking awareness*, (3) *improper access*, (4) *insufficient control*, (5) *errors*, and (6) *collection*.

Overall, the participants showed moderated general privacy concerns with high variance, whereas, they conveyed that Hushtweet should address individual privacy concerns. The participants trusted Hushtweet and slightly disagree that it is risky.

3.6.3 *Hypotheses H1-H5*

We tested Hypotheses H1-H5 using a Structural Equation Model (SEM) (see Figure 3.2) for the Basic and Full-featured groups, since these groups represent the two boundary cases, with and without the developed features. Considering groups where the privacy concerns are only partially addressed might yield biased results, therefore, we did not include other groups in this analysis. We omitted the items (questions) that did not contribute to an acceptable *internal*

Hypothesis	Basic group		Full-featured group	
	Relationship Coefficient	Confirmed	Relationship Coefficient	Confirmed
H1: MP $\xrightarrow{+}$ T	0.843***	✓	0.862***	✓
H2: MP $\xrightarrow{-}$ R	0.042	✗	0.092	✗
H3: T $\xrightarrow{-}$ R	0.189	✗	0.410**	✓
H4: T $\xrightarrow{+}$ W	0.756***	✓	0.706***	✓
H5: R $\xrightarrow{-}$ W	0.208**	✗	0.076	✗

***. Correlation is significant at $p \leqslant 0.001$, **. at $p \leqslant 0.01$, and *. at $p \leqslant 0.05$.

Table 3.4: Results of the relationships between the constructs.

scale consistency of at least $\alpha = 0.7$. Also, we excluded constructs with *factor loadings* < 0.7. As a result, the privacy concerns *collection* and *errors* were excluded from the models. Furthermore, we checked the model fit of SEMs by calculating a confirmatory factor analysis [121]. The SEMs are at least acceptable with a *comparative fit index* and *Tucker-Lewis index* > 0.9, a *root-mean-square error* of approximation < 0.8 and a *normed chi-square* $(X^2/df) < 5$.

Our results are summarized in Table 3.4. We found that the mitigation of privacy concerns positively affects *trusting beliefs* (H1) in a strong way. Therefore, Hypothesis H1 is confirmed. The relationship between *mitigated privacy concerns* and *risk beliefs* was not statistically significant for any experimental group. Thus, we cannot confirm Hypothesis H2. Hypothesis H3 is significant only in the Full-featured group. Therefore, a negative impact from *trusting beliefs* on *risk beliefs* can only be partially supported. Furthermore, the results confirmed H4 where we found that *trusting beliefs* positively and strongly influence the *willingness to use*. With regard to H5, *Risk beliefs* do not significantly influence the *willingness to use* in the Full-featured group. However, in the Basic group, the influence is statistically significant, and it is positive with a weak effect ($r = 0.208$, $p = 0.001$), therefore, Hypothesis H5 can be partly falsified. A discussion of these results follows in Section 3.7.2.

3.6.4 *Moderation analysis for demographic and privacy-related user variables*

The different characteristics of users might have a moderation impact on the relationships between the constructs. Analyzing that impact may result in insights for future Hushtweet development considering specific groups of users. In this section, we study the impact of (1) demographic variables: *gender*, *age*, and *education*, and (2) privacy-related variables: *ID misrepresentation* and *privacy invasion*. For that, we conducted an exploratory moderation analysis for the Full-Featured group, where the participants used a Hushtweet mockup with all the developed features to address their privacy concerns.

For moderation analysis, we used the PROCESS procedure in the SPSS[3] software with standardized variables [106]. Dummy coding was used for the cat-

3 https://www.ibm.com/analytics/spss-statistics-software

egorical variables in our analysis [131]. We chose the following reference variables for the dummy coding method: "Never" for *ID misrepresentation*, "highschool" for *education*. Some categories of ordinal variables could not be considered representative because of the very small number of associated participants. Therefore, we excluded "some school, no degree" and "doctoral degree" from *education*. To omit extreme outliers, we used boxplots [68]. After the moderation analysis, we conducted simple slope analyses to examine the interaction effects [4]. Next, we report the results per variable.

As shown in Table 3.5, the *age* had a moderating effect on the predictions of (1) *mitigated privacy concerns* on *trusting beliefs*, (2) *mitigated privacy concerns* on *risk beliefs*, and (3) *trusting beliefs* on the *willingness to use*.

For *education*, a moderating effect could be observed regarding *mitigated privacy concerns* on *risk beliefs*. The interaction effect was found for people having a Master's degree compared to those with a highschool graduation.

ID misrepresentation was found to moderate relationships between constructs for certain value categories. When it is over 75%, *ID misrepresentation* moderates the relationship between *mitigated privacy concerns* and *trusting beliefs*. For *mitigated privacy concerns* and *risk beliefs*, interaction effects with *mitigated privacy concerns* were found for the categories 26%-50% and 51%-75%. For the moderation with *trusting beliefs* and *risk beliefs*, the interaction effect with *trusting beliefs* was significant for the categories 26%-50% and 51%-75%. The last moderation of *ID misrepresentation* was found for *trusting beliefs* and the *willingness to use*. The categories 26%-50% and over 75% significantly interacted with *trusting beliefs* for the prediction of *willingness to use*.

Hypothesis with User Variables	**F**	r^2%	Categories	t	β
H1: MP $\xrightarrow{\text{age}}$ T	133.541	63.73***	-	2.40*	0.09
H2: MP $\xrightarrow{\text{age}}$ R	6.176	7.58***	-	−2.26*	−0.13
H4: T $\xrightarrow{\text{age}}$ W	53.415	41.27***	-	−2.63**	−0.13
H2: MP $\xrightarrow{\text{edu}}$ R	3.640	13.06***	-	2.02*	0.144
H1: MP $\xrightarrow{\text{id}}$ T	50.684	67.56***	> 75%	3.09**	1.37
H2: MP $\xrightarrow{\text{id}}$ R	4.338	15.05***	26 − 50%	2.62**	0.396
			51 − 75%	3.08**	0.778
H3: T $\xrightarrow{\text{id}}$ R	6.007	19.80***	26 − 50%	2.42*	0.386
			51 − 75%	2.09*	0.534
H4: T $\xrightarrow{\text{id}}$ W	20.968	46.29***	26 − 50%	2.32*	0.302
			> 75%	−2.04*	−0.501
H2: MP $\xrightarrow{\text{inv}}$ R	23.812	23.86***	-	4.83***	0.315
H3: T $\xrightarrow{\text{inv}}$ R	29.362	27.87***	-	4.42***	0.27
H5: R $\xrightarrow{\text{inv}}$ W	4.575	5.68**	-	2.27*	0.159

***. Correlation is significant at $p \leqslant 0.001$, **. at $p \leqslant 0.01$, and *. at $p \leqslant 0.05$.

Table 3.5: User variables in the Full-featured group. Here, four variables are illustrated: *age*, *education* (edu), *ID misrepresentation* (id), and *privacy invasion* (inv).

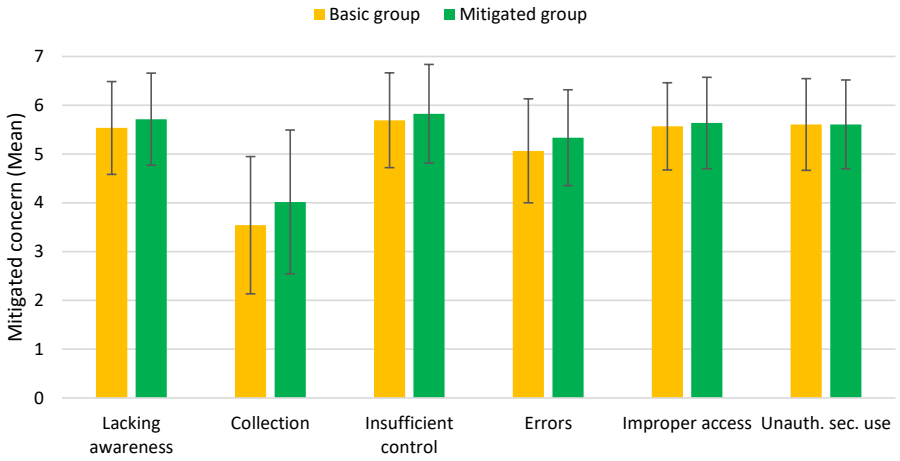


Figure 3.5: Mean scores of *mitigated privacy concerns* based on the IUIPC scale for the Basic group and the groups where each concern is mitigated.

For privacy invasion, we found moderations for: (1) *mitigated privacy concerns* and *risk beliefs*, (2) *trusting beliefs* and *risk beliefs*, and (3) *risk beliefs* and the *willingness to use*.

3.6.5 *Impact of developed features*

As mentioned above, there are six mockups, each of which contains three features implemented to mitigate a particular concern. Through the IUIPC scale, we measured how much each concern was mitigated in all groups and mockups, i.e., the higher the score, the more the participants receive the concern as mitigated. To evaluate the implications of the developed features on the privacy concerns, we compare the score of the mitigated privacy concern in the corresponding group with the Basic group.

The results are shown in Figure 3.5. All the score differences were tested and found to be statistically significant. We can notice that participants of the mitigated groups rated the corresponding concern as more mitigated. However, the mean differences are very limited ranging between 0.002 and 0.47. Also, we notice the large standard deviations of the scores $0.8 < SD < 1.5$, which indicates that the participants vary in their perception of the concerns mitigation.

Interestingly, we observe also cross-concern impacts of some features. In particular, the features added to mitigate the *lacking awareness* concern led to an increase in the feeling of control from 5.69 in the Basic group to 5.86 in the *lacking awareness* group. In contrast, the *errors* group that was confronted with simulated errors and the features to handle them felt less in control with a mean score of 4.82.

3.7 DISCUSSION

In this section, we discuss the results of our user study on (1) the relevance of the privacy concerns, (2) the relationships between the constructs, (3) the

impact of user variables on these relations, and (4) the impact of the developed features on privacy concerns. Lastly, we conclude the section by describing the limitations of this work.

3.7.1 *Relevance of privacy concerns*

Our results in Table 3.3 show that *unauthorized secondary use* is the most important concern, followed by *lacking awareness* and *improper access*, while *errors* and *collection* were the least relevant. These findings are aligned with the results of Smith et al. [240], where they found that *unauthorized secondary use* and *improper access* affect privacy concerns more than *errors* and *collection*. The prominence of *lacking awareness* can be due to the recent recurrent data breaches and intransparency of OSNs, which made users more eager to know the privacy practices allegedly applied by service providers.

3.7.2 *Relationships of the constructs*

Our results show that addressing privacy concerns by software features does have a positive impact on the *trusting beliefs* in Hushtweet. This indicates that the developed features successfully reflect the benevolent purpose of Hushtweet. In a broader sense, we conclude that addressing privacy concerns increases trust in the context of HOSN.

For *risk beliefs*, we found that addressing privacy concerns does not necessarily reduce the *risk beliefs*. Olivero et al. [199] pointed out that risk awareness increases the demand for control, which is one of the *mitigated privacy concerns*. However, addressing this concern seems to have a limited impact on the overall *risk beliefs* in our context. A possible explanation for this might be that users are still aware of the existing risks accompanying their data processing during the usage of social networks, especially since the existence of these risks is highlighted by the implemented software features.

Interestingly, we observe that *trusting beliefs* reduce *risk beliefs* only when users interact with the application where all privacy concerns are addressed. We conclude that having a mature application that addresses a multitude of privacy concerns is essential to establish the relationship between *trusting* and *risk beliefs*.

Furthermore, results reveal that *trusting beliefs* positively impact the *willingness to use*. This conforms with the conclusions of Malhotra et al. [171]. Another salient finding is that sometimes users are a bit more willing to use Hushtweet the higher their *risk beliefs* are. One explanation can be that users may prefer risky products under conditions of curiosity, variety seeking, or boredom [62]. These conditions induce users to tolerate more risk and thus promote the *willingness to use*. However, a positive relation of *risk beliefs* to *willingness to use* is not always confirmed.

3.7.3 *Impact of user variables*

We found that the user demographic variables, *age*, and *education*, have an impact on the relationships between the studied constructs, while no effect of *gender* was observed. Furthermore, the privacy-related variables, *ID misrepresentation*, and *privacy invasion*, also moderate these relationships. Next, we discuss the impact of each variable.

3.7.3.1 *Demographics*

AGE. Although older users might be more cautious concerning their privacy, due to their longer experience with information technology [90], they were easier to convince by the developed software features to increase their *trusting beliefs* and reduce *risk beliefs*. In addition, the negative relationship between *trusting* and *risk beliefs* was found to be more prominent for older users. Among younger users, on the other hand, the positive effect of *trusting beliefs* on *willingness to use* is more strongly maintained than among older users.

EDUCATION. Users with high-school graduation believed Hushtweet to be less risky the more their privacy concerns were addressed. This is slightly the opposite for users with a Master's degree. One explanation could be that users with a higher level of education are more aware of privacy risks [216], and the implemented software features sensitized them even more to these risks making them more cautious.

3.7.3.2 *Privacy-related variables*

IDENTIFICATION MISREPRESENTATION. Individuals who differed in the frequency of *ID misrepresentation* showed different expressions concerning the constructs studied. With respect to the *mitigated privacy concerns* and *trusting beliefs*, the former positively predicts the latter for both individuals who never misreported their identity and those who misreported it very often.

Users, who never misrepresented their IDs believed Hushtweet to be less risky the more their privacy concerns were mitigated. This is in contrast to those who often intentionally disclosed false IDs. With higher *mitigated privacy concerns*, their *risk beliefs* slightly increased. We assume that users who more often misrepresent their personal information have generally a higher risk awareness. As the selected Hushtweet software features emphasized the associated risks that are aimed to be reduced, these users might be strengthened in their cautiousness.

Trusting beliefs similarly affected *risk beliefs* in a negative way. The *trusting beliefs* of users who sometimes falsified their IDs had a smaller impact on their *risk beliefs* than that of individuals who never misrepresented their IDs.

Lastly, predicting *willingness to use* based on *trusting beliefs* is positively maintained for all the categories of users w.r.t. *ID misrepresentation*. However, *willingness to use* is found to be exceptionally high for users who often misrepresented

their IDs. We again relate this to our assumption that this set of users are highly aware of their privacy, and they use *ID misrepresentation* as a privacy protection strategy [129]. Therefore, Hushtweet might be especially appealing to them due to its privacy-friendly characteristics.

PRIVACY INVASION. With increasing experiences of privacy invasions, the following impacts became weaker: (1) *mitigated privacy concerns* on *risk beliefs*, (2) *trusting beliefs* on *risk beliefs*, and (3) *risk beliefs* on the *willingness to use*. That is expected as users with more *privacy invasion* experience are more aware of the social network risks [300]. Therefore, it is more challenging to reduce their *risk beliefs* by mitigating privacy concerns or increasing *trusting beliefs*.

3.7.4 *Impact of software features on privacy concerns*

Our results showed that the developed features reduced the privacy concerns. However, this reduction is minor especially for the concerns *improper access* and *unauthorized secondary use*. This can be due to the fact that these two concerns are partially based on the benevolence of the provider, which is difficult to be adequately reflected through the user interfaces of the application. Additionally, we notice that the impact of the features relates to how direct a particular feature addresses a concern. Some features confront users explicitly with the targeted concern, e.g., we deliberately included an error in the application to present how the application mitigates the *errors* concern. This led to a more prominent impact of the features. In contrast, other features have a smaller impact, which can be due to mitigating the concern without a clear reference to it, e.g., unauthorized secondary use.

Remarkably, we found that some features that were meant to address a specific concern affected also other concerns. In particular, the features of the *lacking awareness* concern are found simultaneously to mitigate the *insufficient control* concern. Thus, we conclude that raising the users' awareness positively contributes to an enhanced feeling of control. On the other hand, experiencing errors during the usage of the application led to a reduced feeling of control. That can be explained by the fact that the errors deviate the application workflow from the expectations of the user, and more importantly, hinder the proper response to the user actions, which makes the user feels less in control.

3.7.5 *Limitations*

This study contributes to the existing body of knowledge, however, it has several limitations.

1 Our study focuses on one approach to privacy-friendly OSNs, namely HOSN, and considers a single application in particular, Hushtweet. There are several other OSN approaches (e.g., DOSNs) and applications (e.g., Diaspora) that also aim to provide privacy-enhanced services.

2 Although we recruited 2300 participants for the user study, it is challenging to guarantee that our sample is representative of the Twitter population in terms of demographic and privacy-related variables. As mentioned in Section 3.6.1, the education level among our participants is higher than that of Twitter users. Our participants are mainly from the U.S. and India, the countries with the most Twitter users in the first and third places [196]. However, we had no participants from Japan, the country with the second highest number of Twitter users.

3 Despite our careful selection of the implemented software features, we cannot ensure that they influenced participants similarly or contributed to the various concerns in the same way. Further research could be conducted in this regard, for example, by including interview questions as part of the study to assess the impact of each feature on different participants.

3.8 CONCLUSION

The users' privacy concerns in OSNs have increased especially in the last few years. This is due to the practices of service providers that compromise the privacy of users, e.g., massive data collection and data exchange with third parties (e.g., data brokers). Additionally, the recurring data breaches and the emergence of privacy regulations (e.g., GDPR) helped raise users' awareness of privacy risks. Hushtweet provides an alternative solution that allows users to enjoy the social exchange with additional means of privacy control. However, as with any novel (privacy-friendly) technology, it is challenging to gain users' trust and adequately convey the privacy practices through the user interfaces.

In this chapter, we studied, in the context of OSNs, the relationships between *(mitigated) privacy concerns*, *trusting beliefs*, *risk beliefs*, and *willingness to use*. We showed that mitigating privacy concerns with software features in the user interface of Hushtweet increases its trustworthiness. The software features particularly affected older people and those with less experience regarding privacy-related issues. Our study also indicated that the more people trust the application, the higher is their *willingness to use* it. Furthermore, it became apparent that the choice of concern to be dealt with in the application development should be wisely made because some concerns are notably more important than others. For instance, we found that the *lacking awareness* concern is highly relevant for OSN's users, and addressing this concerns provides users additionally with a sense of control.

The previous chapter proposed an approach to privacy-friendly OSNs and demonstrated its viability through actual implementation. This chapter furthered in that line of work by studying the users' perception of the proposed approach from different aspects. In the light of the requirements met by this approach, we proceed in the next chapter by elaborating on the second research question RQ2: *How to apply distributed analytics in OSNs privately and efficiently?*.

Part III

PRIVACY-ENHANCED DISTRIBUTED ANALYTICS

4

PRIVACY-ENHANCED ASSOCIATION RULE MINING

In the last part of the thesis, we presented the concept of Hybrid Online Social Network (HOSN), which combines benefits from the central and decentralized variants (COSN and DOSN). HOSN allows users to maintain their data distributed in a private network beyond the reach of the central service providers. Under this setting, applying analytics on the user distributed data can be of a high value for the users. The analytics can be used to customize the social network services. Moreover, the results of the analytics can be exchanged with the service providers or third parties in return for additional services or monetary compensation. However, applying analytics on distributed data while preserving user privacy is challenging. In this chapter, we address this challenge for one particular analytics technique, namely Association Rule Mining (ARM), which can be used as a basis for various recommendation services.

4.1 INTRODUCTION

OSNs use recommender systems to improve many of their services by suggesting interesting and *new* content to individuals, e.g., suggesting songs in Spotify, or recommending friends on Facebook. Recommender systems are built using a combination of data mining and machine learning methods [184]. One of the efficient methods is Association Rule Mining (ARM) [265]. ARM captures relations between items; these relations can, for example, lead from items of interest to the user to potentially interesting new items. ARM is simple yet effective and additionally, it has two privacy-friendly features: (1) it is independent of any personal user model because the rules naturally create an abstraction from the users and focus on inherent traits of the items that cause them to be linked together. (2) Unlike other recommender systems where extensive user- or item-profile information is needed (e.g., collaborative and content-based filtering systems), ARM can function in the absence of a rich history of user preferences or behavior [200].

Previously, several approaches were proposed to apply ARM in a privacy-preserving manner [57, 83]. In distributed systems, the state-of-the-art privacy-preserving ARM approaches are cryptography-based and use the Apriori algorithm for mining the rules [36, 256]. However, the number of required cryptographic operations and the Apriori computations restrict the efficient applicability of these approaches to limited-scale applications, rather than large-scale applications such as OSNs.

4.1.1 *Summary of contributions*

In this chapter, we present an approach to achieve very good real-world performance for privacy-enhanced ARM on distributed data. More precisely, our work focuses on FI mining, which is the most expensive phase of ARM. Our approach is based on a combination of graph sampling and distributed FI mining. More precisely, we introduce a privacy-enhanced version of the Metropolis-Hasting Random Walk (MHRW) [250] sampling method for social graphs. The proposed method derives representative data samples from a limited and randomly selected set of users. For FI mining, we propose using a distributed FP-Growth algorithm [147], which boosts the efficiency of the mining process remarkably. Our proposed combination does not only improve the privacy of users by curtailing the data required for the mining, but also reduces the communication and computational overhead. We evaluate our approach on three large-scale datasets, collected from real-world social networks. Results show that we maintain very high precision and recall rates for quite small samples in well-connected networks while sweeping low communication efforts. The content of this chapter is based on the following publications.

PUBLICATIONS

- **Wainakh, A.**, Grube, T., Daubert, J., & Mühlhäuser, M. (2019, September). Efficient privacy-preserving recommendations based on social graphs. In *the 13th Conference on Recommender Systems (RecSys) (pp. 78-86). ACM.*
- **Wainakh, A.**, Strassheim, A., Grube, T., Daubert, J., & Mühlhäuser, M. (2021, August). Enabling Privacy-Preserving Rule Mining in Decentralized Social Networks. In *the 16th International Conference on Availability, Reliability and Security (ARES) (pp. 1-11).*

Contribution Statement
I was the main author and led the process of idea generation, implementation, evaluation, and writing. Tim Grube and Jörg Daubert contributed with helpful discussions and suggested formulations improved the editorial quality. Aleksej Strassheim contributed to a part of the implementation through his Bachelor's thesis. Max Mühlhäuser provided helpful mentoring and critical reviews.

4.1.2 *Outline*

The remainder of this chapter is organized as follows. First, we present a background on graph sampling and ARM under our application scenario in Section 4.2. Then, we present related work on privacy-preserving and efficient ARM in Section 4.3. In Section 4.4, we provide a detailed description of our approach, including our sampling and FI mining processes. Section 4.5 discusses

our simulation study and the results of our experiments. Finally, Section 4.6 summarizes our contributions and draws conclusions.

4.2 BACKGROUND

In this section, we introduce (1) graph sampling and (2) ARM. Then, we formalize our problem setting.

4.2.1 *Graph sampling*

Graph sampling is a technique to derive a subset of nodes and/or edges from a larger graph that represents the original population. This technique aims at preserving the characteristics of the original population of the sample. The most relevant properties of the sampling process are (1) the selection method and (2) the sample size. As the related work covers the effect of the size of the sample extensively, even for ARM [43, 205], our work takes a closer look at sampling methods for ARM with special consideration of privacy.

There are multiple classes of graph sampling methods, namely node (vertex) sampling, edge sampling, and traversal-based sampling [120]. In the node and edge sampling classes, a set of nodes and edges are selected at random. Speaking of OSNs, the nodes would typically represent users and the edges would represent friendship relations. Applying these sampling methods requires global knowledge, i.e., requires the knowledge of the population and their connections, such that valid users or connections can be randomly chosen [146, 217].

In contrast, traversal-based sampling approaches start with a set of users and grow the sample iteratively based on the users' connections. One of the most common approaches in this category is the class of Random Walks (RWs), which contains a variety of algorithms that sample a graph by walking from users to their connections and beyond [217, 249]. In RW, the next user is selected from the neighbors of the current user with uniform probability. One main limitation of RW is its bias towards highly-connected users, leading to a large deviation from the desired uniform distribution [151]. MHRW [178] was proposed to mitigate this limitation. It was adapted and used for P2P networks and social networks [88, 250]. MHRW introduces a proposal function to change the transition probabilities. More specifically, the proposal function reduces the probability of visiting users who have a larger number of neighbors $\mathcal{F}$ (highly-connected) compared with the currently visited user u_k. That is achieved as follows. After selecting the next user (candidate) u_{k+1} , a uniform random $p \in [0, 1]$ is generated. If the inequality (4.1) holds, we say the proposal is accepted and the walker proceeds to the candidate, otherwise, another neighbor is selected.

$$\frac{|\mathcal{F}(u_k)|}{|\mathcal{F}(u_{k+1})|} \geqslant p : 0 \leqslant p \leqslant 1 \tag{4.1}$$

Another limitation of RW is that it might get stuck, especially in directed graphs, when it reaches sink users or local dense communities. Sink users are

users without outgoing connections, thus, sampling walkers can reach them but cannot proceed to other users afterward. To address this issue, several proposals exist, e.g., multiple independent RW [88] and multi-dimensional RW [217].

4.2.2 *Association rule mining*

ARM captures the relations between items. In OSNs, items refer to a user's interests. An interest is expressed by the user creating or consuming content, e.g., liking or adding a post, or joining a group. We refer to a single piece of content as an *item* i from the set $I = \{i_1, i_2, \ldots, i_m\}$ of all items. Let the set $U = \{u_1, u_2, \ldots, u_n\}$ comprises all users u. The *interested in* relation $r_{u,i}$ models a user's u interest in an item i. We denote this relation as

$$r_{u,i} = \begin{cases} 1 & \text{if } u \text{ is interested in } i \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

For each user $u \in U$, let T be the set of interesting items for u such that $T \subseteq I$ and $\forall i \in T, r_{u,i} = 1$. Subsequently, the dataset D is the set of all users' interesting items $D = \{T_1, T_2, ..., T_n\}$.

ARM derives *Frequent Itemsets (FIs)* from the dataset D. The *support* $s_D(X)$ of an itemset $X \subseteq I$ is the ratio of users for which the set of interesting items T in D contain X. An itemset X is considered frequent, if its support exceeds the *support threshold* θ, i.e., if $s_D(X) \geqslant \theta$. The set FI_D summarizes the frequent itemsets found in the dataset D. Given two distinct itemsets X, Y with $X \subset I$, $Y \subset I$, and $X \cap Y = \emptyset$, an *association rule* ar_i is an implication $X \rightarrow Y$. For each association rule ar_i, the confidence $c_D(ar_i)$ indicates how often the rule is found to be true. Association rules ar_i that exceed the *confidence threshold* β, i.e., $c_D(ar_i) \geqslant \beta$, are considered *reliable* in D. The problem space of ARM covers two main challenges: (1) deriving the itemsets FI_D under θ and (2) establishing *reliable* association rules of FI_D under β. In this work, we focus on the first challenge as it causes the main communication and computational costs.

To derive the FIs, there exist several algorithms [113]. Two of the most common algorithms are Apriori [3] and FP-Growth [102]. Apriori is a breadth-first search methodology, where the main idea is to iteratively generate the candidate itemsets of size $k+1$ from only FIs of size $k : k \geqslant 1$. This is based on the observation that if any itemset of size k is not frequent, then its super-itemset of size $k+1$ cannot be frequent. With this technique, Apriori reduces the search space, i.e., the number of the candidate itemsets. However, it is still considered a costly algorithm for two reasons. First, in the case of a large number of frequent 1-itemsets, e.g., 10^6, Apriori generates more than 10^{11} 2-itemsets candidates. Second, it requires multiple passes over the whole dataset to count the occurrences of all candidate itemsets in every stage of the algorithm [102].

In contrast, frequent-pattern growth (FP-Growth) does not generate or test candidate itemsets and requires only two passes over the dataset. In the first pass, FP-Growth computes the list of frequent 1-itemsets. In the second pass, it transforms the dataset into an FP-tree structure, where each node of the tree

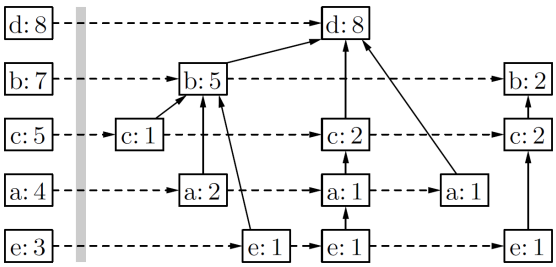


Figure 4.1: Example of FP-tree. Each node indicates an item and the frequency of the prefix path (itemset).

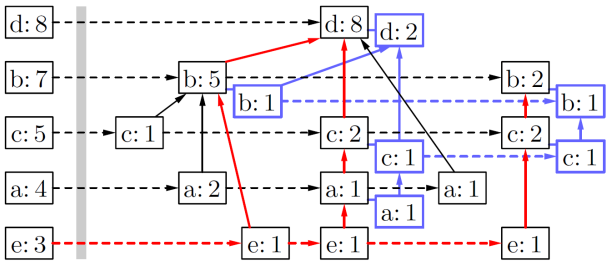


Figure 4.2: Example of Conditional FP-tree for item e is in blue. This sub-tree is extracted from FP-tree by selecting the prefix paths of item e.

represents an item and each path in the tree represents an itemset, as shown in Figure 4.1. Then for each item, we build the conditional FP-tree, which consists of prefix paths in the FP-tree occurring with the item as the lowest node (suffix), as shown for item e in Figure 4.2. Only paths (itemsets) meeting the threshold support are considered. Lastly, from the conditional FP-tree, the frequent itemsets are derived.

4.2.3 *Problem setting*

We consider a DOSN, where the data is fully distributed among users, i.e., each user $u_k \in U$ has their own data (interesting items) T_k. Our goal is to mine FIs from the distributed user data while meeting the following requirements.

- *Privacy*: Observing the users' privacy demands in DOSNs, mining FIs from users' items poses the challenge of protecting the items from other users. As complete obfuscation (unobservability) would defy the very purpose of a social network, we rely on the *relaxed* privacy requirement of *unlinkability*: No user should be able to link items to an individual user, i.e., determine the items of any user. We consider users to follow the *honest-but-curious adversary* model, i.e., users attempt to use the received information to learn the items of others without deviating from the protocol.

- *Distribution*: One of the key techniques of DOSNs to achieve privacy is to eliminate the full control of central entities. To align with this distribution, FI mining—originally designed as a single database algorithm—must be adapted to accept distributed data as well distributed control of the algorithm.

- *Efficiency*: We refer here to curtailed communications and computations. Most DOSNs applications are performed on decentralized servers with occasional inter-server interaction, while traditional FI mining requires the inclusion of

all users. The sheer scale of social networks renders FI mining in DOSNs one of the largest applications; consequently, any method applied in such applications needs to consider efficiency as a high priority requirement.

4.3 RELATED WORK

In this section, we first present related work on privacy-preserving ARM and second, on the efficiency of ARM.

4.3.1 *Privacy-preserving ARM*

Approaches that apply ARM in a privacy-preserving manner can be classified into two categories: (1) perturbation and (2) cryptography-based approaches.

In perturbation approaches, the data is anonymized before mining the rules by modifying [57], blocking [83], or sanitizing the sensitive attributes [71]. These anonymization techniques can—inherently—only maintain partial properties of the complete dataset. Moreover, unlike our problem setting, most of the approaches in this category assume a centralized environment.

In contrast, cryptography-based approaches usually assume distributed data. Multiple parties apply ARM collaboratively without disclosing their data. A leading approach [256] was proposed by Kantarcioglu et al. [133], where the rules are mined in two steps. First, local FIs (per user) are collected and combined via a secure union. Next, a secure sum protocol is used to identify FIs that also meet the global support. The secure union process requires each user to encrypt the FIs of every other user using RSA encryption to achieve the commutative property. This yields a computational cost of $O(\frac{t^3 n^2}{support})$, where t is the number of bits in the encryption key, and n is the total number of users. In addition, the users need to send $O(n^2)$ messages in total. Thus, the computational and communication loads on each user increase with the number of users in a second-order polynomial rate. In large-scale applications, with millions of users, this introduces a substantial overhead on the users' devices. Some improvements for the t^3-factor exist, e.g., using secure multi-party algorithms for itemset intersections [256] and using Elliptic-curve cryptography [36]. Collectively, despite some improvements, the computational and communication costs of these approaches remarkably increase with the number of users, which limits their suitability to smaller-scale applications than OSNs.

4.3.2 *Efficient ARM*

We suggest that the efficiency of privacy-preserving ARM protocols can be improved by using efficient ARM techniques, therefore, we investigate these techniques. Kotsiantis et al. [138] classified them into four categories: (1) data sampling, (2) reducing the passes over the dataset, (3) parallelization, and (4) adding extra constraints on the structure of rules. In this section, we focus on the first three categories, as we leverage them in our approach.

For data sampling, some works focus on determining the proper sample size to achieve a desired accuracy for ARM, e.g., by using progressive sampling [43]. Other works proposed different sampling methods for the items, e.g., multi-stage sampling [312], and ontology-based sampling [286]. However, the sample might not contain all itemsets with the same occurrence rate as in the complete dataset. This phenomenon is known as *sampling error* [227]. To alleviate this issue, the support threshold for the sample can be reduced [258] or the negative border can be used to obtain missed FIs [258]. More sampling techniques can be found in [37]. All the aforementioned sampling approaches tackle the problem of efficient ARM for centralized databases, which does not apply to our setting of distributed data.

To reduce the passes over the dataset, Han et al. [102] proposed the FP-Growth algorithm, which mines FIs based on a Frequent Pattern Tree (FP-tree) structure. FP-Growth improves the efficiency of the mining process by: (1) compressing the dataset into a much smaller data structure, (2) most importantly, reducing dataset scans to only two, (3) avoiding the costly generation of a large number of candidate sets, and (4) decomposing the mining process into a set of smaller tasks.

For parallelization, a follow-up work [147] proposed to parallelize the FP-Growth algorithm by partitioning the process into a set of independent mining tasks, which can be executed on distributed machines. This work is based on the MapReduce infrastructure. Similarly, Shi et al. [234] presented a parallel version of FP-Growth for the Apache Spark framework. Unlike our work, the previous proposals do not consider the privacy aspect of the mining process.

4.4 DISTRIBUTED PRIVACY-ENHANCED FREQUENT ITEMSET MINING

In this section, we explain our method, which consists of two steps: (1) user and itemset *sampling*, and (2) *frequent itemset mining*. In the sampling phase, we reduce both the communication and computational costs by restricting the number of users and itemsets involved in the mining process. A subset of users then mine the FIs collaboratively in a privacy-enhanced manner.

4.4.1 *User and itemset sampling*

In centralized ARM, the users' items are collected by a central server, where they are processed to derive the rules. In our privacy-enhanced scenario, the items remain distributed with their users. Applying distributed privacy-enhanced ARM (e.g., cryptography-based approaches) by all the users is inapplicable, as it yields massive communication and computational overhead. Therefore, we propose restricting privacy-enhanced ARM to a very limited number of users; we call them *prime users* $U' \subset U$ with $|U'| \ll |U|$. However, as each prime user has only their own items in DOSNs, the mining process might result in poor rules, i.e., the prime users' items might be insufficient to produce rules that represent the population. To address this issue, we sample items from a larger number of users in a privacy-enhanced manner, and pass these samples to the

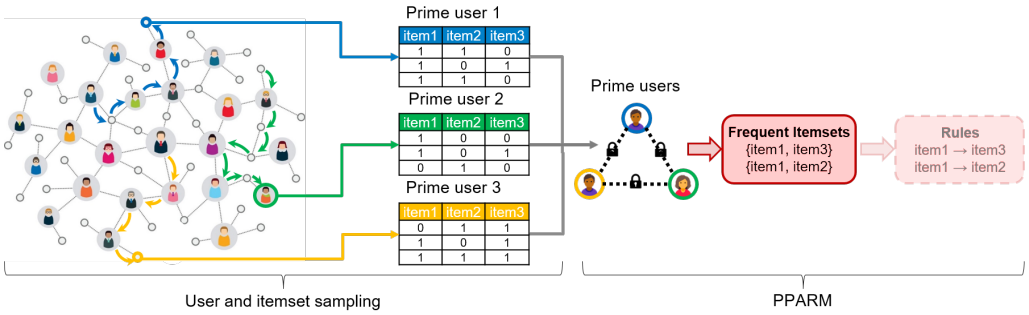


Figure 4.3: Overview of the sampling and mining processes[†].

prime users, who then apply privacy-enhanced ARM. With this approach, we address both, efficiency and privacy. Only a few users contribute items and communicate while preserving almost all frequent itemsets, hence, improving the efficiency. The low ratio of sampled users and the participation ambiguity render it impractical to link items to users.

Considering our requirements, privacy, distribution, and efficiency (see Section 4.2.3), we chose Metropolis-Hasting Random Walk (MHRW) [178] as our sampling method. MHRW is non-deterministic, as the users are selected randomly, thus, less predictable and that contributes to the privacy of the visited users (see Section 4.5.5). MHRW is traversal-based and distributed by nature, in addition, no global knowledge is required, in contrast to different random walk variants (e.g., Maximum-Degree RW [14]). Furthermore, MHRW mitigates the heavy bias of the traditional RW toward highly-connected users during the walk itself [151], unlike other proposals, which require central post-processing (e.g., Re-Weighted RW [223]).

PRIVACY. To maintain privacy, i.e., to conceal the relation of users and their items, we introduce two main changes to MHRW: First, by the nature of random walks, no user can tell which users participated in the MHRW beyond their predecessor and successor. In our approach, we enable the users visited by the walker instance to decide with a *contributing probability* p_{co} whether to contribute their items or not. Hence, with a sufficiently low p_{co}, neither a MHRW successor nor the prime user can distinguish if the predecessor contributed items or not. Second, to calculate the proposal function in MHRW, the degrees of the current user u_k and the selected successor (candidate) u_{k+1} are needed as shown in Eq. (4.1). To privately share the necessary information, we redefine the proposal function as follows.

$$\frac{|\mathcal{F}(u_k)|}{p} \geqslant |\mathcal{F}(u_{k+1})| : 0 < p \leqslant 1 \tag{4.3}$$

The user u_k calculates $|\mathcal{F}(u_k)|/p$, sends it to u_{k+1}, and expects a boolean answer. We refer to MHRW with the two aforementioned changes as Metropolis-Hasting *Anonymous* Random Walk (MHARW).

SAMPLING WORKFLOW. To collect a sample of a desired size $|S_T|$, we employ a set of walker instances W, each is launched to collect a sub-sample $|sS_T|$,

Algorithm 1: User and Itemset Sampling†

Data: U: set of all users, S_k: sub-sample No. k, T_k: items of user u_k, p_{in}: initializing probability, p_{co}: contributing probability

```
1  At each user u_k ∈ U: u_k randomly generates 0 ⩽ p_1, p_2, p_3 ⩽ 1;
2  if p_1 < p_in then
3  |  u_k starts a walk;
4  end
5  if |S_k| < |sS_T| then
6  |  if p_2 < p_co ∧ ¬contributed then
7  |  |  add T_k to S_k at random position;
8  |  |  contributed ← True;
9  |  end
10 |  while ¬terminating_case do
11 |  |  u_k randomly selects u_{k+1} ∈ F(u_k);
12 |  |  u_k sends a proposal |F(u_k)|/p_3 to u_{k+1};
13 |  |  if response = True then
14 |  |  |  if (u_{k-1} = u_{k+1}) ∧ contributed then
15 |  |  |  |  remove T_k from S_k;
16 |  |  |  end
17 |  |  |  u_k sends S_k to u_{k+1};
18 |  |  |  break;
19 |  |  end
20 |  end
21 end
22 else
23 |  u_k becomes a prime user U' ∪ {u_k};
24 end
```

such that $|sS_T| \cdot |W| \approx |S_T|$. The complete sampling process goes as follows (more details in Algorithm 1).

1 *Establishing the walks:* We establish multiple MHARW walker instances in parallel as shown in Figure 4.3. The number of walks $|W|$ can be estimated by an *initializing probability* p_{in}, with which each user in the DOSN starts a walk: $|W| \approx |U| \cdot p_{in}$.

2 *Contributing to the sample:* Each walker instance collects a sub-sample of items. A user visited by a walk determines locally whether to contribute to the sub-sample based on p_{co}.

3 *Passing the sample to a friend:* The sub-sample is passed from a user u_k to their successor u_{k+1}, which is selected randomly from the u_k's friends. A proposal is made by u_k and sent to u_{k+1}. If the friend accepts the proposal, the successor is determined to be u_{k+1}. In case the successor u_{k+1} and predecessor u_{k-1} are the same user, the user u_k does not contribute to the sub-sample to avoid leaking their items.

4 *Termination:* As soon as the sub-sample size reaches the predefined target size $|sS_T|$ or a terminating case, the walk stops. The last user in the walk becomes a prime user u'.

5 *Compensating sample offset:* To ensure the target sample size $|S_T|$ is reached even when some MHARW instances get stuck, we introduce Algorithm 2: The prime users establish a network (e.g., through a relay server) to communicate and share the sizes of their collected sub-samples. Then, each prime user can locally deduce whether the target sample size $|S_T|$ is reached. If not, the prime users who are unstuck, i.e., reached $|sS_T|$ successfully, proceed the walks to compensate the shortage of the sample entries. Consequently, these resumed walks end up with new prime users.

The main parameters $p_{in}, p_{co}, |S_T|$ along with time-stamps for synchronizing the establishment of the walker instances can be predefined (hard-coded) in the application of the DOSN.

Algorithm 2: Compensating Sample Offset†

Data: S_k: sub-sample of prime user u'_k, $|sS_T|$: target sub-sample size

Prime users share their sub-sample sizes with each other;
At each prime user $u'_k \in U'$:
if $|S_k| = |sS_T|$ **then**
    Let ν be the number of users who are unstuck;
    for $S_i \in \{S_1, .., S_w\}$ **do**
        if $|S_i| = |sS_T|$ **then**
            $\nu \leftarrow \nu + 1$;
        end
    end
    $\text{increment} = \left(\sum_j |sS_T| - |S_j|\right) / \nu$;
    $|sS_T| \leftarrow |sS_T| + \text{increment}$;
    Resume sampling;
end

4.4.2 *Frequent itemsets mining*

Once the sample is collected, the prime users collaboratively mine the FIs. We proposed to use the FP-Growth algorithm [102], which skips the expensive candidate generation of Apriori, and requires only two passes over the dataset D [102]. As we consider distributed data, we base our approach on the parallelized version of the FP-Growth algorithm, Distributed FP-Growth (DFP) [147]. The DFP algorithm has been used for big data frameworks, MapReduce and Spark [147, 234]. In this work, we apply DFP under a privacy-friendly setting as follows.

1 *Counting item support:* Prime users collaboratively count the support values of all items by applying the secure sum protocol [133] (extensions of this protocol can also be applied, e.g., CRDM [269] and k-secure sum [317]). For that, the prime users form a ring network and select one user to establish the protocol. The selected prime user u'_1 creates a list of all their items and counts their supports (see Figure 4.4). To protect this information from other users,

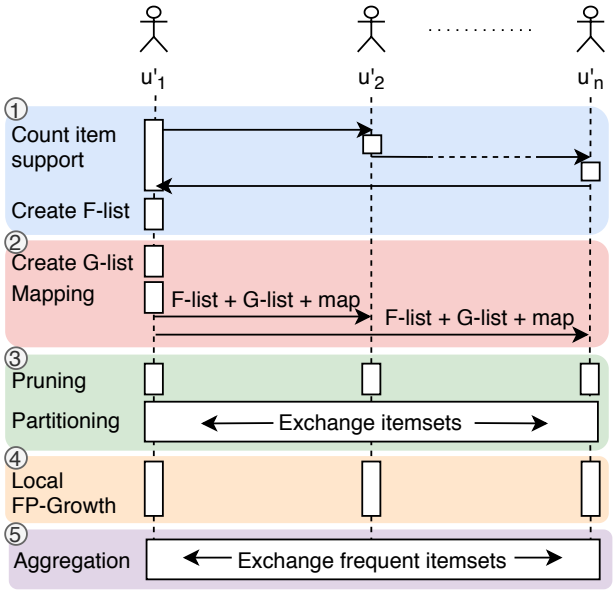


Figure 4.4: Sequence diagram of distributed FP-Growth†.

user u'_1 takes two measures: (a) adding random integers to the supports, (b) appending random items (non-existing in their data) with random support values. The list is passed from a prime user to another through the ring, where each user adds their items' supports to the list. The round ends up at u'_1 who subtracts the random supports. Then, he omits all the infrequent items i, i.e., items with support $s_D(i) < \theta$. The resulting list is called *F-list*.

2 *Grouping items:* The user u'_1 divides the items of F-list into a set of groups; the number of the groups corresponds with the number of prime users $|U'|$. The list of the groups is called *G-list*. Then, the groups are randomly assigned to the prime users in a mapping list. Next, all lists, i.e., F-list, G-list, and mapping list, are shared with all prime users.

3 *Partitioning:* Every prime user prunes their itemsets (sub-sample) by deleting the infrequent items. Further, each itemset is sorted in descending order w.r.t. the support values of the items. Then, each item $i \in T_k$ is substituted with the corresponding group-id. For example, we have after pruning and sorting the itemset $T = \{i_1, i_4, i_6, i_3\}$, as shown in Figure 4.5. Substituting the items with their group-ids yields $T = \{g_1, g_2, g_2, g_1\}$. For each group-id, say gid, if it appears in T, locate its right-most appearance, say L, and output the itemset $\{T[1]...T[L]\}$. This itemset is sent to the corresponding prime user. In our example, for $gid = 1$, the right-most appearance $L = 4$, thus, the output itemset is $\{i_1, i_4, i_6, i_3\}$. For $gid = 2$, $L = 3$, and the output itemset is $\{i_1, i_4, i_6\}$.

4 *Local FP-Growth:* Each prime user receives the itemsets associated with their group and applies FP-Growth locally. This starts with creating a local FP-tree, followed by building the conditional sub-trees, and finally deriving the FIs. More details can be found in [102].

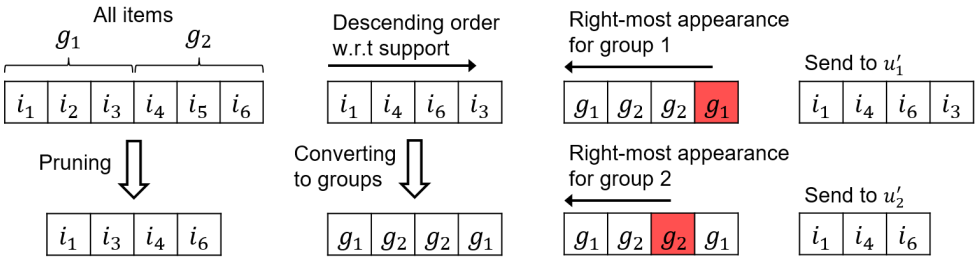


Figure 4.5: Partitioning phase of DFP.

5 *Aggregation:* The resulting FIs are then shared with all the prime users and aggregated to have the complete list of FIs.

By the end of this process, the total FIs are known by all the prime users. FIs then can be used to build association rules or for further analysis.

4.5 EMPIRICAL EVALUATION

In this section, we first explain the setup of our experiments based on real-world datasets. Second, we evaluate the correctness of the FI mined from the samples, and the quality of the sampling method. Third, we discuss the efficiency and privacy aspects of both the sampling and mining approaches.

4.5.1 *Experimental setup*

We use three datasets collected from real-world social networks [180]: Flickr (social photo sharing), Orkut (social networking), and Livejournal (blog with social networking). In all these networks, users can join interest groups and form friendship connections with each other; the used datasets contain both information. The interest groups are used as items for FI mining. Table 4.1 shows statistical information about the datasets. Prior to our experiments, we filtered out the isolated users, which exist in very limited numbers in the real social networks; excluding these users is a common practice in the field [283]. All experiments were repeated ten times and the average of the measures is reported. Throughout the experiments, we use the following parameter settings.

Datasets	***Flickr***	***Livejournal***	***Orkut***
#Users	1,715,255	5,203,764	3,072,441
#Connections	22,613,980	76,937,805	223,534,301
Avg. node degree	13.18	14.78	72.75
#Interest groups	103,648	7,489,073	8,730,859
Avg. groups per user	4.62	21.25	106.44

Table 4.1: Statistics of the datasets†.

Sample size	p_{in}	p_{co}
20%	0.5×10^{-2}	0.5
1%	2.5×10^{-4}	0.5

Table 4.2: Values of initializing and contributing probabilities†.

Dataset	θ	#FIs
Flickr	0.16%	1003
LiveJournal	0.73%	1001
Orkut	1.47%	1001

Table 4.3: Support thresholds†.

- *Initializing probability* p_{in}*:* The parameter p_{in} estimates the number of walks performed to collect the sample. Small p_{in} leads to a few and long walks. The longer the walks are, the more likely to get stuck. Therefore, we keep the length of the walks moderated by adapting the p_{in} for different sample sizes (see Table 4.2).

- *Contributing probability* p_{co}*:* Smaller p_{co} increases user privacy, but also increases the communication overhead by leading to longer walks. Thus, this parameter should be customized to reach a desired balance between privacy and efficiency. We fix $p_{co} = 0.5$ in all the experiments. With this probability, an adversary can only randomly guess whether a visited user is part of the sample.

- *Termination condition.* We limit the maximum length of all the walks to $100 * |sS_T|/p_{co}$. For MHARW, we also limit the number of proposals sent by a user to 1,000.

- *Support threshold* θ *(dataset):* Itemsets are considered frequent, if their supports exceed the threshold θ. In FI mining, θ is application-specific; it is usually adjusted interactively until meaningful FIs are discovered. We empirically choose a set of θ values (see Table 4.3) that produce a considerable number of FIs ($\geqslant 1,000$).

- *Support threshold* θ_S *(sample):* To reduce the sampling error (see Section 4.2) , we set $\theta_S \leqslant \theta$ as suggested in [258]. For the experiments, we use a range of values for θ_S and observe the correctness of FIs. We adjust the range of θ_S such that the whole spectrum of the precision and recall rates from 0 to 1 is observed.

4.5.2 *Frequent itemsets correctness*

We apply the FP-Growth algorithm on a dataset D and the respective sample S. The FIs found in both D and S are referred to as FI_D and FI_S, respectively. To evaluate the correctness of FI_S, we consider FI_D as a reference and calculate the *precision* and *recall* rates. We also compute the area under the precision-recall curve using the *average precision* score AP.

We compare MHARW with the traditional RW, the Anonymous Random Walk (ARW) [274], and uniform sampling (UNI) as baselines. Although the uniform sampling is not applicable in our distributed application scenario (see Section 4.2), we consider it since it was used in several approaches to improve the

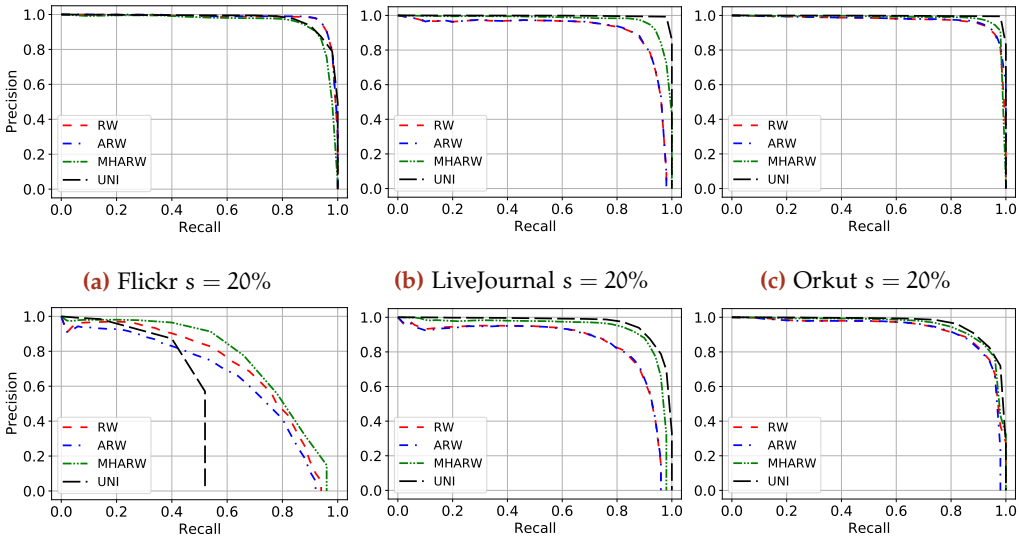
efficiency of ARM, e.g., [205, 258]. We treat all the social graphs of our datasets as undirected; we elaborate more on this decision in Section 4.5.3.1. We discuss our results in the light of different (1) sample sizes and (2) datasets.

4.5.2.1 *Sample size*

We conduct experiments with different sample sizes $s \in \{1\%, 20\%\}$. In Figures 4.6 (a-c), we can see that under the sample size $s = 20\%$, all the examined traversal-based methods achieve high precision and recall rates ($AP > 0.95$). For lower sample size $s = 1\%$, i.e., less participating users, we notice that the precision and recall rates degrade in all the datasets and for all sampling methods. That is expected as sampling errors increase with smaller sample sizes [43]. However, interestingly, MHARW shows high resilience to the changes in the sample size with $0.02 \leqslant \Delta AP \leqslant 0.07$, compared with RW and ARW where $0.02 \leqslant \Delta AP \leqslant 0.3$. Consequently, the superiority of MHARW over RW and ARW becomes more notable in lower sample sizes as shown in Figures 4.6 (d-f).

4.5.2.2 *Datasets*

Here, we take a look at the impact of the different dataset topologies and data distributions on the correctness of FIs. In Flickr, the Figure 4.6 (d) shows the sample size of 1%, where MHARW reaches $AP > 0.87$, while UNI has low performance with $AP \leqslant 0.80$ and a recall of 0.7. This can be explained by the fact that Flickr contains a high number of users with none or a small number of interest groups (items) [180]. Including these users in the sample reduces the quality of the FIs. Considering the correlation between degree distribution and the group count [180], the traversal-based sampling methods, which are biased towards highly connected users, are less likely to include these users compar-



(a) Flickr $s = 20\%$ **(b)** LiveJournal $s = 20\%$ **(c)** Orkut $s = 20\%$

(d) Flickr $s = 1\%$ **(e)** LiveJournal $s = 1\%$ **(f)** Orkut $s = 1\%$

Figure 4.6: Precision-recall rates of the sample frequent itemsets†.

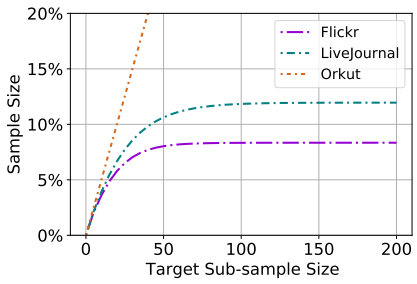


Figure 4.7: Reached sample sizes by random walk†.

ing with UNI. Among the walks, MHARW performs the best. This indicates that including a moderated number of highly connected users increases the quality of the sample in this case.

In LiveJournal, MHARW outperforms RW and ARW in both sample sizes. The difference becomes more notable in sample size $s = 1\%$, where MHARW has $AP = 0.93$ while the other walks have $AP \leqslant 0.85$.

In Orkut, we notice in Figure 4.6 (c,f) that all the walks perform very well and close to UNI with $AP \geqslant 0.94$. However, our proposed MHARW is consistently better with $AP \geqslant 0.96$ than the baselines RW and ARW. We reckon that this high correctness stems from the higher connectivity (average degree 72.75) in Orkut, which allows the walker instances to smoothly pass through different parts of the social graph, thereby collecting representative samples.

4.5.3 *Sampling quality*

We evaluate our sampling method considering two factors: (1) reaching the target sample size, and (2) replicating the Node Degree Distribution (NDD) of the population, which is one of the main graph properties [283].

4.5.3.1 *Reaching the target size*

RW, and inherently ARW and MHARW, get stuck, if they enter local communities or reach sink users. With this challenge in mind, we analyze in this section whether RW is able to reach the target sample size in our datasets.

We target sample size $s = 20\%$ with $p_{in} = 0.5 \times 10^{-2}$. In Figure 4.7, we plot the growth of the sample size (y-axis) w.r.t. the target sub-sample size (x-axis). Increasing the target sub-sample size allows the walks to proceed, if possible, to reach the target sample size. In Flickr and LiveJournal (directed graphs), we can see that the RW converges to a constant sample size and does not reach the target size. This is due to hitting sink users and getting stuck within local communities. In contrast, Orkut is an undirected graph, thus, there are no sink users. Therefore, we notice that the sample size increases linearly with the target sub-sample size.

In a further analysis, the same behavior was observed for both ARW and MHARW. These methods converge at a lower sample size comparing with RW because they require longer walks, as only a subset of the visited users contribute their items, which in turn increases the chance of reaching sink users.

To reach the target sample size, we discuss two approaches. First, we can increase the number of walks, which leads to shorter walks, thus, less likely to reach sink users. However, more walks correspond to more prime users. That causes a remarkable rise in the communication cost for FI mining. We elaborate more on the correlation between the communication cost and the number of prime users in Section 4.5.4.3. The second mitigation is to treat the social graph as an undirected graph as suggested in [282], assuming that users can communicate bidirectionally in DOSNs. Therefore, sink users no longer exist, as every link to sink users can also be used again to return to the previous user. The results of this mitigation reveal that RW, ARW, and MHARW in Flickr and LiveJournal show linear behavior similar to Orkut.

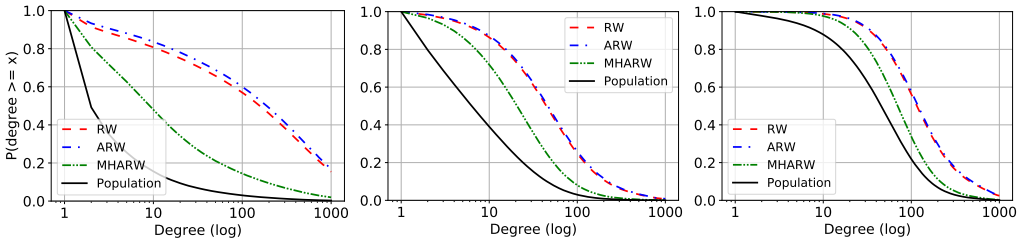
4.5.3.2 *Node degree distribution*

In this section, we analyze the NDD of the user samples, which is one of the important properties of graphs [283]. Related work [180] revealed a correlation between the degree distribution and the group count in our examined datasets. Therefore, NDD does not only represent a graph property but can also be considered as an indicator of the item distribution among users. We treat all graphs as undirected graphs.

We represent NDD through the complementary cumulative distribution function $\varphi(x) = P(|\mathcal{F}(u)| \geqslant x)$, where $|\mathcal{F}(u)|$ is the degree of a user $u \in U$ [35]. Figure 4.8 shows that the behaviors of RW and ARW are almost congruent on all datasets. As assumed before, both reveal a bias towards users with higher node degrees. This is evident from their node degree distribution, which is clearly skewed towards higher degrees. In contrast, we see that MHARW mitigates the bias to some extent and behaves closer to the population. Interestingly, MHARW shows very limited changes in response to a considerable change of sample sizes. This became apparent when we applied the same experiment on sample size $s = 20\%$.

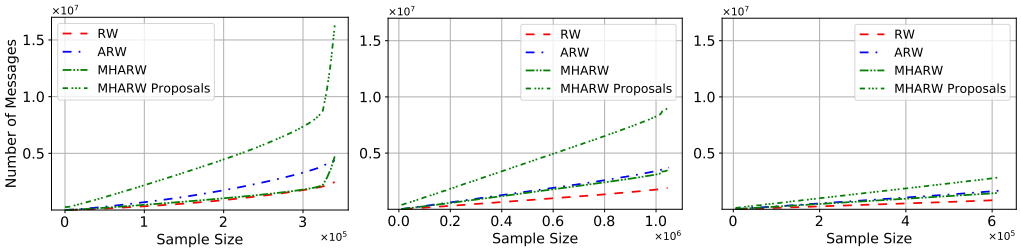
4.5.4 *Efficiency*

In this section, we evaluate the efficiency of our approach w.r.t. the number and size of messages during the sampling and mining process.



(a) Flickr **(b)** LiveJournal **(c)** Orkut

Figure 4.8: Node degree distribution of sample $s = 1\%$ †.



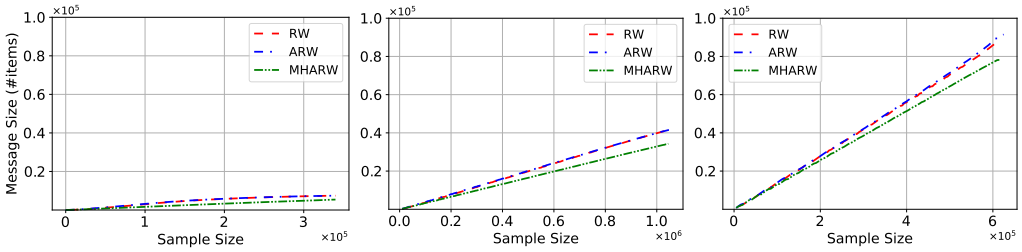
(a) Flickr **(b)** LiveJournal **(c)** Orkut

Figure 4.9: Number of messages for sampling $s = 20\%$ †.

4.5.4.1 *Number of messages for sampling*

Figure 4.9 shows the number of sent messages for RW, ARW, and MHARW, with a sample size of $s = 20\%$ on all datasets. In addition, we count the proposals sent by users in MHARW (see Section 4.4.1). In our evaluation on the Flickr dataset, we can see that MHARW requires fewer messages than ARW and more than RW. ARW and MHARW utilize a contributing probability p_{co}, such that not every visited user contributes to the sample. Therefore, it is expected that both require more messages than RW to reach the same sample size. Since the MHARW reduces the bias towards highly connected nodes, it is, in contrast to ARW, less likely to be stuck in local, well-connected communities. These dead ends are causing the necessity of additional messages in the ARW-based sampling. Nevertheless, MHARW starts showing a steep ascend in messages at larger sample sizes. This can be an artifact of our proposed Algorithm 2 to compensate the sample offset, where the unstuck prime users proceed with their walks. Although this technique successfully extends the sample to reach the target sample size, it makes some walks longer, thus, more likely to be stuck.

In the analysis of the sampling methods on LiveJournal, we observe in Figure 4.9 (b) a similar behavior but with a smaller spike on larger sample sizes. On Orkut, the sampling methods do not spike for larger sampling sizes. Further, the message counts linearly correlate with the increase of the sample size. In contrast to Flickr and LiveJournal, Orkut is larger and the users obtain more connections. As such, the sampling methods have a lower probability of getting stuck in local communities, i.e., the sampling procedure is smoother. This also leads to a lower number of proposals for MHARW in Orkut, unlike Flickr and LiveJournal. However, the bandwidth usage of these proposals is limited as they are quite small messages. Following our adaptation of the proposal mentioned in Section 4.4.1, the current user sends a message with a float number and the candidate responds with one bit (0/1). Furthermore, the number of these proposals can be minimized by using a rejection-controlled walk [151], where the proposal acceptance ratio is increased. However, this leads to a bias in favor of the highly connected users and thus to a larger deviation of NDD from the population.



(a) Flickr **(b)** LiveJournal **(c)** Orkut

Figure 4.10: Size of messages for sampling $s = 20\%$ †.

4.5.4.2 *Size of messages for sampling*

Considering the maximum number of interest groups in our datasets (see Table 4.1), we assume that at most 3 bytes are required to communicate a group id. Hence, the size of a message would be the number of groups included in the message multiplied by a factor of three. However, for simplicity, our results indicate the size of the messages in terms of the number of included groups. Figure 4.10 depicts that the size of the messages of MHARW is notably smaller than RW and ARW in all the datasets. As highly connected users have more interest groups (items) [180], the bias of RW and ARW towards these users tends to collect samples with a larger number of items, thus, creating larger messages. The size of the messages is almost linearly growing with the sample size. The gradient of the message size growth is higher in Orkut compared with Flickr and LiveJournal. This confirms our assumption as Orkut's users have the highest average number of items (see Table 4.1).

4.5.4.3 *Number of messages for FI mining*

In this section, we compare our approach with UNIFI-KC [133], one of the state-of-the-art cryptography-based approaches for privacy-preserving ARM. UNIFI-KC requires $(n^2 + 2n - 3)k$ messages to mine the FIs, where n is the number of participants in the mining process, and k is the number of iterations, which also corresponds to the size of the resulting FIs. Meaning, to mine FIs of the size two, the algorithm goes through two iterations; that is a result of using the Apriori algorithm for mining. In contrast, our DFP approach requires $2n^2 - 1$ messages and does not require multiple iterations. In Figure 4.11, we can see the total number of messages of our DFP-based approach compared to UNIFI-KC with $k \in \{1, 2, 3\}$. We notice that the number of messages required for UNIFI-KC increases remarkably with the increase of k, while DFP can mine FIs of arbitrary sizes without generating any extra messages.

4.5.5 *Privacy*

In this section, we discuss the privacy benefits of our sampling and mining processes w.r.t. three aspects, namely data minimization, node degree protection, and unlinkability between users and items.

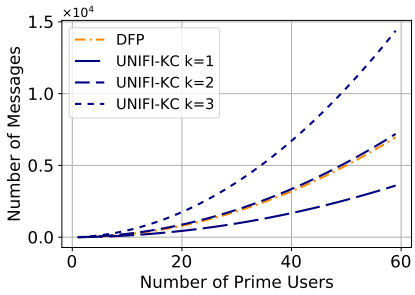


Figure 4.11: Number of messages for frequent itemset mining by UNIFI-KC and DFP†.

4.5.5.1 *Data minimization*

Data minimization is an enforced practice by legal frameworks, e.g., the EU's GDPR, to improve the privacy of users. For that, we only consider and process the data of a random subset of users (i.e., the randomly sampled users) while the data of the remaining users is hidden and, therefore, remains private. By adjusting the sample size, we can control the amount of data required (within the context of our approach) to achieve a desired performance. Experiments showed that by using the data of only 1% of the user population, our approach can derive high-quality FIs.

4.5.5.2 *Node degree protection*

The proposal function of MHARW is adapted in Eq. (4.3) to protect the node degree (number of friends) of the current visited user $|\mathcal{F}(u_k)|$. User u_k sends a proposal $|\mathcal{F}(u_k)|/p$ to the candidate successor and expects an accept/reject answer, where p is a random variable $0 < p \leqslant 1$. As mentioned in Section 4.2.3, we assume an honest-but-curious adversary, thus, all users follow the protocol properly. However, as $E[p] = 1/2$, the candidate could estimate the node degree given enough number of proposals. The error in estimating the node degree decreases with the increase of the number of proposals. Empirical results for this correlation are shown in Table 4.4. As the candidate is chosen with uniform probability, the number of proposals $|Pr(u_{k+1})|$ sent to a particular candidate u_{k+1} is correlated with the node degree of the visited user $|\mathcal{F}(u_k)|$ and the number of visits to that user u_k, $|Visits(u_k)|/|\mathcal{F}(u_k)| \approx |Pr(u_{k+1})|$. The higher the node degree is, the more likely that less proposals are sent to a particular candidate. Considering the average node degrees in our datasets (see Table 4.1), users in Orkut (with average node degree 72.75) are more protected against this leakage comparing with Flickr and Livejournal, where the average node degrees are lower, 13.18 and 14.78, respectively. For example, a user in Orkut with 72 friends can be visited probably 72 times before their node degree is leaked to one of their friends with an estimation error $\leqslant 50\%$.

Number of proposals	1	5	10	100	1000
Estimation error	50%	20%	14%	4.6%	1.4%

Table 4.4: Node degree estimation error.

The average number of visits per user can be estimated from the number of messages in Figure 4.9. Each message represents a step in the walk, i.e., a visit to one user. In our case, where we have connected and non-bipartite graphs, MHRW was proven to have a stationary distribution $\pi(u) = 1/n$ after a considerable number of steps [89, 250]. In other words, the probability of visiting a user at a specific walk step converges to $1/n$. Thus, we can estimate the average number of visits per user by dividing the number of messages by the population n. The resulting averages for the sample size $s = 20\%$ per dataset are shown in Table 4.5. As we can see, the average number of proposals is < 1 for all datasets, meaning that the risk of disclosing the node degree with estimation error $\leqslant 50\%$ is low. However, these estimations are based on averages and do not represent special cases, e.g., when the walks are stuck in local communities, where user can be visited much more frequently, and thus increasing the risk of leaking their node degree to their candidates. Potential mitigations for this issue can be:

- Restricting the number of proposals sent to individual candidates. For example, to achieve estimation error $> 20\%$, the proposals needs to be limited to < 5. This leads to a reduction in the number of proposals, thus, improvement in efficiency. However, it may also restrain the traversability of the walker.
- Adding moderated random noise ϵ to the degree $(|\mathcal{F}(u_k)| \pm \epsilon)/p$. The noise can be adjusted w.r.t. the node degree $|\mathcal{F}(u_k)|$ and the number of proposals $|Pr(u_{k+1})|$. As mentioned earlier, with one proposal, the candidate has 50% estimation error, which is sufficiently high. When the number of proposal increases, the noise can be introduced and increased accordingly to compensate the decreasing estimation error. The final estimation error FE on the node degree is calculated as follows $FE = EE_n + EE_p + EE_n.EE_p$, where EE_n the error caused by the noise and EE_p the error caused by the random variable p (more details on this equation can be found in Appendix B). For example, in case of $|Pr(u_{k+1})| = 5$, we know that $EE_p \approx 0.2$ (see Table 4.4). Thus, adding noise $\epsilon = 0.3 * |\mathcal{F}(u_k)|$, i.e., $EE_n = 0.3$ will lead to a final estimation error of $FE \approx 0.3 + 0.2 + 0.05 \approx 0.55$. Adding noise may slightly change the behavior of MHARW in regard to the bias towards highly connected users.
- Other verification methods can be considered, such as zero-proof knowledge or order-preserving cryptography. These approaches provide formally proven guarantees. Nevertheless, the computation overhead of such methods remains a major drawback.

Datasets	*Flickr*	*Livejournal*	*Orkut*
Avg. node visits	2.8	3.9	0.58
Avg. node degree	13.18	14.78	72.75
Avg. number of proposals	0.21	0.26	0.008

Table 4.5: Estimations of the number of proposals based on the number of visits.

4.5.5.3 *Unlinkability*

Here, we discuss the ability of an adversary to link a user with their set of items. In our approach, the collected data contains only the items of a subset of users. For the adversary to link a set of items within the collected data with a particular user, they need to know first whether this user contributed their data to the sample or not. Given a sample size and user population, an external adversary can guess if a particular user contributes their data with probability equals the sample rate, which can be considerably low as we saw before (e.g., 1%). For an internal adversary (curious user), the distributed nature of the traversal-based sampling method MHARW limits their knowledge to local, i.e, the adversary only knows their direct neighbors (predecessor and successor in the walk), which are likely trusted since they are considered friends. With a contributing probability $p_{co} = 0.5$, a curious user can only randomly guess whether their predecessor and successor users have participated in the sample. Furthermore, users can easily detect and avoid risky situations, e.g., predecessor and successor being the same, by not contributing their items as described in Section 4.4.1.

Assuming the adversary knows that a particular user u_i has contributed their data to the sample, the ability of the adversary to link the user with their items is inversely correlated with the size of the sample. The smaller the sample, the better the adversary can guess the items of the user. If the predecessor and successor of a target user are colluding, they can identify the items of that user.

After sampling, each prime user has a sub-sample, which contains the items of a set of users. In the DFP approach, we aim to not disclose the complete sub-sample of a prime user to others. To achieve that in the first phase of the algorithm, we use the secure sum [133] to calculate the support of all items (itemsets of size 1). This technique is secure as long as there is no collusion between the prime users because they cannot distinguish the support values of each other from the random number. Clearly, if the predecessor and successor of a victim prime user in the ring are colluding, they can calculate the itemsets of the victim by subtracting the support values at the successor from those at the predecessor. This risk can be mitigated by splitting each support value into several segments and performing the secure sum for each segment with a permuted order of the prime users [21].

In the partitioning phase (see Figure 4.4), prime users share their sub-sampled itemsets with each other. However, the itemsets are divided and distributed among the corresponding users according to the G-list. This distribution prevents the receiver from learning whether the itemset is complete or it is a subset of an itemset. However, the collusion of multiple prime users might enable them to collect the itemsets of a particular prime user. This can be alleviated by applying probabilistic forwarding similar to an anonymous communication technique proposed in [52], where each user sends their sub-sampled itemsets to a random user with a certain probability, to be forwarded later to the corresponding user.

4.6 CONCLUSION

ARM is an effective algorithm to provide recommendations for users in OSNs. In this chapter, we introduced an approach to mine frequent itemsets—the main phase of ARM—from user distributed data. Our approach is based on a combination of distributed sampling and mining algorithms. First, we proposed a privacy-enhanced version of Metropolis-Hasting Random Walk sampling method, which was leveraged to collect samples of user data. Then, we applied the distributed FP-Growth algorithm under a privacy-aware setting. We evaluated our approach on three large-scale, real-world OSN datasets. Results showed that users can collaboratively mine very high-quality frequent itemsets while maintaining the decentralized nature and privacy advantages of their network. The quality of the frequent itemsets is tightly related to the connectivity of the network and the data distribution. In well-connected networks, the approach achieves high average precision scores ($\geqslant 0.96$) for as low as 1% sample size with a remarkable reduction in communication and computational costs. In networks with highly sparse data, our approach outperforms the unbiased uniform node sampling for small sample sizes. In comparison to other traversal-based sampling mechanisms, our MHARW approach achieves a better quality of frequent itemsets while also reducing the size of messages. Our combination of sampling and privacy-enhanced FIs mining provides a privacy improvement compared to the traditional centralized mining approaches. Compared to the state-of-the-art distributed privacy-preserving ARM, i.e., cryptography-based approaches, we have achieved unlinkability with a scalable approach that incurs a reasonable overhead, making it a suitable solution for the application at hand, OSNs.

Besides ARM, there are plenty of data mining and machine learning algorithms that are used to generate recommendations. Neural networks are one of the prominent algorithms in this field. The next chapter elaborates on training neural networks in a distributed manner.

5 PRIVACY ENHANCEMENTS ON FEDERATED LEARNING

The last chapter introduced an approach to mine frequent itemsets from distributed data while maintaining user privacy. Frequent itemsets are used to improve social network services by generating recommendations. In this chapter, we move on to one of the main components in the state-of-the-art recommender systems, namely neural networks [48]. We focus on training neural networks in a distributed fashion via the emerging technique, Federated Learning (FL). In particular, we elaborate on improving the privacy aspect of FL by adapting its underlying architecture to be more distributed. This in turn makes FL more applicable to user data in DOSNs and HOSNs.

5.1 INTRODUCTION

In recent years, data breaches that violate the privacy of millions of users of online services have increased dramatically [174]. At the same time, more services are making use of large amounts of personal data as part of machine learning implementations to provide value to users. These two factors have contributed to making users of services increasingly concerned about their privacy. The concept of FL has been proposed partially to alleviate this issue by allowing multiple users to build a joint model without sharing their data, under the coordination of a central server [175]. The users train the joint model locally and share only its updates, while the server collects and aggregates these updates to obtain the enhanced joint model. However, the distributed training process in FL, together with its strong dependence on a central server, have increased the attack surface against users yet again [132]. Fully decentralized learning based on a P2P topology has been proposed to eliminate the role of the central server by putting more control in the hands of users, thus enhancing their privacy. Additionally, fully decentralized learning copes with performance and reliability issues associated with having a central server, such as a performance bottleneck and a single point of failure [255]. However, the convergence time and robustness against user churn remain open challenges.

A hierarchical architecture lies in between the two extremes, centralized and fully decentralized architectures, as a solution able to cope with complexity, scalability, and failure-related issues [124]. Applying FL based on a hierarchical architecture is referred to as Hierarchical Federated Learning (HFL). HFL naturally matches emerging decentralized infrastructures (e.g., edge and fog computing) and heterogeneous nature of real-world systems [153] (see Section 5.4). Most importantly, HFL provides a unique capacity for improving privacy.

5.1.1 *Summary of contributions*

In this work, we identify privacy-related issues in FL that are aggravated by its centralized architecture. We then explore and discuss the advantages of using HFL as a possible remedy for these issues. Our discussion suggests that (1) HFL can reduce the centralization of power and control in the hands of the central server, (2) HFL allows flexible placement of defense and verification methods within the hierarchy and enables these methods to be applied more efficiently and effectively, and (3) HFL creates the possibility to employ the trust between users to mitigate a number of threats. The content of this chapter is based on the following publication.

PUBLICATION

Wainakh, A., Guinea, A. S., Grube, T., & Mühlhäuser, M. (2020, September). Enhancing privacy via hierarchical federated learning. In *European Symposium on Security and Privacy Workshops (EuroS&PW) (pp. 344-347). IEEE.*

Contribution Statement
In this paper, I was the main author and contributed all the original concepts. Alejandro Sanchez Guinea and Tim Grube contributed helpful mentoring and comments during the writing process. Max Mühlhäuser provided helpful observations that helped to point out and address several issues.

5.1.2 *Outline*

The rest of this chapter is organized as follows. Section 5.2 provides a background on neural networks and FL. In Section 5.3, we discuss privacy-related issues in FL and then the potential implications of HFL on these issues in Section 5.4. Finally, we wrap up with a conclusion section.

5.2 BACKGROUND

In this section, we present the fundamentals of Neural Networks and FL.

5.2.1 *Neural networks*

Neural networks are a subset of machine learning algorithms; a network is comprised of layers of nodes (neurons) including an input layer, one or more hidden layers, and an output layer. The neurons are connected by links associated with weights $\boldsymbol{W}$. The model can be used for a variety of tasks, e.g., regression analysis, classification, and clustering. In the case of classification, for example, the task of the model $\hat{f}$ is to approximate the function $f(\boldsymbol{x}) = y$ where y is the class label of a multidimensional data sample $\boldsymbol{x}$, e.g., an image—matrix of pixels.

To fulfill this task, the model is trained by optimizing the weights $\boldsymbol{W}$ using a loss function l and training data consisting of input data $x_i : i \in [1, N]$ and corresponding labels y_i in order to solve [85]

$$\min_{\boldsymbol{W}} \sum_{i=1}^{N} l_{\boldsymbol{W}}(x_i, y_i)\,. \tag{5.1}$$

Minimizing the loss function can be achieved by applying an optimization algorithm suitable for the problem at hand. Gradient descent is one of the basic optimization algorithms for finding a local minimum of a differentiable function. This algorithm is based on gradients $\nabla\boldsymbol{W}$, which are the derivative of the loss function w.r.t. the model weights $\boldsymbol{W}$. The core idea is to update the weights through repeated steps t in the opposite direction of the gradient because this is the direction of steepest descent.

$$\boldsymbol{W}(t+1) = \boldsymbol{W}(t) - \eta\nabla\boldsymbol{W}\,, \tag{5.2}$$

where η is the learning rate, which defines the step size for the model updates in the parameter space. An extension of gradient descent, called Minibatch Stochastic Gradient Descent is widely used for training neural networks. This algorithm takes a batch of data samples from the training dataset to compute gradients $\nabla\boldsymbol{W}$ and, subsequently, updates the weights. The batch size B is the number of data samples given to the network for each weight update. The number of epochs E is the number of times in which the whole training dataset was passed to the network.

5.2.2 *Federated learning*

Federated Learning (FL) is a machine learning technique that enables a set of users U to collaboratively train a joint model under the coordination of a central server [132]. The training phase consists of several *communication rounds*. For each round t, the server selects a subset of users $|K_t| \ll |U|$ to train the model locally on their devices using their data. In particular, the selected users optimize the model weights $\boldsymbol{W}$ based on the gradients $\nabla\boldsymbol{W}$. These users can take one step of gradient descent (FedSGD [175]) or multiple steps (FedAvg [175] as shown in Algorithm 3) before sharing the weight updates (or gradients $\nabla\boldsymbol{W}$) with the server. The server collects the updates from all the selected users K_t, then calculates their weighted average to obtain the updated global model as follows

$$\boldsymbol{W}_{t+1} = \sum_{k=1}^{|K_t|} \frac{v_k}{v} \boldsymbol{W}_{t+1}^k\,, \tag{5.3}$$

where v_k is the number of data samples of user k, and v is total number of data samples in round t. This process is repeated until the model potentially converges [175]. This setting mitigates a number of privacy risks that are typically associated with conventional machine learning, where all training data should be collected, then used to train a model [132].

Algorithm 3: FederatedAveraging [175]

Data: n_k number of data samples of user k, n total number of data samples in one round, and η learning rate.

Server executes:
initialize W_0
for *each communication round* $t = 1, 2, \ldots$ **do**
  $K_t \leftarrow$ (a random subset of users)
  for *each user* $k \in K_t$ *in parallel* **do**
    $W_{t+1}^k \leftarrow$ **UserUpdate**(k, W_t)
  end
  $W_{t+1} \leftarrow \sum_{k=1}^{|K_t|} \frac{n_k}{n} W_{t+1}^k$
end

User locally executes **UserUpdate**:
for *each local epoch* **do**
  for *each batch* **do**
    $W \leftarrow W - \eta \nabla W$
  end
end
return W to the server

5.3 OPEN ISSUES IN FEDERATED LEARNING

In spite of the various benefits of FL, several privacy-related issues still exist. We identify three of these issues, which arise primarily from the centralized nature of FL. Next, we describe the three issues, namely centralized control, limited verifiability, and constrained defenses.

CENTRALIZED CONTROL. The server in FL plays the role of a central coordinator that performs the following core functions: (1) sampling users, i.e., selecting which users participate in the training process, (2) broadcasting the training algorithm and model, (3) aggregating the model updates, and (4) broadcasting the updated global model. Although such a setting does not impose a large management overhead on users, it places all control on a single party, the server. Thus, the scalability of the system mainly depends on the resources of the server, which potentially represents a performance bottleneck and a single point of failure. Furthermore, concentrating such functions on the server leaves the users with limited or no control over the process. Consequently, the users have to rely on trusting the server to perform all the functions correctly while maintaining the best privacy practices to protect their model updates. In this respect, a malicious server or an adversary who could compromise the server can be a very powerful attacker. Several important attacks against users have been published that leverage this powerful server-based attacker, e.g., by reconstructing users' training data [288, 316].

LIMITED VERIFIABILITY. In FL, the server and users perform several computations locally and share the results with each other. The users share their updates, while the server shares the aggregated model. In order to allow the

server and users to prove to each other that they perform the expected computations correctly and share a legitimate output, the computations need to be verified. Multiple approaches based on zero-knowledge proofs have been proposed to apply this verification [132]. However, these approaches suffer from two main limitations. First, the types of proofs provided are limited (e.g., range proofs [30]). Second, the time required for verification grows typically exponentially with the number of users [31, 296], which renders these approaches unscalable, and thus unsuitable for large-scale applications. Another technique to tackle the verification issue is to use a Trusted Execution Environment (TEE) to perform the computations [132]. However, these environments currently are not widely available on user devices, especially smartphones. In addition, TEEs do not protect against users who train the model with invalid data.

CONSTRAINED DEFENSES. Although FL provides improvements for user privacy, it opens the door for plenty of attacks, which can be applied by both malicious servers and malicious users. In this work, we focus on privacy attacks, where the attacker aims at inferring information about users. These attacks occur in two modes: passive and active.

- *Passive attacks*: The attacker observes the joint model and periodic updates, which can be used to infer information about other users. Shokri et al. [237] introduced a *membership inference* attack using shadow models, which can be performed in FL. Melis et al. [177] proposed a *property inference* attack, leveraging snapshots of the global model. Zhou et al. [316] obtained the training data of a target user from their shared updates (gradients). To mitigate such attacks, several defense techniques can be applied by the server and users. The users can perturb their model updates before sharing them by using one or a combination of the following methods: (1) adding noise (e.g., local differential privacy [176]), (2) sharing only a fraction of the updates [235], or (3) while training the model, using regularization techniques such as dropout [244]. However, these techniques usually lead to a substantial loss of model accuracy. That is, the trade-off between privacy and accuracy needs to be considered [134]. On the server side, protecting user privacy requires breaking the linkability of the individual users with their updates. This can be achieved by a secure computation, which requires a coordinated decision of both the server and users. Several techniques for secure computations have been proposed, e.g, secure aggregation [23] and secure shuffling [20]. Besides the efficiency issues these techniques suffer from, they hinder the server from detecting malicious updates, creating by that an attack surface for malicious users.

- *Active attacks*: The attacker participates in the training process as a user, who maliciously modifies their updates to infer information about other users. Hitaj et al. [115] proposed a *reconstruction* attack where the attacker provokes the target user to overfit the model on their training data. Melis et al. [177] presented a *property inference* attack based on multi-task learning. Nasr et al. [186] used gradient ascent to perform a *membership inference* at-

tack. To detect and mitigate such attacks, several methods can be considered, such as anomaly detectors [233], median-based aggregation [304], trimmed mean [304], and redundancy-based encoding [50]. However, some of these methods require accessing the individual user updates [233], which is impeded by other defense mechanisms, e.g., secure aggregation. In addition, the heavy computations and uncertainty of model convergence remain main limitations [40].

5.4 HIERARCHICAL FEDERATED LEARNING

FL is, by definition, coordinated by a central server [132]. We relax this definition to apply FL within a hierarchical architecture and refer to it as *Hierarchical Federated Learning (HFL)*. A restricted version of HFL, considering only three layers, has been presented in [153, 159, 160]. However, privacy was not considered a prime goal. In HFL, there is one *root server*, connected to multiple *intermediate servers*, which are organized in a tree structure as shown in Figure 5.1. The lowest layer of intermediate servers connects to the users, which are clustered in *user groups* G. The hierarchy can contain multiple layers of intermediate servers and can be unbalanced such that different branches vary in their number of layers. For simplicity, we proceed our explanation assuming a balanced tree. A selected subset of the users K_t^g of each user group $g \in G$ (in Layer 0) send their model updates to a designated intermediate server in the next higher layer to be aggregated. At each intermediate server in Layer 1 (see Figure 5.1), the following aggregation is taking place

$$W_{t+1} = \sum_{k=1}^{|K_t^g|} \frac{\nu_k}{\nu} W_{t+1}^k \,. \tag{5.4}$$

The aggregation process continues in multiple stages (on each layer) towards the root server. Each set of intermediate servers S in Layer 1 send their updated models to the corresponding intermediate server in Layer 2, where the models are aggregated as follows

$$W_{t+1} = \sum_{g=1}^{|S|} \frac{h_g}{h} W_{t+1}^g \,, \tag{5.5}$$

where h_g is the number of data samples in user group g, and h is the total number of data samples from all groups in S. Every two subsequent layers can have a different number of communication (aggregating) rounds before pushing their models to the next higher layer. After aggregation, the global model is forwarded along the hierarchy downwards to the users.

From an abstract perspective, a hierarchy is a hybrid solution between centralized and fully decentralized architectures. On the one hand, a hierarchy can cope with the scalability and single-point-of-failure issues of centralized architectures. That is evidenced by the existence and active operation of hierarchical architectures within complex, large-scale systems such as the Internet [251]. On

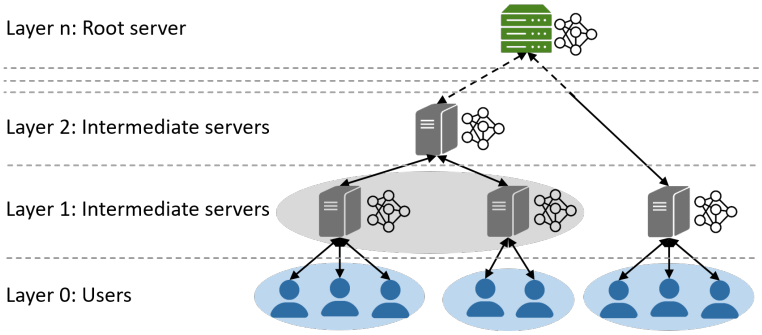


Figure 5.1: Example of hierarchical federated learning architecture†.

the other hand, a hierarchy helps to tackle the management challenges encountered in fully decentralized architectures. We can observe this advantage in the organizational structure of companies [124].

FL was originally proposed with the intention to be based on a cloud computing infrastructure [175]. Such centralized infrastructure represents a network bottleneck, especially with the increase of resource-constrained devices (e.g., smartphones and IoT sensors). To alleviate this issue, there is a trend towards decentralized computing infrastructures (e.g., edge and fog computing) [84], which are introducing hierarchies in communication and computing architectures. These architectures decrease the burden on the cloud by leveraging *edge devices* (e.g., Wifi routers and cellular base stations) to carry out a significant portion of computation, storage, and communication locally. HFL represents a natural step in extending FL into such emerging infrastructures.

Online social networks represent yet another application where the system architecture matches with HFL. Motivated by privacy-related issues, more decentralized social networks are emerging [275]. Diaspora[1] and Mastodon[2] are two of the most popular, which leverage the concept of local servers. Users can choose which local server to connect to and where to store their data. This architecture represents a hierarchy where some functions are delegated to the higher-layer servers. In contrast, privacy-related functions (e.g., data storage) are pushed down in the hierarchy to local servers and users. Similarly, HFL can give lower-layer servers more control over the privacy-related steps in the training process. However, this will introduce some computational overhead for these servers.

5.5 PRIVACY IMPLICATIONS OF HFL

In this section, we discuss the implications of HFL in relation to the privacy-related issues of Section 5.3.

1 https://diasporafoundation.org
2 https://mastodon.social/

5.5.1 *Centralized control.*

In FL, the distribution of functions between the server and users is unbalanced; the control is concentrated on the server. In contrast, HFL allows different FL functions to be distributed throughout the hierarchy to the intermediate servers. Next, we introduce alternative placements for each of the FL functions.

- *Sampling users*: In FL, the server is supposed to select randomly a subset of users to participate in each training round [175]. Due to the large user population and random selection, it is unlikely that an individual user is selected in many training rounds in a given time frame. This, in turn, moderates the amount of data used by that user to train the model, thus improving their privacy. However, a malicious server could misuse this sampling privilege and deliberately selects specific users in consecutive rounds. This allows the server to reconstruct users' private data by analyzing their periodic updates or isolating their shared models [288]. HFL allows sampling to be performed in multiple stages, where a lower layer is sampled by the next higher one, till the top. Thus, the (root) server does not sample individual users, but intermediate servers, where only the aggregate of updates are preserved not the updates of individual users. Consequently, applying the aforementioned attacks is much more challenging in the higher layers of the hierarchy. This can cope with the gap of trust between the users and the server in FL. In HFL, the lowest layer of the intermediate servers need to perform the user sampling. We argue that each two subsequent layers in HFL are more likely to have a better base for trust and accountability. Therefore, it is less likely for users to be attacked by the lowest layer of the intermediate servers. In fog computing, users can be sampled by edge devices, which can be, for instance, a router in the neighborhood. In decentralized social networks, the local server, which is chosen by the users, can perform sampling.

- *Broadcasting the training algorithm and model*: Congzheng et al. [242] showed that malicious training algorithms can leak information about the training data. In FL, users can either fully trust the server to send a benign algorithm or inspect the algorithm locally. Detecting that the algorithm is attempting to leak information about the user data is challenging and introduces additional computational overhead on users' devices. Therefore, this task can be delegated in HFL to trusted intermediate servers, which can be seen as a protective layer where the algorithms are inspected and verified before being pushed to the users.

- *Aggregating the model updates*: In FL, the server collects the updates from individual users and aggregate these updates. The update of an individual user can leak information about them. To protect this update from the server and external adversaries, secure computation techniques such as secure aggregation [23] are used. The application of these techniques on a large scale results in additional computing power and communication requirements. As opposed to FL, in HFL, multiple levels of aggregation occur according to the number of layers in the hierarchy. Each layer collects and aggregates the up-

dates from the lower layer. This aggregation scheme implies that only the lowest layer of intermediate servers receives individual updates from users, while all the higher layers process only aggregated updates. Consequently, the need for secure computations in the upper layers is reduced, which can be interpreted as a performance improvement. In addition, the cascade aggregation process helps to protect the identities of the users from the higher-layer servers.

- *Broadcasting the updated global model*: The server distributes directly the global model to all users. If this model is poisoned, it can easily reach and affect the users. The model can be poisoned by a malicious server or user. Hitaji et al. [115] showed that a malicious user can poison the global model so that certain users (victims) reveal information about themselves by overfitting the model to their data. The leaked information then is extracted by the malicious user from the updated global model in following training rounds. The flat topology at the users level in FL allows this type of attack easy access, while HFL provides the possibility to control the model dissemination more effectively through the multiple layers. The intermediate servers can be considered as inspection stations where the model is checked. If the root or a intermediate server is malicious and distributes a poisoned model, the intermediate servers in the lower layers could refrain from pass it further to the users. The existence of a malicious user poisoning the model for one user group (a branch of the hierarchy) does not directly affect other groups, unless the poisoned model was successfully passed up the hierarchy via intermediate servers and propagated downwards again.

5.5.2 *Limited verifiability.*

The poor scalability of the verification methods in FL (e.g., [31, 296]) hinders their deployment in large-scale applications. In HFL, verifying the updates of the users can be performed by the corresponding intermediate servers. Grouping users in comparably small groups reduces the computational overhead on the intermediate servers and makes the deployment of verification methods more feasible.

5.5.3 *Constrained defenses.*

Most of the defense methods in FL come with the cost of lower model accuracy [235], considerable computational overhead [304], or both [50]. HFL provides the possibility to apply these methods in a more flexible manner. One possibility is to apply different methods in different parts of the hierarchy, such that not all the users suffer from a heavy loss in accuracy nor have to perform costly computations.

As the upper layers in HFL process only aggregated models, not individual users' updates, the necessity of secure aggregation can be reduced or even eliminated in these layers. This can lead to a performance improvement and

allows intermediate servers in higher layers to apply anomaly detection methods (e.g., [233], see Section 5.3) to detect malicious models received from lower layers. Detecting a malicious model from a specific intermediate server can be followed by excluding that server from further aggregation rounds while maintaining the training process functioning normally in the rest of the hierarchy. Furthermore, the affected server can be notified about its malicious model so it can take more computational expensive countermeasures (e.g., [304]) in their local user group.

HFL allows leveraging the trust between users as an additional line of defense. In social networks, grouping users based on trusted graphs [128] can reduce the probability of attacks within groups, as it is assumed to be more difficult for malicious users to break into the groups through social engineering. Consequently, users may relax their local defenses (e.g., reduce the noise added locally) to achieve a model with higher accuracy. At the same time, the intermediate servers can take the responsibility of protecting their groups from attacks across groups by applying specific defense methods (e.g., adding noise to the aggregated updates).

5.5.4 *Summary of advantages*

The potential advantages of HFL that we discussed earlier are based on specific assumptions mainly regarding trust within the hierarchy. In Table 5.1, we summarize these advantages and the corresponding assumptions.

5.6 CONCLUSION

On the one hand, FL allows users to maintain their data local, which can be seen as a privacy advantage. On the other hand, FL assigns all the coordination operations to a central server, which must be trustworthy for the users if privacy shall be ensured. In this chapter, we presented the privacy-related issues stem from the centralized scheme of FL, namely centralized control, limited verifiability, and constrained defenses. Then, we discussed how Hierarchical Federated Learning (HFL) can potentially alleviate these issues. HFL enables high flexibility in functionality distribution through the hierarchy. This in turn facilitates the applications of known defense and verification methods. HFL allows leveraging the trust between the participating users in different application scenarios to mitigate several threats. Overall, our work contributed to the research on identifying the privacy-related issues in FL and highlighted potential mitigations by applying FL a hierarchical architecture. By that, we ultimately help increase the robustness of FL against privacy risks.

This and the previous chapter focused on applying two analytics techniques in a distributed manner. The literature showed that while applying such techniques several attacks might threaten user privacy. The next two chapters further investigate threats to distributed analytics, in particular FL.

	Advantage	Assumption
Control	Reducing the centralization of control by delegating functions, namely user sampling and user updates aggregation to intermediate servers	Trusted intermediate servers (Layer 1)
	Inspecting the training algorithm by intermediate servers before propagating to users	Trusted intermediate servers (Layer 1)
	Inspecting the global model by intermediate servers before propagating to users	Trusted intermediate servers (Layer 1)
Verification	Reasonable overhead of applying verification methods to user updates by intermediate servers	User groups are relatively small && trusted intermediate servers (Layer 1)
Defense	Flexible application of different defenses on parts of the hierarchy	-
	Isolating malicious/compromised parts of the hierarchy	-
	Reducing the need to the costly secure computations in higher layers	-
	Applying anomaly detection methods against poisoning in higher layers	Trusted intermediate servers
	Reducing the need for local obfuscation defenses on user side	Users are trusted within their groups && trusted intermediate servers (Layer 1)
	Delegating the defense application to intermediate servers (Layer 1) to protect their user groups	Trusted intermediate servers (Layer 1)

Table 5.1: Overview of HFL advantages and assumptions.

Part IV

THREATS TO DISTRIBUTED ANALYTICS

6

SYSTEMATIC ANALYSIS OF THREATS IN FEDERATED LEARNING

In the previous chapters, we elaborated on applying analytics in a distributed setting; in particular, we investigated distributed ARM and Federated Learning (FL) for neural networks. Although the distributed setting delivers privacy advantages by granting users more control over their data, the distributed data processing introduces at the same time novel security and privacy threats. A comprehensive overview of these threats does not exist yet (in the form necessary), especially for the rapidly emerging technique FL. Therefore, in this chapter, we explore and systematize the attacks presented in the literature against FL. Additionally, we discuss the applicability of these attacks considering real-world scenarios.

6.1 INTRODUCTION

The distributed nature of the training process in FL has created a new attack surface. As we will discuss in more detail in this chapter, threats come, for instance, from malicious users who actively participate and can adversely affect the training process. Therefore, they are able to threaten, for instance, the integrity of the aggregated model (e.g., poisoning attacks [11, 18]), or the confidentiality of the users' personal data or personalized models (e.g., model inversion attacks [114, 316]). Recently, researchers have been extensively investigating potential vulnerabilities and attacks in FL. This led to an increasing number of publications on attacks against FL, consequently, raising serious concerns about the robustness and privacy in FL. Some strong claims even exacerbate these concerns by stating that "federated learning is fundamentally broken" [114], or that some attacks are reaching 100% accuracy in poisoning the model [11].

Upon taking a closer look, however, the situation may not be as bad as proclaimed. A number of these attacks are applicable only under specific conditions and assumptions. For example, some attacks use the batch size of 1 for training a Neural Network (NN) model (e.g., [313]), or assume a special distribution of data among users (e.g., [114]). In many cases, such assumptions do not hold in real-world deployments. Thus, the applicability of such attacks is questionable. In addition, several attacks are evaluated with limited or impractical setups. For instance, some attacks are evaluated using oversimplified datasets (e.g., [309]) or simplified NN models (e.g., [70]). This in turn affects the generalizability of the experiments, results, and conclusions. A recent work by Shejwalkar et al. [231] fueled this discussion by demonstrating that, contrary to the common belief, FL is highly robust against several attacks in the literature under practical considerations—even without applying any defenses. Considering the aforementioned issues, the severity of the threats posed by the

attacks discussed in the literature becomes imprecise and debatable. Therefore, it is essential to closely and systematically investigate such issues. As a result, a more comprehensive and realistic view of the significance of threats in FL can be obtained.

6.1.1 *Summary of contributions*

In this chapter, we conduct a qualitative and quantitative analysis of publications about attacks against FL via a Systematic Mapping Study (SMS). We first identify research trends in the field, the properties of the research community such as affiliations, and targeted publication venues. Then, we provide a structured overview of the attacks with two classification schemes that are based on: (1) the properties of the attacks, (2) the choice of experimental setups used to evaluate the attacks. We analyze the distribution of publications among the defined attack classes and derive the foci and gaps in the research landscape. Next, we highlight several special assumptions made in some of the works and their implications on the applicability of the attacks. Finally, we identify common fallacies in the evaluation setups and the impact of these fallacies on the validity and generalizability of the results. Our work shows that each of the studied papers at least makes one of the special assumptions or suffers from one fallacy. Notably, several fallacies affect the majority of the papers. The contributions of this chapter can be summarized as follows.

- Providing a comprehensive quantitative analysis of all 44 relevant papers on attacks against FL from 2016 to the first quarter of 2021.
- Identifying four research gaps that put the the reviewed literature into perspective regarding the effectiveness of contributed attacks in practice.
- Highlighting three recurring assumptions made in the reviewed papers and limiting the applicability of the proposed attacks to real-world deployments. These assumptions are related to the hyper-parameters of the Machine Learning (ML) model, the fraction of malicious users, and data distribution.
- Identifying six fallacies in the evaluation practices that can cause overestimation of the attacks' effectiveness. The main fallacies stem from the choice of: datasets, models, and the size of the user population. In addition, we also propose a set of recommendations to mitigate these fallacies.

REMARK

This study does not undermine the importance of the research or the findings on attacks in FL, rather it is an attempt to help researchers clarify the scope of these attacks by reflecting on the assumptions and evaluation practices.

The content of this chapter is based on the following paper.

PUBLICATION

Wainakh, A., Zimmer, E., Subedi, S., Keim, J., Grube, T., Karuppayah, S., Guinea, A. S., & Mühlhäuser, M. (2022). Federated Learning Attacks Revisited: A Critical Discussion of Gaps, Assumptions, and Evaluation Setups. In *IEEE Access [under review]*.

Contribution Statement
In this work, I led the process of idea generation, study conduction, results analysis, and writing. Ephraim Zimmer, Tim Grube, Shankar Karuppayah, and Alejandro Sanchez Guinea contributed with helpful discussions and formulations to enhance the editorial quality of the manuscript. Sandeep Subedi and Jens Keim contributed to the conduct of the study and the analysis of the results. Insightful discussions with Max Mühlhäuser helped to improve several aspects of the work.

6.1.2 *Outline*

The remainder of this chapter is organized as follows. In Section 6.2, we present a summary of related reviews on the literature. Then, in Section 6.3 we elaborate on the methodology of our study. In Section 6.4, we present the results of the mapping process. Subsequently, we further analyze our results and discuss their implications in Section 6.5.

6.2 RELATED WORK

Several studies in the literature provide overviews of the privacy and security issues in FL, either as a part of general analysis of ML systems or as a specific analysis of the FL setting.

Privacy and Security in ML. Due to the growing recognition of the threats that ML systems might face, many researchers presented surveys and systematization of these threats, such as Papernot et al. [203] Al-Rubaie et al. [219], De Cristofaro [55], Rigaki et al. [218], and Zhang et al. [308]. These studies mainly focus on the privacy aspect of centralized training and only address FL to a limited extent. Other surveys specifically tackle the privacy and security of deep learning models, namely the works of Mirshghallah et al. [179] and Liu et al. [161].

Privacy and Security in FL. In our study, we aim to (1) structure all the publications dealing with privacy and security FL attacks according to classification schemes, thus providing a *structured* and *comprehensive* overview of the research field, (2) conduct a *quantitative analysis* of the publications, highlighting ares of focus and gaps in the literature, and (3) provide a critical discussion of the *applicability* of the proposed attacks by taking a closer look at their assumptions and evaluation setups.

There are studies in the literature with partially overlapping goals. Enthoven et al. [69] presented a structured overview about attacks and defense mechanisms in FL, but only for privacy attacks against deep learning models. Lyu et al. [169] elaborated additionally on security attacks and pointed out weaknesses in current countermeasures through a qualitative analysis of the literature. However, this study neither provides a quantitative analysis nor discusses the applicability of the attacks. A concise taxonomy of attacks in FL was introduced by Jere et al. [127]. Although the proposed taxonomy is well thought out and justified, the study considers only a small portion of the attacks available in the literature without discussing their quantity or applicability. Kairouz et al. [132] presented an extensive report of open issues and challenges in FL, including privacy and security issues. However, the extent to which these issues are applicable in real-world scenarios is discussed only briefly. Similar but less comprehensive surveys in terms of the level of detail were also published [5, 150].

The aforementioned studies provide very valuable insights into the privacy and security in ML and FL by summarizing and systematizing the existing research in this field. However, none of them meet all of our goals, and all of them have (largely) failed to meet two of our main goals, namely quantitative analysis and discussion of attack applicability.

6.3 METHOD

Systematic mapping [208, 209] is a secondary study method that establishes classification schemes and structures in a research field. The analysis of the study focuses on the frequency of publications in each of the defined categories. Such analysis provides valuable insights into the progress, foci, and gaps in the research field. These insights are usually not covered by the commonly used systematic literature review method, which focuses on surveying primary studies to collect evidence concerning existing solutions [136], while overlooking the frequency of publications. In the context of this thesis, an SMS helps us to direct our further research towards contributions that fill the existing gaps. To conduct an SMS, the following steps are taken:

1 Define a set of guiding questions to be answered by the analysis of the study.

2 Conduct a search to find the relevant papers.

3 Refine the selection of papers by following inclusion and exclusion criteria.

4 Define classification schemes to structure the papers into categories.

5 Map the papers to the defined categories.

6 Answer the guiding questions by analyzing the frequency of papers appearing in the defined categories.

Petersen et al. [208, 209] proposed a guideline for conducting SMSs, which serves as a basis for our study outlined in this section.

Database	Papers	Search string
Google Scholar	537	("federated learning" OR "collaborative learning") AND ("inference attack" OR "privacy attack" OR "poisoning attack")
ACM Digital Library	21	
IEEEXplore	81	
arXiv	97	

Table 6.1: Automatic search results and search strings†.

6.3.1 *Objectives and guiding questions*

The ultimate goal of our study is to analyze past research and to guide further research on attacks for FL w.r.t. practical relevance. This concerns, in particular, the effectiveness of attacks, the question of how realistic the assumptions are, and fallacies observed. As a basis, we analyzed the attacks proposed in 44 scientific papers identified as pertinent for the SMS (see Sections 6.3.2 and 6.3.3). In light of the general goals of the study as set forth in Section 6.2, we base our systematic work on the following three guiding questions.

1 What are the research trends in the domain? For this question, we look at the development of the number of papers over years, the communities that conduct the research, and the type of the research.

2 What are the different types of attacks carried out against FL? We identify the attacks that have been proposed in FL and their properties.

3 Which are the evaluation setups commonly used in the literature? We determine the common evaluation practices in the field and discuss their implications.

Answering the aforementioned questions requires two sub-steps: (1) identifying and capturing the research trends, attack types, and evaluation setups in terms of categories and characteristics, (2) analyzing the literature and mapping it according to the newly-established categories and characteristics of the respective research trends, attack types, and evaluation setups.

6.3.2 *Search strategy*

To set up the main search process, a brief pilot search (pre1 in Figure 6.1) was carried out, where an initial set of relevant papers was collected. These papers allowed us to determine relevant keywords and search terms as well as suitable venues to target in the main search process, and to identify a set of well-known authors in this research area. For this pilot search, Google Scholar was used since it is one of the indexing databases that cover a large number of publishers. Several search queries were applied using combinations of keywords, which have shown to be important in the research field at hand, including "federated learning", "privacy attack", and "security attack". In the following, more details about the main search phase are elaborated, which consists of two main steps: automatic and manual search.

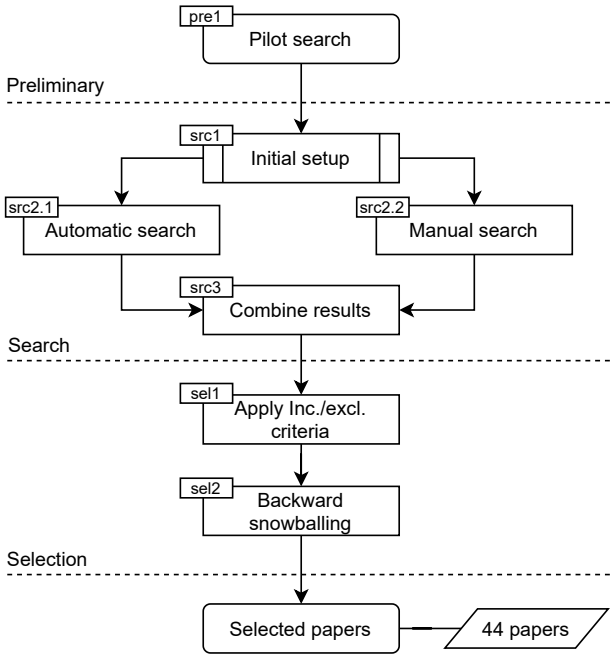


Figure 6.1: Search and selection process of our SMS. We start with a pilot search as a preliminary step. Then, we conduct the main automatic and manual search. Lastly, we select the relevant papers†.

Type	Abbreviation	Name
Journal	TOPS	Transactions on Privacy and Security
	TIFS	Transactions on Information Forensics and Security
Conference	S&P	Symposium on Security and Privacy
	CCS	Conference on Computer and Communications Security
	USENIX	USENIX Security Symposium
	PETS	Privacy Enhancing Technologies Symposium
	EuroS&P	European Symposium on Security and Privacy
	NDSS	Network and Distributed System Security Symposium
	CSF	Computer Security Foundations Symposium
	ACSAC	Annual Computer Security Applications Conference
	ESORICS	European Symposium on Research in Computer Security
	NeurIPS	Neural Information Processing Systems
	ICML	International Conference on Machine Learning
	ICLR	International Conference on Learning Representations
	InfoCom	International Conference on Computer Communications
	AISTATS	Artificial Intelligence and Statistics

Table 6.2: Manual search sources†.

1 Automatic Search (src2.1 in Figure 6.1): We conducted the automatic search by relying on several popular search engines, namely ACM Digital Library, IEEEXplore, Google Scholar, and arXiv. ACM Digital Library and IEEEXplore were considered as they cover the key research communities (i.e., ML and security communities) and the most cited publications from ACM and IEEE computer society. In addition, Google Scholar was used to ensure comprehensive results and avoid any bias towards specific publishers. Furthermore,

using arXiv helped to cover the most recent advancements, which are not yet accepted for publication. The keywords to include FL-related terms were: "federated learning" and "collaborative learning". Additionally, precise keywords related to attacks, namely "inference attack", "privacy attack", and "poisoning attack" were also added to the search. Subsequently, a search string was composed using the updated keywords. The results of the automatic search are shown in Table 6.1.

2 Manual Search (src2.2 in Figure 6.1): The titles of the papers published in a set of selected journals and conferences were manually reviewed. The journals and conferences were chosen based on the results of the pilot search to cover all the venues where papers on attacks in FL are published. The complete list of sources is shown in Table 6.2. As a complementary procedure, a number of well-known researchers in the field (e.g., H. Brendan McMahan) were identified and their publications (on Google Scholar, private webpages, and university webpages) were tracked. The manual search resulted in identifying 20 potentially relevant articles.

The combined total number of papers gathered from the automatic and manual searches is 756. These papers were found to contain our search strings or to have relevant titles. However, the focused search still yielded many papers that are not relevant to our study. Therefore, it was crucial to specify strict criteria to select the relevant papers among them.

6.3.3 *Selection process*

As depicted in Figure 6.1, after the search, the selection process was conducted. This process consists of two steps: (1) applying inclusion and exclusion criteria and (2) performing a complementary forward and backward snowballing search (see below). The following inclusion and exclusion criteria were applied to titles and abstracts. In those cases, where the title and abstract do not provide enough information, the body of the paper was considered.

- *Inclusion criteria*: (1) Papers discuss attacks in FL. These papers include those that introduce novel attacks as well as papers that review existing attacks. (2) Papers published between 2016 (the year of coining the term *Federated Learning* by McMahan et al. [175]) and the first quarter of 2021 (the time of conducting this study).
- *Exclusion criteria*: (1) Papers about FL that do not cover any attack. (2) Papers not accessible in full-text. (3) Duplicate papers.

After filtering the papers based on the inclusion and exclusion criteria, forward and backward snowballing techniques were applied. The forward snowballing technique identifies papers that have cited the papers found in the search phase. As the majority of the selected papers after filtering is quite recent (2019-2020), they have not yet been cited by many other papers. Therefore, the focus was more on backward snowballing (sel2 in Figure 6.1). In this technique,

the lists of references in the selected papers were reviewed and the relevant papers were added to our list. Applying this technique resulted in adding only a few more new papers, as our automatic and manual searches had already covered almost all the relevant sources.

The final number of papers considered for our analysis is 44. It is worth mentioning that after applying the inclusion and exclusion criteria, there was a remarkable drop in the number of papers. That is due to the fact that there is a huge number of papers that refer to FL and its attacks, but do not contribute to this topic.

6.3.4 *Information extraction and classification*

Each of the final selected papers was examined in detail to extract information on the research trends, attack types, and evaluation setups. This information was utilized to (1) propose an initial set of classification schemes and (2) sort the papers accordingly. Then, we iterated, after a first orientation path, over these two steps multiple times to refine the classification schemes and remap the papers to the defined categories. In the following section, three main classification schemes are presented that correspond to our guiding questions.

6.4 RESULTS OF THE MAPPING

In this section, we elaborate on the classification schemes and the results of paper mapping. The frequency of papers in each category is presented as the exact number of papers and the percentage w.r.t. the total papers number 44. The results are also provided in detail in Tables 6.4 and 6.5. This section is structured along with the previously established guiding questions.

6.4.1 *Research trends*

We investigate the trends in the research field through four aspects, namely the *year* of publications, the *affiliations* of the researchers, the *venues* they target to publish their works, and lastly, the *type of research* conducted according to Wieringa et al. [291].

YEAR OF PUBLICATION. In Figure 6.2, we see the publication year for the studied papers. The first attack against FL was published in 2017 by Hitaj et al. [114]. In the following years, the number of attacks is remarkably increased. That reflects the growing attention towards FL in general and its privacy and security issues, in particular. This considerable number of attacks can also be seen, for an external observer, as an indication of the abundance of FL vulnerabilities. Overall, FL is a hot topic and the number of its applications is growing, therefore, investigating its weaknesses becomes crucial, and this likely will lead to more studies on attacks in the upcoming years.

AUTHOR AFFILIATIONS. Our mapping study illustrates that most of the attacks (38 i.e. 86%) come from academia. This can be due to the fact that re-

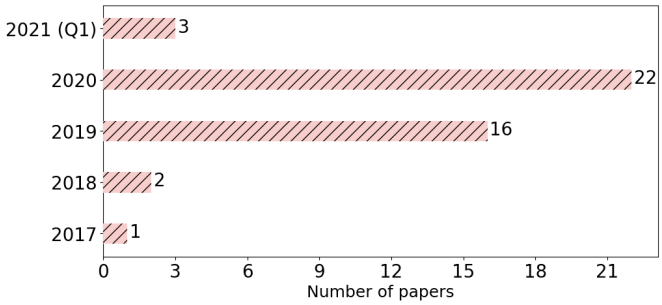


Figure 6.2: Number of papers per year†.

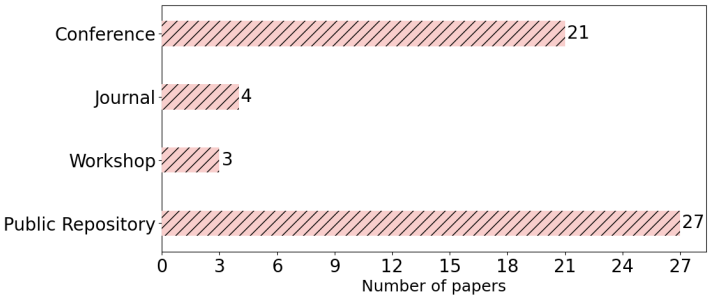


Figure 6.3: Venue types of the selected papers†.

searchers in academia freely explore the possibilities to hack technologies and then propose mitigation measures, while industry tends to focus on making their services more robust and secure. That is evident from the substantial number of papers on defense mechanisms from industry, especially Google [6, 10, 13, 23, 176], whereas a fewer number of attacks (2 i.e. 4%), were proposed by industry. Joint projects between academia and industry can also be found in 4 (9%) papers.

VENUE TYPE. In our study, we took into account peer-reviewed venues (journals, conferences, and workshops) as shown in Table 6.2, in addition to public repositories (arXiv). The paper distribution among these venues is depicted in Figure 6.3. We can see the tendency of the community to push their studies to public repositories, where 27 (61%) of the papers are found. This can be due to the fast pace of publications in this field, which urges researchers to share their ideas and results promptly as preprints. Out of these, 10 (22%) are simultaneously published in a peer-reviewed venue, mainly conferences. After arXiv, the conference papers come first with 21 (47%) papers. The low number of publications in journals might be a result of the novelty of the FL concept and the rapid development of its attacks.

RESEARCH TYPE. To identify the research characteristics in this field, we categorize the papers based on the type of the conducted research. We adopt the research types proposed by Wieringa et al. [291].

- Solution: Proposes an approach to solve a problem. The approach can be novel or an improvement on existing ones. The proposed approach should be supported by good arguments or by other means.

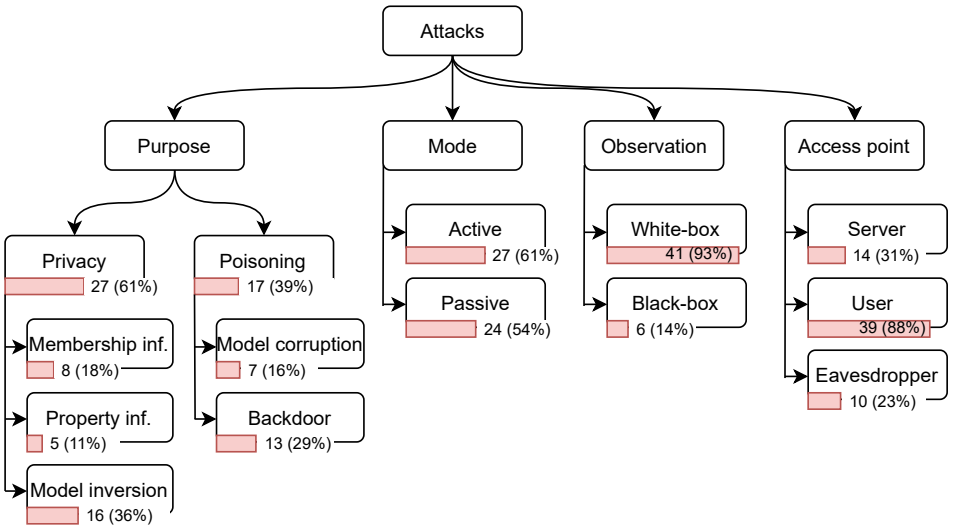


Figure 6.4: Attack classification with the paper distribution. The percentage is w.r.t. the total number of papers 44. Most categories are not exclusive as some papers consider multiple threat models and discuss different forms of the proposed attack, therefore, the papers might sum up to more than 44. More details on the individual papers can be found in Table 6.4†.

- Validation: Investigates the validity of a novel approach that has not yet been "realized". The validation can be performed through experiments, simulations, mathematical proofs, etc.
- Evaluation: Studies the properties of an existing approach (analyze, assess, and evaluate) to achieve a better understanding of its potentials and limitations.
- Philosophical: Provides new insights, a new way of thinking, or a new conceptual view of research.
- Opinion: States the authors' position towards a specific topic without introducing any research results.
- Experience: Describes the personal experience of the authors in conducting "a practice".

Our mapping shows that the studied papers fall into only two categories, namely Solution and Evaluation, with 33 (75%) and 11 (25%), respectively. On the one hand, the novelty of this research field can be a reason for the abundance of papers within the Solution category, since there are many privacy and security aspects that need to be addressed. On the other hand, this novelty may explain the absence of papers from other categories, such as Experience, which typically requires more time to put the research approaches into practice and develop experience in the domain.

6.4.2 *Attack types*

To identify the types and properties of the attacks, we consider several aspects, namely attack's purpose, mode, observation, and access point [55, 187]. Next,

we introduce the common attack categories w.r.t. each of the aspects. The distribution of publications among these categories is depicted in Figure 6.4.

PURPOSE. The attack's purposes can be classified into two main categories.

1 Privacy attacks (inference attacks): These attacks extract information about the training dataset, i.e., user data [55], and fall into three groups based on the obtained information:
   - Membership inference: The adversary aims to determine whether a particular individual (or a data record) belongs to the training dataset [237].
   - Property inference: The adversary aims to infer features of the training dataset, where these features are not intended to be used for the main task of the model [81].
   - Model inversion (attribute inference): The adversary aims to infer sensitive features used as input to the model [114].
2 Poisoning attacks: The adversary maliciously alters the model to achieve one of the following goals.
   - Model corruption (label-flipping): The adversary corrupts the model to reduce its overall accuracy in its main task. This attack can target specific classes or be untargeted [259].
   - Backdoor: The adversary implants a backdoor sub-task in the model while maintaining a good accuracy of the main task. This backdoor is used later in the production phase to exploit the model, e.g., by forcing misclassification of a specific input [11].

Our mapping shows in Figure 6.4 that the majority of the papers focus on privacy attacks with 27 (61%), while 17 (39%) for poisoning attacks. This may be explained by the fact that FL is mainly promoted as a mitigation for several privacy risks [175]. Therefore, many researchers investigate the potentials and limitations of privacy in FL by crafting various attacks. Among the different types of attacks, the ones that dominate the research publications are the model inversion 16 (36%) and backdoor 13 (29%). Model inversion is one of the most severe attacks, since the adversary, in some cases, can fully reconstruct the user data. Backdoors are quite powerful in manipulating the model performance in the production phase, which might leave a long-term impact on the systems.

MODE. An adversary might act in two different modes.

1 Passive: The adversary attempts to learn from the observed information, without interrupting or deviating from the regular training process. This mode is widely common in privacy attacks [309, 316].
2 Active: The adversary acts maliciously in the training process, e.g., they manipulate the training data or model updates. This mode is needed for poisoning attacks [11, 18].

Figure 6.4 shows that 27 (61%) attacks are launched in the active mode, while 24 (54%) are in the passive mode. Out of these attacks, 7 (16%) are applied

in both modes. This distribution can be correlated with the capabilities of the adversary in the two modes, i.e., in the active mode, the adversary is more powerful, thus, a wider variety of attacks can be performed.

OBSERVATION. The adversary's capability to observe the parameters of the target model might vary among different attacks. We consider two possibilities.

1 Black-box: The adversary can query the model, thus, knows the inference result of a particular input. However, they do not observe the model parameters [74].

2 White-box: The adversary can observe the model parameters [287]. This capability typically enables adversaries to carry out more sophisticated attacks.

As the model parameters are typically shared between the server and all the users in FL, most of the attacks 41 (93%) assume the white-box scenario. The black-box is considered only in 6 (14%) attacks.

ACCESS POINT. The adversary might exist at different locations with different roles in the system.

1 Malicious server: In FL, the server coordinates the training process by initializing the model, disseminates the training algorithm, collects and aggregates the updates from users. The server can be malicious and tries to analyze the updates of the users to learn extra information about their data [287].

2 Malicious user: A user can behave adversely during the training to poison the model [18] or to infer information about other users [187]. In FL, the model is sent to the users' devices where the users are typically granted full access to the model parameters. This access privilege, in turn, amplifies the malicious users' capabilities and enables them to perform sophisticated attacks.

3 External eavesdropper: External adversaries that eavesdrop on the communication between the server and users, and thus, have access to the shared gradients, might be able to reconstruct the user data [316].

Figure 6.4 illustrates that 39 (88%) attacks are conducted by users, while only 14 (31%) attacks assume the server to be malicious or curious, and 10 (23%) papers include attacks that can be carried out by external eavesdroppers. This reflects a keen interest in the attacks from the user side. The is can be explained by the fact that users are usually easier to attack and that these attacks are facilitated mainly by the distributed nature of FL.

6.4.3 *Common evaluation setups*

The effectiveness of the proposed attacks is mostly demonstrated through an empirical evaluation. This evaluation needs to be extensive and comprehensive to provide sufficient evidence for the attack validity under specific settings. In

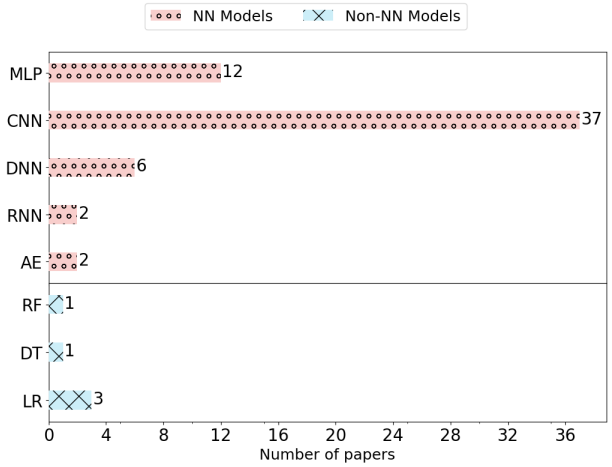


Figure 6.5: Target model types used to evaluate the attacks. In most of the papers, the proposed attacks were evaluated against more than one model. The red bars refer to the NN models, while the bars in blue show other kinds of models†.

this section, we examine the experimental settings commonly used for evaluating FL attacks by looking into four aspects, namely target models, datasets, countermeasures, and implementation technologies.

TARGET MODELS. We refer here to the joined ML model that is trained through the FL process, thus, targeted by the attack. The type of the model can vary, as FL by definition is not restricted to specific types. The attacks might be designed to target one or multiple model types, or they can be completely model-agnostic. On a high level, we can classify the target models in the literature into NN models and non-NN models.

The mapping results reveal that only 3 (6%) attacks target non-NN models. As shown in Figure 6.5, these three attacks consider Logistic Regression (LR) [74, 168], while only one of them targets also Decision Tree (DT), and Random Forest (RF) [168]. Other attacks mainly focus on NN models, which can have diverse architectures. Interestingly, we observe that the model type Convolutional Neural Network (CNN) is dominant as a target model in 37 (84%) attacks. This high attention can be explained by the fact that CNNs are considered the state of the art for a wide variety of computer vision applications [107]. Other NN models such as Recurrent Neural Network (RNN) (e.g., Long Short-Term Memory) and Autoencoder (AE) (e.g., Transformers) were the target of attacks only twice in the literature. Furthermore, one attack is claimed to be model-agnostic [170].

DATASETS. To train the target model, various datasets were used in the literature. These datasets can be categorized into three groups based on the type of the data: Text, image, and key-value pairs. In total, 42 distinct datasets were used in the attack evaluations, and they are:

1 Text: CLiPS Stylometry Investigation [270], Yelp-author [123], Reddit [100], Amazon Review [192].

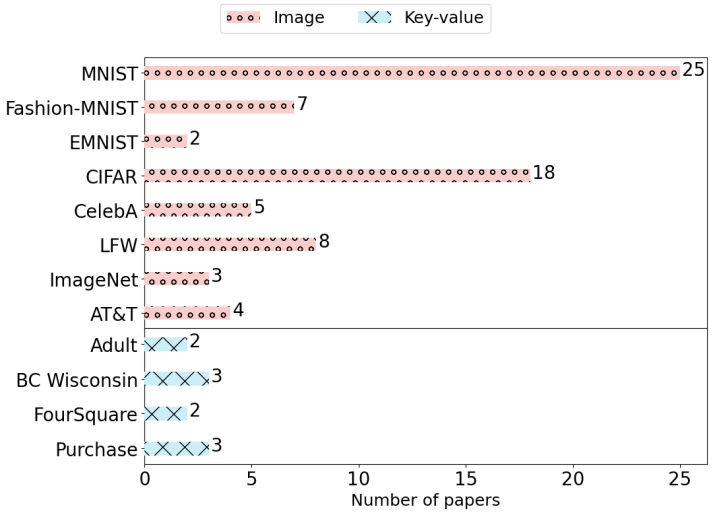


Figure 6.6: Datasets used more than once in the literature to evaluate attacks. In most of the papers, the proposed attacks were evaluated using more than one dataset. The red bars refer to the image datasets, while the bars in blue show the key-value datasets[†].

2 Image: MNIST [145], Fashion-MNIST [294], LFW [245], CelebA [305], AT&T [225], CIFAR [140], CH-MNIST [212], ChestX-ray8 [284], EndAD [211], EMNIST [45], Fer-2013 [92], HAM10000 [262], ImageNet [58], PIPA [311], SVHN [188], PubFig [141], Omniglot [142], mini-ImageNet [271], VGG2Face [34], fMRI [41], CASIA [152], Face [215].

3 Key-value: Purchase [237], BC Wisconsin [64], Adult [64], FourSquare [299], Human Activity Recognition [7], Landmine[252], Texas-100 [237], UNSW-Benign [239], Parkinson Data [155], Yelp-health [119], Bank Marketing [185], Credit Card [87], Drive Diagnosis [64], News Popularity [77], KDDCup99 [112], DIoT [190].

Over 69% of these datasets are used only once in the literature. The more prominent datasets are shown in Figure 6.6, where MNIST and CIFAR are the most common ones, used in 25 (56%) and 18 (40%) attacks, respectively. This conforms also with the common datasets in the ML community [101]. The popularity of MNIST can be due to several reasons, e.g., its small size, which enables researchers to train their models quickly and report results. In addition, MNIST as well as CIFAR are widely supported by many ML frameworks, thus, they can be easily used [294]. The average number of datasets used per paper is 2.6, with a maximum of 7 and a minimum of 1.

COUNTERMEASURES One of the main methods to evaluate the proposed attacks is measuring their effectiveness against the state-of-the-art defense mechanisms. We explored the mechanisms used in the examined papers; they can be classified into three main categories.

1 Perturbation: This mechanism reduces the information leakage about the users in FL by applying one of the following perturbation techniques.

- Noisy updates: A user may add noise to their data [213] or the updates before sending them to the server [316]. The noise can also be added on

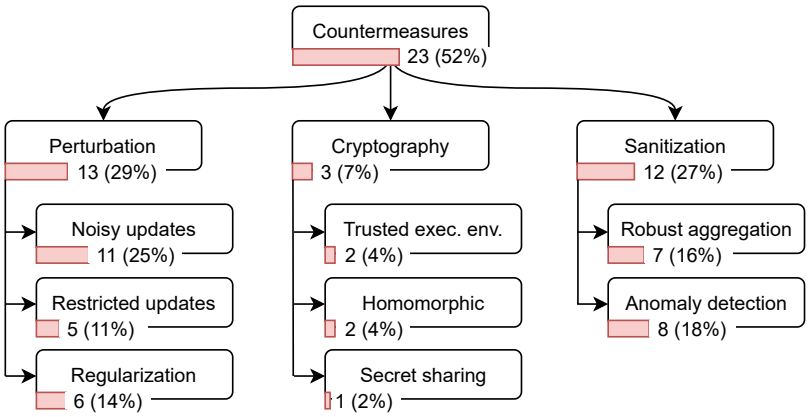


Figure 6.7: Countermeasures classification with the paper distribution. The percentage is w.r.t. the total number of papers 44. Some categories are not exclusive, therefore, the papers might sum up to more than 44. More details on the individual papers can be found in Table 6.5. Only 52% of the attacks are evaluated against countermeasures. Noisy updates is the most used technique with 25%.

the server side [176]. The amount of noise can be carefully specified to achieve *differential privacy*.

- Restricted updates: Before sharing the updates with the server, a user can limit the number of updates [235], or compress the updates, e.g., by applying quantization [137].
- Regularization: While training the model locally on the user's device, the user can apply regularization techniques such as dropout, or batch normalization [261].

2 Cryptographic approaches: exposing the updates of an individual user can lead to severe information leakage about their training data [316]. Several techniques based on cryptography are proposed to mitigate this risk.

- Homomorphic encryption: Users can encrypt their updates with homomorphic encryption before sharing them with the server. Because of the homomorphic property, the server can compute the aggregation of the encrypted updates from all users to obtain an updated and encrypted global model. This model then is shared with the users, who can decrypt it [8].
- Secret sharing: Users can encrypt their updates with keys derived from shared secrets. That is, the server needs to aggregate the encrypted updates and thus the shared secrets from a sufficient number of users in order to be able to decrypt the aggregate [23].
- TEE: The aggregation process on the server can be moved into a TEE, such that the executed code can be attested and verified to not leak individual users' updates [132].

3 Sanitization: This mechanism is proposed to mitigate poisoning attacks. In this respect, two defense mechanisms have been developed in the literature

- Robust aggregation: To limit the impact of malicious updates on the global model, different update aggregation methods (other than the default weighted averaging [175]) are proposed. For instance, the mean of the updates can be trimmed, or instead of the mean, the median of the updates is used as the aggregate to optimize the global model [304].
- Anomaly detection: The malicious updates are usually assumed to be anomalies. To identify the anomalous updates (outliers), various techniques can be used, such as clustering [233] or measuring similarity with a reference set of samples [38].

The mapping results are depicted in Figure 6.7, where we see that perturbation and sanitization are commonly used in 13 (29%) and 12 (27%) of the papers, respectively. This corresponds with the view of many researchers that perturbation techniques (particularly, differential privacy) are the de facto standard for privacy-preserving ML [149]. Another reason for the high popularity of perturbation techniques can be that they have been extensively researched not only in the FL community, but in the ML community in general. In contrast, sanitization is specially used only in federated settings. On the other hand, cryptography-based approaches are considered for the attack evaluation in a fewer number of papers (3 i.e. 7%). One explanation could be that these approaches provide formally proven guarantees, so that empirical experiments are not required to evaluate their impact on the attacks. Therefore, they are mostly mentioned only briefly in the papers under our review, without a thorough discussion.

IMPLEMENTATION TECHNOLOGIES. To ease the reproducibility of the evaluation results, researchers are encouraged to share appropriate descriptions of their implementations along with their source code [48]. In order to learn about the status of the selected papers in this respect, two factors were examined:

- Technologies description: Here, we checked whether or not the authors stated clearly which technologies they used to implement their experiments, such as programming languages and libraries.
- Source code availability: We checked whether the source code is publicly available or not.

Table 6.3 shows that 25 (57%) papers reveal information about the technologies used in their implementation. A special notice can be put on the popularity of Python as a programming language and PyTorch as a specific Python package in this field. The large share of Python-based implementations can be due to the fact that Python is easy to use and provides a large number of packages for ML tasks. PyTorch is user-friendly and suitable to create custom models, for that and other reasons, it is widely used in the ML research community. On the other hand, 19 (43%) papers do not report detailed information about their implementation. Moreover, the mapping shows that the source code of only 6 (14%) papers is shared publicly.

Reported		Unreported
Python	23 (52%)	
PyTorch	18 (41%)	-
Public source code	6 (14%)	
Total	25 (57%)	19 (43%)

Table 6.3: Paper distribution w.r.t. reporting details on the implementation techniques†.

6.5 DISCUSSION

In this section, first, we derive gaps in the research field from the mapping results in Section 6.4. Second, we highlight several special assumptions made in the problem settings of some papers that might reduce the applicability of the attacks in real-world scenarios. Third, we identify fallacies in the evaluation of the attacks and discuss their implications on the generalizability of the results.

6.5.1 *Main research gaps*

We base our discussion here on the results of Section 6.4. In addition, we are looking at how the papers are distributed over pairs of categories by the means of bubble charts, as shown for example in Figure 6.8, where we show how attacks with specific purposes are distributed w.r.t. the access point. It is worth mentioning that the categories in some classification schemes are not disjoint, therefore, the total number of publications may sum up to more than 44.

GAP 1

Little research is conducted about attacks on the server side and by eavesdroppers.

Description. Figure 6.8 illustrates that membership, property inference, model corruption, and backdoor attacks are rarely studied on the server side or with an eavesdropper adversary. This might be due to two reasons. First, it is widely assumed in the literature that FL is coordinated by a trusted server. Second, approaches that protect against curious servers and eavesdroppers, such as secure aggregation [23], were proposed and widely used by the research community because of the firm protection guarantees they achieve. However, applying such approaches still incurs nonnegligible overhead [241], despite the improvements, which leaves open questions about their efficiency in real-world applications.

Implications. FL is mainly introduced for privacy protection, meaning that privacy-critical (personally identifiable) data is supposed not to reach the server, making server-side attacks seem less relevant (similar for eavesdroppers). Additionally, servers (service providers) are supposedly better equipped to repel attacks compared with users. However, numerous events in recent years showed that providers were subjects to many successful attacks, where user data was breached [174]. In such server-side attacks, the adversaries could access far more (or more valuable) data than in user-side attacks. Therefore, it is of high

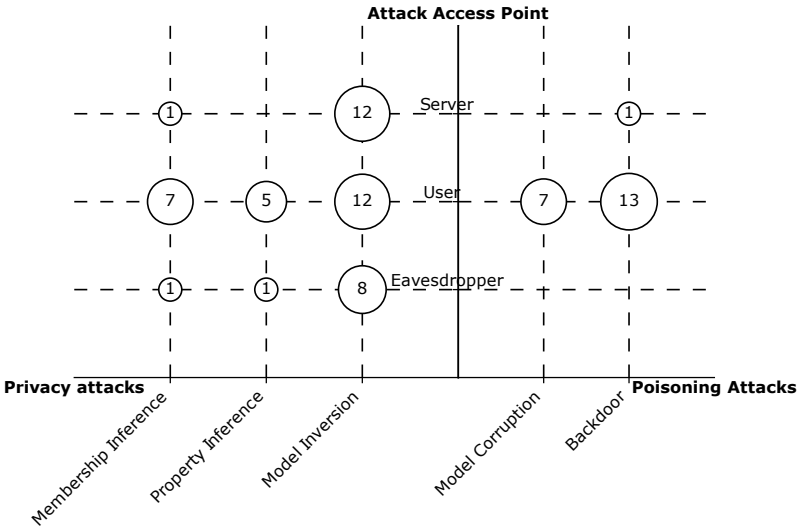


Figure 6.8: Bubble chart that shows the paper distribution on two dimensions: attack purpose and access point. Aside from model inversion, we notice low number of attacks for the server and eavesdropper access point†.

importance to study how attacks by a curious or compromised server can impact the FL process. We argue that attacks on the server side are becoming even more relevant in FL especially considering the emergence of applying FL in different architectures, such as hierarchies in edge networks [118, 159, 232]. In such environments, there are multiple entities that play the role of *intermediate* servers, i.e., collect and aggregate the updates from users, thus, introducing more server-type access points.

For eavesdroppers, recent model inversion attacks on gradients were proved to successfully reconstruct user training data [313, 316]. This opens the door for more investigations about how gradients or model updates can be exploited to apply other attack types, especially privacy attacks.

GAP 2

Very little effort is devoted to studying attacks on ML functions other than classification.

Description. ML models can be used to fulfill a variety of functions, such as classification, regression, ranking, clustering, and generation. However, our SMS shows that there is a heavy bias towards the classification function with 42 (95%) of the attacks. Other functions, namely regression, generation, and clustering were addressed in only 4 (9%), 1 (2%), and 1 (2%) attacks, respectively.

Implications. This gap introduces a lack of knowledge w.r.t. a large spectrum of models and applications that have different functions than classification. These functions are of high importance in many domains, e.g., ranking in natural language processing [303] and recommender systems [202]. It is an open question how the existing attacks impact these functions. It is worth mention-

ing that a similar gap was also observed for adversarial attacks in general ML settings by Papernot et al. [203].

GAP 3

There is a lack of research regarding attacks on ML models other than CNNs.

Description. Although FL is not restricted to NN models, we have seen in the previous section that only 3 (6%) attacks target non-NN models. At a closer look, we depict in Figure 6.9 the types of models targeted by the different attacks. We notice that non-NN models were never targeted by membership inference or backdoor attacks. For NN models, we observe that RNNs were not studied under any type of privacy attacks or model corruption attacks. Additionally, no research has been carried out yet on backdoors for DNNs. The AEs also have received very little attention with only 2 privacy attacks. Overall, this illustrates the limited diversity in the literature considering the target models.

Implications. NN models are the state of the art in several applications, e.g., face recognition [12], however, other ML models are still of high value and usage in real-world systems, e.g., genome analysis [54], culvert inspection [82], and autocompletion suggestions filtering [302], to name a few.

Within the NN models, there is a variety of network architectures, and as we show above, many of these architectures are not well covered in the evaluation of the attacks, even architectures that are widely used in several applications, e.g., RNN, which is used in Gboard [104]. Consequently, the evaluations of the proposed attacks fall short of providing evidence on how the attacks will perform against other network architectures.

Overall, we notice very little effort devoted to studying the influence of using different model architectures on the effectiveness of the proposed attacks. Only in one paper [85], the authors adequately analyzed the effects of the NN architecture on the success of their attack. Covering this aspect in the evaluation of the attacks is essential to improve the generalizability of the findings.

GAP 4

Limited investigations are conducted on attacks targeting the ground-truth labels of user data.

Description. In supervised learning, users train models with their data (e.g., images) and ground-truth labels (i.e., annotations correctly describe the context of the data). For example, to train an image recognition model, users would need images and labels that refer to the objects found in the images, e.g., "person", "car", or "animal". A considerable number of works (10, i.e. 23%) have shown that the shared gradients in FL can be exploited by adversaries to reconstruct the user training data, e.g., [8, 85, 288, 315]. On the other hand, only 3 (7%) papers [148, 313, 316] investigated attacks that can disclose the ground-truth labels of users.

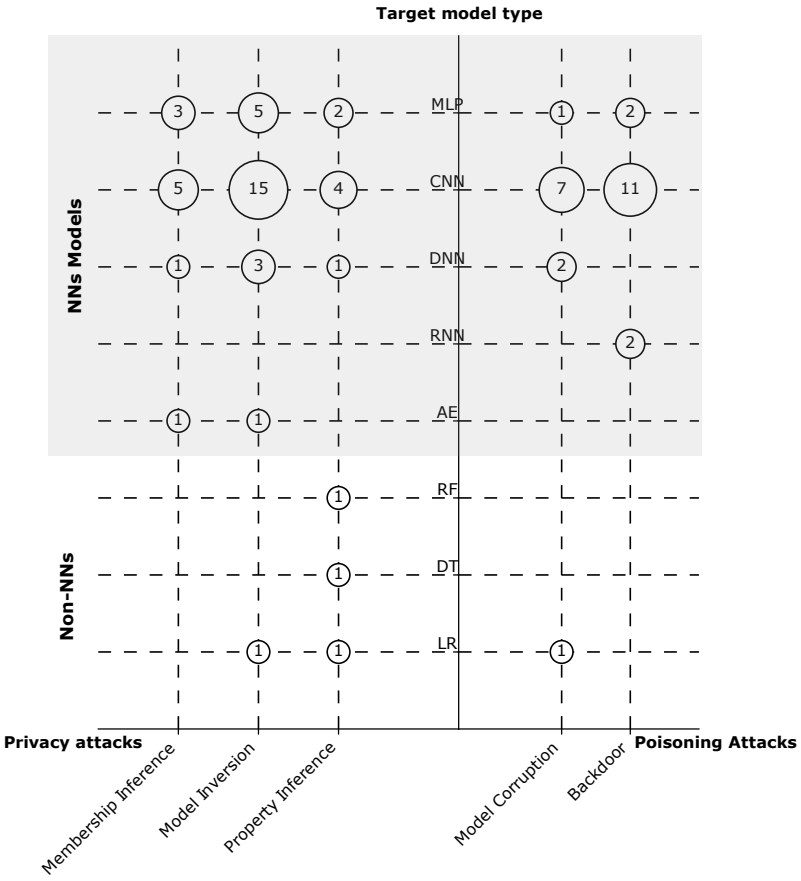


Figure 6.9: Bubble chart that shows the paper distribution on two dimensions: attack purpose and target model. We see very low frequency of all attack types on non-NN models. Also, NN models as Recurrent Neural Network (RNN) and Autoencoder (AE) receive low attention in various attacks[†].

Implications. FL can be used in many applications where ground-truth labels are of highly sensitivity. For instance, medical institutes leverage distributed learning to build joint ML models for disease diagnosis [79, 130]. In several cases, medical data is generated on patients' personal devices [49], e.g., mobile phones [65], thus, an application of FL could bring potential benefits. In that and many other scenarios, training models while preserving user privacy would be essential. Attacks that target the ground-truth labels of user data could lead to the disclosure of their diseases, which is a serious privacy violation. Therefore, it is important to further explore this topic and investigate the extent to which gradients can reveal information about ground-truth labels.

6.5.2 *Special assumptions in problem settings*

There are a number of attacks that succeed only under special assumptions. These assumptions do not apply in many real-world scenarios, resulting in limiting the applicability of these attacks. Here, we highlight the issues of these assumptions and discuss their implications.

ASSUMPTION ISSUE 1

The attacks are effective only under special values of the hyper-parameters of NN models.

Description. The hyper-parameters of NN models include, among others, batch size, learning rate, activation function, and loss function. Tuning the hyper-parameters is crucial to achieve high accuracy in the learning task, especially when comparing different models. Dacrema et al. [48] showed that the lack of optimization for the hyper-parameters of the baselines leads to phantom progress in the field of neural recommender systems. Therefore, the hyper-parameters need to be carefully and fairly optimized to meet the application requirements. On the contrary, we found in the studied papers several assumptions on special values of hyper-parameters that are not commonly used or might contradict the application requirements. The reason is that the effectiveness of some proposed attacks is highly influenced by hyper-parameters, and these attacks are possible only under such special assumptions.

Examples and Implications. In some model inversion attacks, the gradients are used to reconstruct the training data. Zhu et al. [316] and Wei et al. [289] showed that their attacks perform well only when the gradients are generated from a batch size < 8. Zhao et al. [313] proposed an attack to extract the labels of the users from gradients. However, the attack works only when the batch size is 1, which is an exceptional and uncommon value. Hitaj et al. [114] also used a batch size of 1 to evaluate their attack on the At&T dataset.

Using small batches leads to a lack of accurate estimation of the gradient errors, this in turn causes less stable learning. Additionally, this requires more computation power to perform a large number of iterations, where gradients need to be calculated and applied every time to update the weights. While FL pushes the training to the user device, it is essential to consider the limited resources of the user devices. Therefore, the efficiency of the local training process is an important requirement. That is, the batches of very small sizes < 8 increase the computational overhead and are therefore not preferable for FL applications.

Although it is insightful to point out the vulnerabilities that some special hyper-parameters might introduce, it is of high importance to discuss the relevance of these hyper-parameters to real-world problems.

ASSUMPTION ISSUE 2

The attacks succeed only when a considerable fraction of users are malicious and participate frequently in the training rounds.

Description. In cross-device FL, a massive number of users (up to 10^{10}) form the population of the application. Out of these users, the server selects a subset of users ($\sim$ 100 [302]) randomly for every training round to train the model locally and share their updates [175]. This random sampling is assumed to be

uniform (i.e., the probability for a user to participate is $1/\text{user population}$) to achieve certain privacy guarantees for users, in particular differential privacy [2]. Under these conditions, it is rather unlikely for a specific user to participate in a big number of training rounds ($\gg \frac{\text{total number of rounds}}{\text{user population}}$) or consecutive ones. However, this was found as an assumption in a number of papers to enable some privacy and poisoning attacks. Furthermore, several attacks require a large number of users to collude and synchronize in order to launch an attack, which also can be tricky to achieve in some cases.

Examples and Implications. Hitaj et al. [114] assumed that the adversary participates in more than 50 consecutive training rounds in order to carry out a reconstruction attack successfully. A stronger assumption was made by [306], namely to have the adversary participating in all the rounds to poison the model. This requires the adversary to fulfill the FL training requirements [302] and to trick the server to be selected frequently, which is a challenge per se considering the setting described above.

State-of-the-art poisoning attacks in cross-device FL [15, 74] assumed up to 25% of the users to be malicious. Considering that cross-device FL is mainly intended to be used by a massive number of users, the effective execution of these attacks would require the compromise of a significant number of devices. This in turn requires a very high effort and considerable resources, which could make the attacks impractical at scale [231]. For instance, a real-world FL application such as Gboard [104] has more than 1 billion users [53]. This means that the adversary needs to compromise 250 million user devices to apply these attacks successfully [231]. However, it is worth mentioning that there are many ML applications (i.e. potential FL applications) in the market that are used by a smaller user base. Therefore, a smaller number of compromised devices would be required to apply the aforementioned attacks. Yet, to the best of our knowledge, there is no real-world FL applications that represent this case.

It is true that the distributed nature of FL might enable malicious users to be part of the system. However, the capabilities of these malicious users to launch successful attacks need to be carefully discussed in the light of applied FL use cases. Thus, the risk of these attacks is not overestimated.

ASSUMPTION ISSUE 3

The attacks can be performed when the data is distributed among users in a specific way.

Description. FL enables users to keep their data locally on their devices, i.e., the data remains distributed. This usually introduces two data properties; first, the data is non-IID, i.e., the data of an individual user is not representative of the population distribution. Second, the data is unbalanced as different users have different amounts of data [175]. In an ML classification task, for example, this may cause that some classes are not equally represented in the dataset. In any FL setting, it is essential to consider these two properties. While the meaning of IID and balanced data is clear, non-IID and unbalanced data distribution can

be achieved in many ways [132]. In a number of papers, we found that specific distributions are assumed to enable the proposed attacks and to draw general conclusions.

Examples and Implications. A backdoor attack on a classification model by Bagdasaryan et al. [11] was claimed to achieve 100% accuracy on the backdoor task by one malicious user participating in one training round. However, in this work, it was assumed that only the adversary has the data of the backdoor label, which is a strong assumption according to [80, 253]. The massive number of users in FL suggests that the user data covers all the model classes. Therefore, it should be considered that at least one honest user will have additional benign data for the backdoor label.

Another example is found in the model inversion attack of [114], where the authors assumed that all data of one class belongs to one user, and the adversary is aware of that. Additionally, their attack works only when all the data of one class is similar (e.g., images of one digit in the MNIST dataset). These assumptions do not apply to many real-world scenarios, thus, found unrealistic by [187]. Moreover, the model corruption attack introduced in [259] was launched under the setting of IID data, which contradicts the main FL assumptions. Similarly, Nasr et al. [187] evaluated their membership inference attack on a target model trained with balanced data. It is worth mentioning that Jayaraman et al. [126] showed that most membership inference attacks [163, 222, 237] for stand-alone learning also focus only on the balanced distribution scenarios.

Overall, the way of implementing non-IID and unbalanced data distribution needs to be (1) discussed and justified in the light of the application to assure the setup is as realistic as possible, (2) reflected clearly in the conclusions of the evaluation.

6.5.3 *Fallacies in evaluation setups*

Designing a comprehensive and realistic experimental setup is essential to prove the applicability of the attack and the generalizability of the conclusions. Although all the studied papers provide insightful evaluations of their proposed attacks, a number of practices were followed that might introduce fallacies. In this section, we set out to highlight this issue by identifying six fallacies. We discuss the implications of each fallacy on the evaluation results. Then, we propose a set of actionable recommendations to help avoiding it.

FALLACY 1

The datasets are oversimplified in terms of data content or data dimensions.

Description. The datasets are used to train and test the FL model, and also to evaluate the attack. These datasets need to be representative of the population targeted by the model. As we highlighted in Section 6.4, the majority of attacks are evaluated on the image classification task. Therefore, here we focus on the image-based datasets.

Despite the growing calls for decreasing the usage of simple datasets, in particular MNIST [294], it is still one of the most common datasets in the deep learning community [101]. This is due to several reasons such as its small size and the fact that it can be easily used in deep learning frameworks (e.g., Tensorflow, PyTorch) by means of helper functions [294].

MNIST was introduced by LeCun et al. [145] in 1998 and contains 70,000 gray-scale images of handwritten digits in the size of 28×28 pixels. Since then, substantial advances were made on deep learning algorithms and the available computational power. Consequently, MNIST became an inappropriate challenge for our modern toolset [105]. In addition, the complexity of images increased in modern computer vision tasks. That renders MNIST unrepresentative of these tasks [19].

Yet, the phenomenon of the wide usage of MNIST is also observed in the examined papers, where more than 56% of the papers (see Figure 6.6) use MNIST as the main dataset for evaluating the effectiveness of the proposed attacks. The second most common dataset is CIFAR, which is more complex in terms of data content, however, it is a thumbnail dataset, i.e., images with a size of 32×32 pixels.

It is worth mentioning that in 37 (84%) of the papers the authors evaluated their attacks on more than one dataset, which is considered good practice. However, in a considerable number of papers (15 i.e. 34%) the authors used only datasets that either contain simple or small (thumbnail) images.

Examples and Implications. Using oversimplified datasets can lead to the misestimation of the attack capabilities. For instance, the capabilities of privacy attacks to retrieve information about the dataset are tightly related to the nature of this dataset. Consequently, the complexity and size of the images in the dataset impact the attacks' success rate. It is clear that obtaining complex and bigger images require higher capabilities. This is evident in the literature through several examples. Melis et al. [177] introduced a privacy attack that exploits the updates sent by the users to infer the membership and properties of data samples. In [316], the authors demonstrated that the proposed attack of [177] only succeeds on simple images with clean background from the MNIST dataset. However, the attack's accuracy degrades notably on the LFW dataset and fails on CIFAR. In the same context of privacy attacks, Zhu et al. [316] proposed the model inversion attack DLG, which reconstructs the training data and labels from gradients. Their experiments showed that DLG can quickly (within just 50 iterations) reconstruct images from MNIST. However, it requires more computational power (around 500 iterations) to succeed against more complex datasets such as CIFAR and LFW. Recently, we demonstrated in [277] that the accuracy of DLG in retrieving the labels degrades remarkably on CelebA, which has a bigger image size than the thumbnails datasets, MNIST and CIFAR.

Recommendations. It is challenging to find a single dataset that provides an adequate evaluation of the attacks, therefore, it is essential to evaluate the attack on diverse datasets w.r.t. image complexity and dimensions. We encourage researchers to also consider real-life datasets, which pose realistic challenges for

the models and attacks, e.g., ImageNet [58] (image classification and localization), Fer2013 [92] (facial recognition), and HAM10000 [44] (diagnosing skin cancers).

FALLACY 2

The datasets are not user-partitioned, i.e., not distributed by nature.

Description. In FL, data is distributed among the users; each user typically generates their data by using their own device, therefore, this data has individual characteristics [175]. The datasets used for evaluating the attacks should exhibit this property, i.e., generated in a distributed fashion. However, only in 2 (4%) of the papers, user-partitioned datasets were used, in particular EMNIST [45], which is collected from 3383 users, thus, appropriate for the FL setting [253]. On the other hand, researchers in the majority of papers (42 i.e. 96%), used pre-existing datasets that are designed for centralized machine learning [165], thus, unrealistic for FL [32]. These datasets then are artificially partitioned to simulate the distributed data in FL. One additional issue with these datasets is that they are by default balanced, while FL assumes the user data to be unbalanced [175].

Examples and Implications. The poisoning attacks proposed in [11] and [19] were evaluated on centralized datasets, such as Fashion-MNIST and CIFAR, for image classification, where the attacks were reported achieving 100% accuracy in the backdoor task. However, by using EMNIST as a standard FL dataset, Sun et al. [253] illustrated the limitations of the previous attacks. More precisely, they showed that the performance of the attacks mainly depends on the ratio of adversaries to the population. Moreover, the attacks can be easily mitigated with norm clipping and "weak" differential privacy.

Although this fallacy was discussed in previous works [32, 165], its implications on the evaluation results need to be investigated further and demonstrated with empirical evidence.

Recommendations. FL-specific datasets should be used for adequate evaluation of the attacks. Researchers have recently been devoting more efforts to curating such datasets. The LEAF framework [32] provides five user-partitioned datasets of images and text, namely FEMNIST, Sent140, Shakespeare, CelebA, and Reddit. Furthermore, Luo et al. [165] created a street dataset of high-quality images, which is also distributed by nature for FL.

FALLACY 3

The attacks are evaluated against simple NN models.

Description. We observe a major focus on attacking NN models in federated settings. These models can have a variety of architectures. The complexity of these architectures vary w.r.t. the number of layers (depth), the number of neurons in each layer (width), and the type of connections between neurons. Our study shows that researchers tend to use simple architectures to evaluate their

attacks in 30 (68%) papers, e.g., 1-layer CNN [70] or 1-layer MLP [15]. Only in 14 (32%) papers, the authors considered complex state-of-the-art CNN models, such as VGG [238], ResNet [109], and DenseNet [122], the winners of the famous *ImageNet Large Scale Visual Recognition Challenge (ILSVRC)* [220].

Examples and Implications. It is reasonable to start evaluating novel attacks on simple models to facilitate the analysis of the initial results. However, this is insufficient to draw conclusions on the risk posed by these attacks to real-life FL-based applications for two reasons. First, modern computer vision applications, e.g., biometrics, use advanced models, mostly with sophisticated architectures, to solve increasing complex learning objectives [135]. Second, in deployed systems, a ML model typically interacts with other components, including other models. This interaction can be of extreme complexity, which might introduce additional challenges for adversaries [72]. For instance, in the Gboard app [104], as a user starts typing a search query, a baseline model determines possible search suggestions. Yang et al. [302] utilized FL to train an additional model that filters these suggestions in a subsequent step to improve their quality.

Several model inversion attacks reconstruct the training data by exploiting the shared gradients [70, 279, 289]. In particular, they exploit mathematical properties of gradients in specific model architectures to infer information about the input data. For example, Enthoven et al. [70] illustrate that neurons in fully connected layers can reconstruct the activation of the previous layer. This observation is employed to disclose the input data in fully connected models with high accuracy. However, the same attack achieves less success when the model contains some convolutional layers.

The NN capacity (i.e., number of neurons) also influences the performance of some attacks, in particular backdoors. It is conjectured that backdoors exploit the spare capacity in NNs to inject a sub-task [157]. Thus, larger networks might be more prone to these attacks. However, this interesting factor still needs to be well investigated [253]. In this regard, it is worth mentioning that increasing the capacity, e.g., for CNNs, is a common practice to increase the model accuracy. However, recent approaches such as EfficientNet [254] call for scaling up the networks more efficiently, achieving better accuracy with smaller networks. This development in the CNNs should be also considered in the evaluation of the attacks.

Recommendations. We highly encourage the researchers to consider the state-of-the-art model architectures that are widely used in the application, where they apply their attack. In addition, it would be insightful for a more realistic security assessment to consider evaluating the proposed attacks on deployed systems that contain multiple components.

FALLACY 4

The attacks are designed for cross-device scenarios (massive user population), yet evaluated on a small number of users $\leqslant 100$.

Description. FL can be applied in cross-silo or cross-device settings. In the cross-silo setting, users are organizations or datacenters (typically 2-100 users

in total), whereas in the cross-device scenario, users are a very large number of mobile or IoT devices (massive up to 10^{10}) [132]. For instance, in applied use cases of FL, Hard et al. [104] reported using 1.5 million users to train the Coupled Input and Forget Gate language model [95]. Yang et al. [302] trained a logistic regression model (for the Gboard application) for 4000 training rounds, where they employed 100 users in each round.

Although many of the studied papers do not explicitly use the term "cross-device" to describe their scenario, they refer mainly to users as individual users who have personal data. However, 27 (61%) of the papers provide an evaluation with a total population of $\leqslant$ 100 users. Moreover, 12 (27%) of the papers did not report at all the user population in their experiments.

Examples and Implications. The total number of users and the users participating per round in FL determine the influence of a single user on the global model. For privacy attacks, this means that each user contributes considerably to shape the model parameters, thus, the parameters more prominently reflect the user personal data. Shen et al. [232] demonstrated that increasing the user population led to a decrease in the accuracy of their property inference attack. For poisoning attacks, using a small number of users amplifies the impact of the poison injected by malicious ones. This was shown in the experiments of [19], where the accuracy of the backdoor task degraded with bigger user populations.

Recommendations. We recommend researchers to consider a large number of users to evaluate novel attacks. For that, it is helpful to use the datasets provided by LEAF [32], which contain more than 1000 users. In case large-scale evaluation is not feasible, researchers are encouraged to discuss at least the potential implications of different user populations on their attacks.

FALLACY 5

The attacks are not evaluated against existing defense mechanisms.

Description. An attack becomes ineffective, if it requires the adversary to make a disproportional large effort to overcome a small defense mechanism [72]. Proposed attacks need to be evaluated in this respect with state-of-the-art defenses. However, we showed in Section 6.4.3, Figure 6.7, that 21 (48%) of the proposed attacks were not evaluated against any of the defense mechanisms. In most of these papers, the authors only discussed theoretically potential countermeasures to mitigate their attacks.

Examples and Implications. Real-world FL scenarios usually involve the use of various defense mechanisms. A rigorous and realistic assessment of the effectiveness of the attacks therefore requires evaluation against appropriate state-of-the-art defense mechanisms. Here, it is important to distinguish between the different categories of defense mechanisms. On the one hand, cryptography-based defenses typically offer formally proven properties, so their impact on attacks can be adequately discussed in some cases without empirical evidence. In these cases, however, system efficiency remains an issue that should be considered. On the other hand, the impact of other defense categories, namely

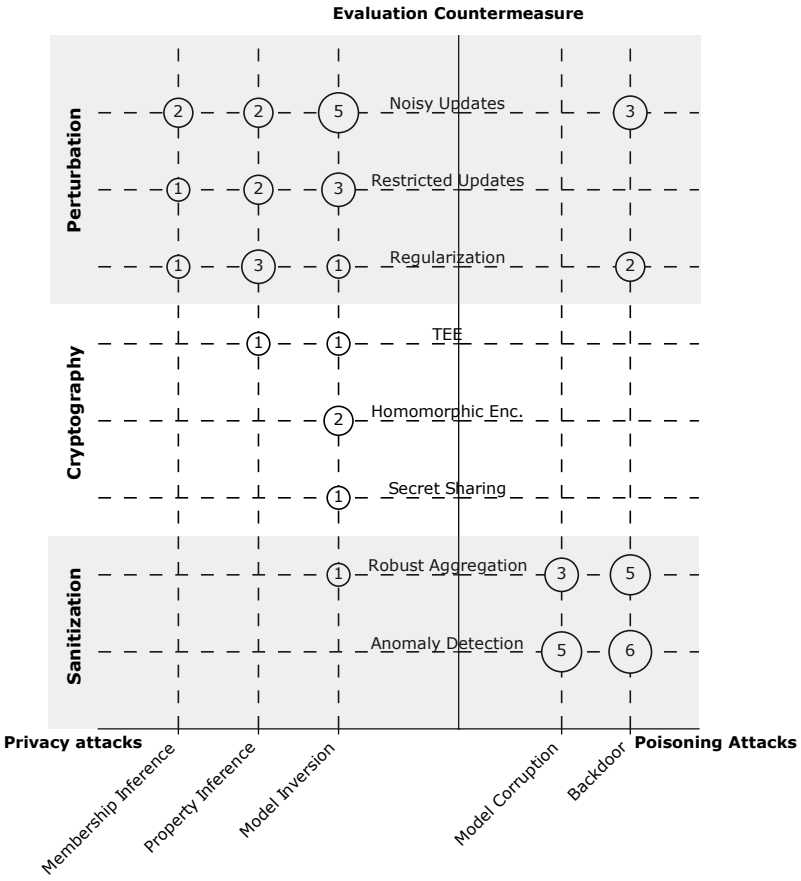


Figure 6.10: chart that shows the paper distribution on two dimensions: attack purpose and countermeasures. We see that perturbation and cryptography-based countermeasures are mainly used for privacy attacks, while sanitization is used for poisoning attacks†.

perturbation and sanitization, against attacks require experimental analysis, as these defenses usually introduce a loss in model accuracy. Thus, the defenses need to be customized to balance the trade-off between accuracy and privacy. An honest discussion if that balance can actually be achieved is also critical—sufficient privacy may lead to an unbearable accuracy loss. In Figure 6.10, we see that most of the implemented defenses in the literature are from these two categories. We see also that perturbation is mainly used for privacy attacks, which reduces the information leakage about individuals, whereas sanitization mitigates the impact of malicious updates from adversaries, thus, used against poisoning attacks.

Recommendations. We highly recommend evaluating novel attacks against a number of appropriate state-of-the-art defenses. For implementing perturbation approaches, emerging libraries such as Opacus[1] and Tensorflow Privacy[2] can be used.

1 https://github.com/pytorch/opacus
2 https://github.com/tensorflow/privacy

FALLACY 6

The results of the experimental evaluations are not easily reproducible.

Description. The majority (97%) of the proposed attacks are validated through empirical experiments. To accurately reproduce the results of these experiments by other researchers, several practices need to be considered. In our analysis, we take into account three main practices: (1) using publicly available datasets, (2) reporting technical details about the implementation, and (3) publishing the source code. Our study shows in Section 6.4.3 that public datasets were used in all the examined papers, which is a good practice. However, 19 (43%) of the papers did not report any details about the technologies used in the implementation. Furthermore, the authors of 38 (86%) of the papers did not publish their source code.

Examples and Implications. Dacrema et al. [48] reported that reproducibility is one of the main factors to assure progress for research, especially with approaches based on deep learning algorithms. To conduct a proper assessment of a novel attack, researchers usually compare it with previous attacks as baselines. Evaluating the different attacks under different settings and assumptions hinders this direct comparison. That is, researchers have to re-implement the respective attacks to reproduce their results under different settings. This becomes even more challenging when the authors do not describe their experiment setups and parameters to the extent of full reproducibility.

Recommendations. We encourage all researchers to share their source code and detailed descriptions of their setups. We also recommend using libraries and benchmark frameworks that support FL, namely Tensorflow-federated [198], PySyft [221], LEAF [32], FATE [301], and FedML [108]. This in turn will help researchers to implement their ideas more easily and improve the consistency of implementations and experiment settings across different papers.

6.6 CONCLUSION

Despite the privacy advantages of FL, the literature showed that the user data still can be prone to several threats. In this chapter, we carried out a systematic mapping study based on recent publications that address attacks in the FL setting. For that, we analyzed 44 relevant papers published between 2016 and the first quarter of 2021. We structured these papers in classification schemes regarding attack types and evaluation settings.

Our analysis indicated the prevalence of works focusing on the classification function and on neural network models, CNN models in particular, which hardly reflects the diversity of ML algorithms. We additionally examined the assumptions of the proposed attacks to identify those with restricted applicability in the context of real-world scenarios. These assumptions range from choosing unorthodox values of hyper-parameters to constructing special kinds of data distribution among users. We further identified six fallacies in the evaluation

of the attacks, which affect the validity of the results and lead to overestimating the effectiveness of the attacks. For instance, the usage of overly simple or centralized datasets was found in the majority of the publications. Moreover, close to half of the attacks were proposed without considering the state-of-the-art defense mechanisms. Notably, there is ambiguity regarding issues on reproducible research. As a constructive step, we presented several actionable recommendations to mitigate these identified fallacies by using modern models, federated learning-specific datasets and frameworks. Overall, our study revealed that each of the examined papers contains at least one of the special assumptions or is affected by one of the evaluation fallacies. Thus, the effectiveness of the attacks in real-world scenarios needs to be further investigated and supported by empirical evidence.

In the context of our thesis, this study provided a structured overview that helped us to orient our research towards novel contributions that address the existing gaps. In the next chapter, we tackle one of these gaps, in particular, we investigate the leakage of ground-truth labels in FL. We demonstrate this leakage by proposing a new attack. Using our findings on the assumptions issues and evaluation fallacies, we carefully craft our assumptions and evaluation setting for our attack to avoid the identified issues and fallacies.

ID	Paper	Year	Venue	Affiliation	Type of Research	Attack Purpose	Attack Mode	Observation	Access Point
1	Hitaj et al.[114]	2017	C,R	A	S	MV	A	W	U
2	Bagdasaryan et al.[11]	2018	J,R	A	S	BD	A	W	U
3	Bhagoji et al. [18]	2018	W	A,I	S	BD	A	W	U
4	Bhagoji et al. [19]	2019	C,R	A,I	S	BD	A	W	U
5	Wang et al.[287]	2019	C,R	A	S	MV	A,P	W	S
6	Nasr et al.[187]	2019	C,R	A	S	MI	A,P	W,B	U,S
7	Zhu et al.[316]	2019	C,R	A	S	MV	P	W	U,S,E
8	Wang et al.[281]	2019	R	A	S	PI	P	W	U
9	Melis et al.[177]	2019	C,R	A	S	MI,PI	A,P	W	U
10	Mao et al.[172]	2019	C	A	S	MI,MV	A	W	U
11	Liu et al. [158]	2019	C,R	A	S	MI	P	W	U
12	Sun et al.[253]	2019	R	I	E	BD	A	W	U
13	Fang et al.[74]	2019	R	A	S	MC	A	W,B	U
14	Zhang et al.[307]	2019	C	A	S	BD	A	W	U
15	Mahloujifar et al.[170]	2019	C	A	S	BD	A	W	U,S
16	Tomsett et al.[260]	2019	C	I	S	BD	A	W	U
17	Cao et al.[33]	2019	C	A	E	BD	A	W	U
18	Baruch et al.[15]	2019	R	A	S	MC,BD	A	W	U
19	Fung et al.[80]	2019	R	A	S,E	MC,BD	A	B	U
20	Zhao et al.[313]	2020	R	A	S	MV	P	W	U,S,E
21	Wei et al.[289]	2020	R	A	E	MV	A	W	U,S,E
22	Pustozerova et al.[213]	2020	W	A	E	MI	P	W	U
23	Geiping et al.[85]	2020	R	A	S	MV	A,P	W	S,E
24	Sun et al.[252]	2020	R	A	S	MC	A	W	U
25	Nguyen et al.[191]	2020	W	A	S	BD	A	B	U
26	Chen et al.[38]	2020	C,R	A	S	BD	A	W	U
27	Song et al.[243]	2020	J	A	S	MV	A,P	W	S
28	Zhang et al.[309]	2020	C	A	S	MI	P	W	U
29	Zhang et al.[306]	2020	J	A	S	MC,BD	A	W	U
30	Tolpegin et al.[259]	2020	C,R	A	S	MC	A	W	U
31	Luo et al.[168]	2020	R	A	S	PI	P	W	U
32	Zhu et al.[315]	2020	R	A	S	PI	P	W	U,E
33	Mo et al.[182]	2020	R	A	S	MV	P	W	U
34	Wu et al.[293]	2020	C	A	S	MV	P	W	U,S,E
35	Wang et al.[285]	2020	R	A,I	S	MV	P	W	U,S,E
36	Xu et al.[297]	2020	C	A	E	PI	A,P	W	U
37	Chen et al.[39]	2020	C	A	E	MI	P	W	U
38	Lu et al.[164]	2020	R	A,I	E	MI	P	W	E
39	Xu et al.[298]	2020	C	A	E	MV	A	W	U
40	Qian et al.[215]	2020	R	A	E	MV	P	W	U,S,E
41	Xie et al.[295]	2020	C,R	A	E	MC	A,P	B	U
42	Wainakh et al.[277]	2021	C	A	S	MV	P	B,W	U,S,E
43	Shen et al.[232]	2021	J	A	E	MV	P	W	U,S
44	Enthoven et al.[70]	2021	R	A	S	MV	P	W	S

Acronyms | Venue: Conference (C), Public Repository (R), Journal (J), Workshop (W) | Affiliation: Academic (A), Industrial (I) | Type of Research: Solution (S), Evaluation (E) | Attack Purpose: Membership Inference (MI), Model Inversion (MV), Property Inference (PI), Model Corruption (MC), Backdoor (BD) | Attack Mode: Active (A), Passive (P) | Observation: White Box (W), Black Box (B) | Access Point: Server (S), User (U), Eavesdropper (E).

Table 6.4: Mapping results for the studied papers w.r.t. meta data and attacks properties[†].

ID	Paper	Target Model	Num. of Datasets	Countermeasures	Public Code	Python	Libraries
1	Hitaj et al.[114]	CNN	2	NU	✗	✗	t7
3	Bagdasaryan et al.[11]	CNN,RNN	2	AD,NU,RA	✗	✓	pt
3	Bhagoji et al. [18]	CNN	1	✗	✗	?	✗
4	Bhagoji et al. [19]	CNN,MLP	2	RA	✗	?	✗
5	Wang et al.[287]	CNN	2	✗	✗	?	✗
6	Nasr et al.[187]	CNN,MLP	3	✗	✗	✓	pt
7	Zhu et al.[316]	CNN,AE	4	NU,RU,SS,HE	✓	✓	pt
8	Wang et al.[281]	CNN,MLP	4	RU,Reg	✗	✓	pt,sk
9	Melis et al.[177]	CNN	7	RU,Reg,NU	✗	?	✗
10	Mao et al.[172]	CNN,DNN	2	✗	✗	?	✗
11	Liu et al. [158]	CNN,AE	3	✗	✗	?	✗
12	Sun et al.[253]	CNN	1	NU,Reg	✓	✓	tf,tff
13	Fang et al.[74]	LR,CNN,DNN	4	AD	✗	?	✗
14	Zhang et al.[307]	CNN	2	✗	✗	✓	pt
15	Mahloujifar et al.[170]	✗	✗	✗	✗	?	✗
16	Tomsett et al.[260]	CNN	1	✗	✗	✓	pt
17	Cao et al.[33]	CNN	1	RA	✗	?	✗
18	Baruch et al.[15]	CNN,MLP	2	AD,RA	✗	✓	pt
19	Fung et al.[80]	CNN	4	AD,RA	✗	✓	sk
20	Zhao et al.[313]	CNN	3	✗	✓	✓	pt
21	Wei et al.[289]	CNN,MLP	5	NU,Reg	✗	?	✗
22	Pustozerova et al.[213]	MLP	1	NU	✗	?	✗
23	Geiping et al.[85]	CNN	3	✗	✗	?	✗
24	Sun et al.[252]	CNN	4	✗	✗	?	✗
25	Nguyen et al.[191]	RNN	3	NU,AD,Reg	✗	✓	pt
26	Chen et al.[38]	CNN	2	AD	✗	?	✗
27	Song et al.[243]	CNN	2	HE,RA,RU,TEE	✗	✓	kr
28	Zhang et al.[309]	✗	1	✗	✗	✓	pt,kr,tf,sk
29	Zhang et al.[306]	CNN	3	AD	✗	✓	pt
30	Tolpegin et al.[259]	CNN,DNN	2	AD	✓	✓	pt
31	Luo et al.[168]	LR,DT,RF,MLP	4	NU,Reg	✗	✓	pt,sk
32	Zhu et al.[315]	CNN,DNN	2	TEE	✓	✓	pt
33	Mo et al.[182]	CNN,DNN	3	✗	✓	✓	pt,th
34	Wu et al.[293]	CNN	3	NU,RU	✗	✓	tf
35	Wang et al.[285]	CNN,DNN	4	✗	✗	✓	pt
36	Xu et al.[297]	CNN	2	✗	✗	?	✗
37	Chen et al.[39]	CNN	2	✗	✗	✓	pt,kr,tf
38	Lu et al.[164]	MLP	2	✗	✗	?	✗
39	Xu et al.[298]	LR,MLP	2	✗	✗	✓	kr,tf,tff,f
40	Qian et al.[215]	CNN,MLP	6	NU	✗	?	✗
41	Xie et al.[295]	CNN	1	RA	✗	?	✗
42	Wainakh et al.[277]	CNN	2	✗	✗	✓	pt
43	Shen et al.[232]	CNN,MLP	4	✗	✗	?	✗
44	Enthoven et al.[70]	CNN,MLP	2	✗	✗	?	✗

Acronyms | Target Model: Convolutional Neural Network (CNN), Multilayer Perceptron (MLP), Deconvolutional Neural Network (DNN), Recurrent Neural Network (RNN), Autoencoder (AE), Logistic Regression (LR), Decision Tree (DT), Random Forest (RF) | Countermeasures: Noisy update (NU), Restricted Updates (RU), Regularization (Reg), Secret Sharing (SS), Homomorphic Encryption (HE), Robust Aggregation (RA), Anomaly Detection (AD), Matching Networks (MN) | Libraries: PyTorch (pt), Torch7 (t7), Tensorflow (tf), TensorflowFederated (tff), Keras (kr), Scikit-learn (sk), Theano (th), Fate (f).

Table 6.5: Mapping results for the studied papers w.r.t. evaluation setups[†].

7

LABEL LEAKAGE FROM GRADIENTS

In the previous chapter, we presented a thorough study on the attacks against FL, and we highlighted several gaps in this research field. One of these gaps is the lack of investigation on threats against the ground-truth labels of user data. Discovering as-yet-unknown threats is a common scientific practice, as it leads researchers and practitioners to find remedies (ideally before attackers exploit the threat), resulting in a very deep understanding of the technologies at hand. Therefore, in this chapter, we contribute a novel attack that discloses the user ground-truth labels by exploiting the shared gradients in a federated setting.

7.1 INTRODUCTION

The general principle of FL is currently believed to reduce the risk and severity of privacy breaches compared to the classical centralized ML setting. That is because personal information (in the sense of primary data i.e. data samples or ground-truth labels) does not leave the user, and sharing learning gradients does not supposedly reveal information about the user [316]. However, a considerable number of recent works have shown that gradients can in fact be exploited to reconstruct the users' training data [8, 85, 288, 315]. On the other hand, protecting the users' ground-truth labels from possible leakage has received only limited attention [148, 313, 316], mainly focusing on gradients generated from a small number of data samples (small batches) or binary classification tasks. Label leakage, however, is a considerable risk for FL. Both, FL as well as the more superordinate setting of distributed ML are used in many applications where ground-truth labels can contain highly sensitive information. For example, in the medical sector, hospitals employ distributed learning to collaboratively build ML models for disease diagnosis and prediction [79, 130]. In some cases, the medical data is collected directly from the patients' personal devices [49], e.g., mobile phones [65], where an application of FL could introduce many potential benefits. Building models in that and many other settings, while maintaining user privacy, would be crucial. Leaking the ground-truth labels of the users' data might disclose their diseases, which is a severe violation of privacy. It is essential to highlight this issue and explore to what extent gradients can leak information about ground-truth labels. For this purpose, developing privacy attacks that exploit gradients is of a high importance in order to foster research and development on the mitigation of respective privacy risks.

7.1.1 *Summary of contributions*

In this chapter, we propose Label Leakage from Gradients (LLG), a novel attack to extract ground-truth labels from shared gradients trained with mini-batch

Stochastic Gradient Descent (SGD) for multi-class classification. The construction of LLG is based on a combination of mathematical proofs and heuristics derived empirically. The attack exploits two properties that the gradients of the last layer of a neural network have: (**P1**) The direction of these gradients indicates whether a label is part of the training batch. (**P2**) The gradient magnitude can hint towards the number of occurrences of a label in the batch. Here, we formalize these properties, provide their mathematical proofs, study an extended threat model, and conduct an extensive evaluation. The key results of the evaluation and their contexts are as follows.

- We consider four benchmark datasets, namely MNIST, SVHN, CIFAR-100, and CelebA. Results show that LLG achieves a high success rate despite the datasets having different classification targets and complexity levels.
- We consider two FL algorithms, namely FedSGD and FedAvg [175]. Results show that for untrained models, LLG is more effective under FedSGD, yet poses a serious threat to expose the ground-truth labels under FedAvg as well.
- We study LLG considering different capabilities of the adversary. Experiments demonstrate that an adversary with an auxiliary dataset, which is similar to the training dataset, can adequately extract the ground-truth labels with an accuracy of $> 98\%$ at the early stage of the model training under the FedSGD algorithm.
- We show that the simple LLG attack can outperform one of the state-of-the-art optimization-based attacks, Deep Leakage from Gradients (DLG) [316], under several settings. Furthermore, LLG is orders of magnitude faster than DLG.
- We also investigate the effectiveness of the attack on various model architectures including simple CNN, LeNet [145], and ResNet20 [109]. Results suggest that LLG is not highly sensitive to the complexity of the model architecture.
- We illustrate the influence of the model convergence status on LLG. Findings reveal that LLG can perform best at the early stages of training and still demonstrates information leakage in well-trained models.
- Finally, we test LLG against two defense mechanisms: noisy gradients and gradient compression (pruning). Results show that gradient compression with $\geqslant 80\%$ compression ratio can render the attack ineffective.

The content of this chapter is based on the following papers.

PUBLICATIONS

- **Wainakh, A.**, Müßig, T., Grube, T., & Mühlhäuser, M. (2021, January). Label leakage from gradients in distributed machine learning. In *18th Annual Consumer Communications & Networking Conference (CCNC) (pp. 1-4). IEEE.*

- **Wainakh, A.**, Ventola, F., Müßig, T., Keim, J., Cordero, C. G., Zimmer, E., Grube, T., Kersting, K., & Mühlhäuser, M. (2022, July). User-Level Label Leakage from Gradients in Federated Learning. In *Proceedings on Privacy Enhancing Technologies (PoPETs)*.

Contribution Statement
I led the process of idea generation, realization, evaluation, and writing. Fabrizio Ventola contributed insightful discussions that helped improve the mathematical proofs, the (ML) models analysis, and he also revised the manuscript. The Bachelor student Till Müßig contributed helpful comments that inspired the conceptual work. He also implemented the main algorithm and initial experiments. Jens Keim participated in implementing the experiments. Carlos Garcia Cordero, Ephraim Zimmer, Tim Grube, and Kristian Kersting contributed fruitful discussions and improvements on the editorial quality of the manuscript. Max Mühlhäuser provided helpful mentoring to improve different aspects of the work.

7.1.2 *Outline*

We proceed as follows. We start off by reviewing our problem setting in Section 7.2. Next, in Section 7.3, we present related work on information leakage from gradients. We elaborate on our findings regarding gradients properties in Section 7.4. The attack is then explained in Section 7.5. Before concluding, we present the results of our evaluation in Section 7.6.

7.2 PROBLEM SETTING

We consider a federated setting where a set of users U jointly trains a neural network model for a supervised task using either FedSGD or FedAvg algorithm [175]. For FedSGD, the users train the model locally for one iteration on a batch of their data samples and ground-truth labels. In FedAvg, each user trains the model for several iterations (multiple batches). The ground-truth labels are the annotations generated typically by the user and specify the correct context of the data w.r.t. the ML task. We assume the users to be honest, i.e., they train the model with real data and correct labels. Then, the users share the gradients resulted from the local training with the server. We assume that the model consists of L layers and is trained with cross-entropy loss [91] over one-hot labels for a multi-class classification task. To extract the ground-truth labels using our attack, it is sufficient to focus on the gradients $\nabla \boldsymbol{W}_L$ w.r.t. the last-layer weights $\boldsymbol{W}_L$ (between the output layer and the layer before), where $\boldsymbol{W}_L \in \mathbb{R}^{n \times h}$: n is the total number of classes and h is the number of neurons in layer $L-1$. The gradient vector $\nabla \boldsymbol{W}_L^i$ represents gradients connected to la-

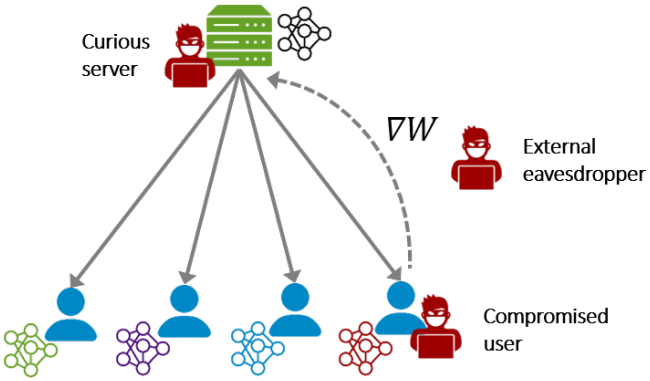


Figure 7.1: Federated learning overview with three potential adversary access points (in red). Gradients are generated by individual users and shared with a central server. An adversary with access to these gradients can exploit them to estimate the presence and frequency of labels, which can be, e.g., a result of a medical imaging technique for disease prediction[†].

bel i on the output layer. We note g_i to refer to the sum of ∇W_L^i elements: $g_i = \mathbb{1}^T \cdot \nabla W_L^i$.

7.2.1 *Threat model*

We assume that an adversary applies the attack against the shared gradients of one target user. The adversary analyzes the gradients to infer the number of label occurrences in the user's input data. In FedSGD, this concerns one batch, while in FedAvg, the data consists of multiple batches. Thus, the more information is carried by the gradients on the labels, the higher is the privacy risk. At the same time, the shared gradients need to reflect the training data of the users to optimize the joint model, i.e., to achieve the learning objective. As a result, the learning objective and the depicted privacy risk are mutually related to the information carried by the gradients. Even though it might seem as a paradox, our work is an attempt to focus on and mitigate the privacy risk imposed by gradient sharing without jeopardizing the learning objective of FL and the model accuracy. Next, we define our threat model w.r.t. three aspects: adversary access point, mode, and observation.

ACCESS POINT. The distributed nature of FL increases the attack surface as shown in Figure 7.1. As for a *compromised user*, an adversary might be able to access the gradients by compromising the user's device, since the gradients are calculated on the user side before being shared with the server. We assume that the user's device can be compromised partially, such that the adversary has no access to the training data or ground-truth labels [289]. Such a scenario can apply, for example, to several online ML applications, where the training data is not stored but rather used for training on-the-fly. In these cases, compromising a device during or after the training phase would not grant the adversary full access to the training data, while still providing access to the model and possibly the gradients. Other scenarios might exploit a vulnerability in the imple-

mentation of the network protocols/interface, such that an adversary accesses only the I/O data. Furthermore, a *curious server* also can access the gradients of an individual user, in case no secure aggregation [23] or other protection techniques are used. In addition, an *external eavesdropper* might intercept the gradients, if the connection between the server and the users is not secure.

MODE. We assume the adversary to act in a passive mode. The adversary may analyze the gradients to infer information about the users, but without hindering or deviating from the regular training protocol. This adversary mode is widely common in privacy attacks [168, 213, 309, 316], where the focus is on disclosing information rather than disturbing the system.

OBSERVATION. The adversary might be capable of observing different amounts of information to launch their attack. We consider three possibilities.

1 *Shared gradients*: The adversary has access only to the shared gradients. This can apply for an external eavesdropper or an adversary with limited access to the user's device.

2 *White-box model*: In addition to the gradients, the adversary is aware of the model architecture and parameters. In the case of a curious server or compromised user, the adversary might have this kind of information.

3 *Auxiliary knowledge*: The adversary has access to all the aforementioned information as well as to an auxiliary dataset. This dataset contains data samples of the same classes as the original training dataset. This is a common scenario in real-world cases, given that neural networks need a considerable amount of labeled data for training to perform accurately. Labeled data is usually expensive and a typical adopted strategy is to train the model on the publicly available datasets and, eventually, fine-tuning the model on ad-hoc data. Therefore, it is often easy to have access to a big part of the training data.

7.3 RELATED WORK

Although the training data is not disclosed to other parties in FL, several works in the literature showed that the data and ground-truth labels can be reconstructed by exploiting the shared gradients. Next, we present existing (1) data reconstruction attacks and (2) label extraction attacks.

DATA RECONSTRUCTION. Aono et al. [8, 9] were the first to discuss reconstructing data from gradients on simple neural networks with a training batch of one sample. Wang et al. [288] moved on to generative attacks, leveraging a Generative Adversarial Network (GAN) to reconstruct the input data in a CNN. In contrast, Zhu et al. [316] introduced an optimization-based attack; the attacker generates dummy input data and output labels, then optimizes them using L-BFGS [156] to generate dummy gradients that match the shared ones. As an improvement, Geiping et al. [85] proposed using cosine similarity and the Adam optimization algorithm. Wei et al. [289] provided a framework for evaluating the optimization-based attacks considering multiple factors, e.g., optimizer, activation, and loss functions. Qian et al. [214] theoretically analyzed the

limits of [316] considering fully-connected neural networks and vanilla CNNs. They also proposed a new initialization mechanism to speed up the attack convergence. Unlike previous approaches, Enthoven et al. [70] introduced an analytical attack that exploits fully-connected layers to reconstruct the input data on the server side, and they extended this exploitation to CNNs. Recently, Zhu et al. [315] proposed a recursive closed-form attack. They demonstrated that one can reconstruct data from gradients by recursively solving a sequence of systems of linear equations. Overall, all the aforementioned attacks, except for [316] (discussed in the next section), only focus on reconstructing the input training data while overlooking the leakage of ground-truth labels, which can be of a high sensitivity. In our research, inspired by the mathematical foundations used in these attacks, we shed more light on the potential vulnerability of label leakage in FL and distributed learning.

LABEL EXTRACTION. While the data reconstruction attacks attracted considerable attention in the research community, a very limited number of approaches were proposed on the label leakage. As part of the optimization approach of Zhu et al. [316], the ground-truth labels are extracted. However, the approach requires a learning phase where the model is sensitive to the weight initialization and can be hard to converge. Moreover, it was found to extract wrong labels frequently [313] and it is effective only for gradients aggregated from a batch size < 8 [182]. Zhao et al. [313] proposed a more reliable analytical approach, which exploits the observation that gradients of classification (cross-entropy loss) w.r.t. the last layer weights have negative values for the correct labels. However, their approach is limited to a one-sample batch, which is uncommon in real-world applications of FL. Li et al. [148] proposed also an analytical approach based on the observation that the gradient norms of a particular class are generally larger than the others. However, their approach is tailored only for a binary classification task in vertical split learning. Overall, the existing approaches are not well generalized to arbitrary batch sizes nor number of classes. Also, the influence of different model architectures on these approaches is yet to be investigated.

7.4 GRADIENT ANALYSIS

In gradient descent optimization, the values of gradient determine how the parameters of a model need to be adjusted to minimize the loss function. Through an empirical analysis, we carefully derive two properties for the sign and magnitude of the gradients that indicate the ground-truth labels. In this section, we formalize these properties, and next, in Section 7.5, we use them as a base for our attack.

PROPERTY 1. *For label i and last layer L in a neural network model with a non-negative activation function, when $\nabla \mathbf{W}_L^i < 0$, label i is present in the training batch on which gradient descent was applied*[1].

1 This property is a generalization of the main observation in [313].

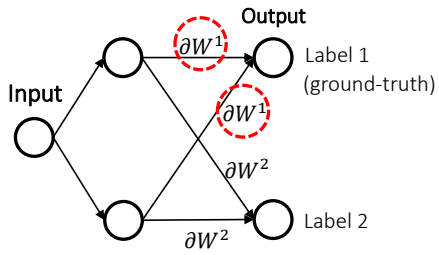


Figure 7.2: Graphical representation of a basic NN model and the gradients ∇W_L^i of the last layer L. For simplicity, the input layer is represented by a single neuron†.

Proof. We consider an NN model for a classification task. The model is trained using the cross-entropy loss over labels encoded with a one-hot encoding. This loss function l is defined as $l(\mathbf{x}, c) = -\ln \frac{e^{y_c}}{\sum_j e^{y_j}}$, where $\mathbf{x}$ is a multidimensional input instance and c represents the ground-truth label of $\mathbf{x}$. While the output vector of the model is $\mathbf{y} = [y_1, y_2, ..., y_n]$, where each $y_i \in \mathbf{y}$ is the score predicted for the i^{th} class, y_c is the score assigned to the ground-truth label, and n is the total number of classes. A graphical representation of a simple NN model and its gradients of the last layer is depicted in Figure 7.2.

Given a batch size B, we have a set $\mathbf{X}$ of B samples and the set of their ground-truth labels C. Thus, we can define a training batch as a set composed of the pairs $\{(\mathbf{x}_1, c_1), \ldots, (\mathbf{x}_B, c_B)\}$. Therefore, we can redefine the loss function as the loss $l(\mathbf{x}, c)$ averaged over a batch of B labeled samples

$$l(\mathbf{X}, C) = -\frac{1}{B} \sum_{k=1}^{B} \ln \frac{e^{y_{c(k)}}}{\sum_j e^{y_{j(k)}}}, \tag{7.1}$$

where $c(k)$ is the ground-truth label for the k^{th} sample in the batch, and $y_{c(k)}$ is the corresponding output score when $\mathbf{x}_k$ is given as an input to the model. We note that the gradient d_i of the loss w.r.t. an output y_i is

$$d_i = \frac{\partial l(\mathbf{X}, C)}{\partial y_i} = -\frac{1}{B} \sum_{k=1}^{B} \left(\frac{\partial \ln e^{y_{c(k)}}}{\partial y_i} - \frac{\partial \ln \sum_j e^{y_{j(k)}}}{\partial y_i} \right) \tag{7.2}$$

$$= -\frac{1}{B} \sum_{k=1}^{B} \left(\mathbb{1}(i = c(k)) - \frac{e^{y_{i(k)}}}{\sum_j e^{y_{j(k)}}} \right), \tag{7.3}$$

where $\mathbb{1}(\alpha = \beta) = 1$ if $\alpha = \beta$, $\mathbb{1}(\alpha = \beta) = 0$ otherwise.

$$d_i = -\frac{1}{B} \sum_{k=1}^{B} \mathbb{1}(i = c(k)) + \frac{1}{B} \sum_{k=1}^{B} \frac{e^{y_{i(k)}}}{\sum_j e^{y_{j(k)}}} \tag{7.4}$$

$$= -\frac{\lambda_i}{B} + \frac{1}{B} \sum_{k=1}^{B} \frac{e^{y_{i(k)}}}{\sum_j e^{y_{j(k)}}}, \tag{7.5}$$

where λ_i is the number of occurrences (frequency) of samples with label i in the training batch. When $i \notin C$, $\lambda_i = 0$, and $\frac{e^{y_i}}{\sum_j e^{y_j}} \in (0, 1)$, thus, $d_i \in (0, 1)$. Instead,

when $i \in C$, we have $-\frac{\lambda_i}{B} \leqslant d_i \leqslant 1 - \frac{\lambda_i}{B}$. Hence, if the gradient d_i is negative, we can conclude that label $i \in C$. Of course, the d_i value moves in this range accordingly to the status of the network weights optimization, e.g. if $i \in C$ and the network performs poorly, then, d_i will be closer to $-\frac{\lambda_i}{B}$. However, the gradients $\mathbf{d}$ w.r.t. the outputs $\mathbf{y}$ are usually not calculated or shared in FL, but only $\nabla \boldsymbol{W}$, the gradients w.r.t. the model weights $\boldsymbol{W}$. We write the gradient vector $\nabla \boldsymbol{W}_L^i$ w.r.t. the weights $\boldsymbol{W}_L^i$ connected to the i^{th} output representing the i^{th} class confidence in the output layer as follows

$$\nabla \boldsymbol{W}_L^i = \frac{\partial l(\mathbf{X}, C)}{\partial \boldsymbol{W}_L^i} = \frac{\partial l(\mathbf{X}, C)}{\partial y_i} \cdot \frac{\partial y_i}{\partial \boldsymbol{W}_L^i} \tag{7.6}$$

$$= d_i \cdot \frac{\partial (\boldsymbol{W}_L^{i\,\mathsf{T}} \mathbf{a}_{L-1} + b_L^i)}{\partial \boldsymbol{W}_L^i} \tag{7.7}$$

$$= d_i \cdot \mathbf{a}_{L-1}\,, \tag{7.8}$$

where $\mathbf{y} = \mathbf{a}_L$ is the activation function of the output layer, b_L^i is the bias, and $y_i = \boldsymbol{W}_L^{i\,\mathsf{T}} \mathbf{a}_{L-1} + b_L^i$. When non-negative activation functions (e.g. Sigmoid or ReLU) are used, $\mathbf{a}_{L-1}$ is non-negative. Consequently, $\nabla \boldsymbol{W}_L^i$ and d_i have the same sign. Considering Eq. (7.5), we conclude that negative $\nabla \boldsymbol{W}_L^i$ indicates that the label i is present in the ground-truth labels set C of the training batch. However, a present label can have a positive gradient according to the value of d_i as discussed earlier. □

PROPERTY 2. *In untrained models, the magnitude of the gradient* $g_i = \mathbb{1}^\mathsf{T} \cdot \nabla \boldsymbol{W}_L^i$ *is approximately proportional to the number of occurrences* λ_i *of label* i *in the training batch.*

Proof. Based on Eq. (7.8), we have

$$g_i = \mathbb{1}^\mathsf{T} \cdot \nabla \boldsymbol{W}_L^i = d_i \left(\mathbb{1}^\mathsf{T} \cdot \mathbf{a}_{L-1}\right)\,. \tag{7.9}$$

We substitute d_i with its expression from Eq. (7.5) as follows

$$g_i = \left(-\frac{\lambda_i}{B} + \frac{1}{B} \sum_{k=1}^{B} \frac{e^{y_{i(k)}}}{\sum_j e^{y_{j(k)}}}\right) \left(\mathbb{1}^\mathsf{T} \cdot \mathbf{a}_{L-1}\right)\,. \tag{7.10}$$

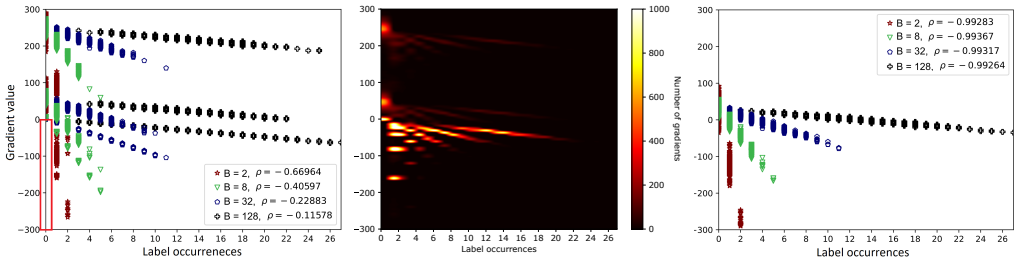
When $\sum_{k=1}^{B} \frac{e^{y_{i(k)}}}{\sum_j e^{y_{j(k)}}}$ is close to zero, we can write

$$g_i \approx -\frac{\lambda_i}{B} \left(\mathbb{1}^\mathsf{T} \cdot \mathbf{a}_{L-1}\right)\,, \tag{7.11}$$

thus, g_i is proportional to λ_i. We denote m to be

$$m = -\frac{\mathbb{1}^\mathsf{T} \cdot \mathbf{a}_{L-1}}{B}\,, \tag{7.12}$$

therefore, $g_i \approx \lambda_i m$. We define the parameter *impact* m as *the change of the gradient value caused by a single occurrence of a label in the training batch*. This value is negative and constant across labels, thus, label-agnostic.



(a) Gradients distributin **(b)** Gradients heatmp **(c)** After calibratin

Figure 7.3: Distribution of gradients obtained from a randomly initialized CNN on a batch of samples of MNIST varying the batch size in $\{2, 8, 32, 128\}$: (a) the distribution shows the correlation between the gradients and the label occurrences, (b) heatmap shows that the majority of the gradients have negative values when the corresponding label is present in the batch, (c) gradients after calibration exhibit a more prominent correlation with label occurrences. Given this strong correlation, it is possible to accurately estimate the label occurrences in the training batch basing on the gradient values[†].

However, for an untrained model, the value of $\sum_{k=1}^{B} \frac{e^{y_{i(k)}}}{\sum_j e^{y_{j(k)}}}$ strongly depends on the model weight initialization. The predication score y_i can be randomly distributed around a uniform random guess $P = 1/n$. The more classes exist in the dataset, the lower is the value of P, thus, the aforementioned summation goes closer to zero. In some cases, y_i might be notably high, although the label i is not present in the training batch. This comes as a result of misclassification and leads to a positive shift in the gradient values. We call this shift *offset* s, and based on Eq. (7.10), we can write

$$s_i = \left(\frac{1}{B} \sum_{k=1}^{B} \frac{e^{y_{i(k)}}}{\sum_j e^{y_{j(k)}}} \right) \left(\mathbb{1}^T \cdot a_{L-1} \right) . \tag{7.13}$$

The offset value varies from one label to another, hence it is a label-specific value. Using our defined parameters impact m and offset s_i, we can reformulate Eq. (7.10) as follows $g_i = \lambda_i m + s_i$. From this equation, it follows easily that the number of occurrences λ_i of label i can be derived from the parameters m, s_i, and g_i.

□

To demonstrate the two gradient properties, we randomly initialized the weights of a CNN composed of three convolutional layers. Then, we check the gradients g_i by evaluating the network on a batch of samples taken from the MNIST dataset [145], which contains 10 classes. We repeat the experiment $1,000$ times with different batch sizes $B \in \{2, 8, 32, 128\}$. Figure 7.3 (a) depicts the distribution of the resulting gradients, where each data point represents the gradient value of one label in one experiment. The y-axis shows the gradient values and the x-axis represents the number of occurrences for a label $i : \forall i \in [1, n]$.

We can see that there are no negative gradients at $\lambda_i = 0$ (framed in red), in other words, the negative gradients always correspond to an existing label in the batch $\lambda_i > 0$, which confirms Property 1. For all the batch sizes, we

notice that the values of the gradients decrease consistently with the increase of the occurrences. This, in turn, confirms Property 2 and our definition of the impact parameter. We also observe that the decrease of the gradient values is roughly constant regardless of the label. This confirms the impact being label-agnostic, as we described earlier. Furthermore, we notice that the magnitude of the impact is negatively correlated with the batch size. Meaning, the more samples are present in a batch, the smaller are the changes of the gradients for a different number of occurrences. This is also clear from the definition of impact in Eq. (7.12). We also can see that there are positive gradients that correspond to $\lambda_i > 0$. The positive value of these gradients is mainly caused by the offset s_i defined in Eq. (7.13). To illustrate their ratio, we depict a heatmap in Figure 7.3 (b). We observe that only a subset of the gradients (18%) are positive, i.e., shifted by the offset, while the majority of the gradients have negative values when the corresponding labels are present in the batch. In Section 7.5.1, we describe our methods to estimate the offset and elaborate on Figure 7.3 (c).

7.5 LABEL EXTRACTION

In this section, we present our attack, Label Leakage from Gradients (LLG), to extract the ground-truth labels from shared gradients. We first introduce different methods to estimate our attack parameters, impact and offset. Then, we explain the attack.

7.5.1 *Attacking parameters estimation*

In the light of the three different threat models outlined in Section 7.2.1, we empirically developed several heuristic methods to estimate the impact and offset.

SHARED GRADIENTS. In this scenario, the adversary has access only to the shared gradients. As mentioned earlier, the impact refers to the change in the value of the gradients corresponding to one occurrence of a label. Our intuition is that a good estimation for the impact is obtained by averaging the gradients over the number of data samples $|\mathbf{D}|$ used by a user in a training round. For FedSGD, $|\mathbf{D}| = B$ the batch size, while for FedAvg, $|\mathbf{D}| = \gamma.B$, where γ is the number of local iterations (batches). Based on Property 1, we know that all negative gradients are indeed indicating existing labels in the training samples. Therefore, we average only the gradients with negative values. Consequently, this average is an underestimation since some gradients may be positive because they are shifted with an offset. We empirically observed that multiplying by a factor that depends on the total number of classes n is a good additive correction, precisely, we multiply by $(1 + 1/n)$. Thus, we estimate the impact m as follows

$$m = \frac{1}{|\mathbf{D}|} \sum_{i:g_i<0}^{n} (g_i) \left(1 + \frac{1}{n}\right) . \tag{7.14}$$

For this threat model, we could not estimate the offset s_i and therefore it is considered to be zero in the attack.

WHITE-BOX MODEL. When the adversary additionally has access to the model architecture and parameters, they can use it to generate more gradients and gain more insights about the behavior of the gradients in this model. Consequently, better estimations for the impact and offset can be achieved. The approximation in Eq. (7.11) indicates that the impact m can be estimated if the gradient g_i and number of occurrences λ_i are known, regardless of the quality of the input data. Thus, dummy data samples, e.g., dummy images of zeros (black), ones (white), or random pixels, can be used to generate g_i under known λ_i. More precisely, we form a collection of dummy batches, each batch contains data samples assigned to one label i. For impact estimation, we pass these batches to a shadow model (a copy of the original model), one at a time, and calculate the average $\bar{g}_i$ for all the batches corresponding to each label $i \in [1, n]$. Then, we average over all classes n and the batch size B as follows

$$m = \frac{1}{nB} \sum_{i=1}^{n} (\bar{g}_i) \left(1 + \frac{1}{n}\right) . \tag{7.15}$$

As mentioned earlier, we assume the offset s_i to be an approximation of misclassification penalties, when the model mistakenly predicts i to be ground-truth. These penalties are mainly related to the status of the model weights, which can be biased to specific classes. Based on this intuition, we estimate the offset s_i by passing batches full of other labels $\forall j \in [1, n] : j \neq i$, each batch full of one label, one batch per run. We repeat this for various batch sizes, in total of z runs. In these runs, the gradients of label i reflect to some extent the misclassification penalties. Therefore, we calculate the mean of these gradients to be our estimated offset, thus, we have

$$s_i = \frac{1}{z} \sum_{k=1}^{z} (g_{i_k}) . \tag{7.16}$$

AUXILIARY KNOWLEDGE. In this scenario, the adversary is able to access the shared gradients, model, and auxiliary data that contains the same classes as the training dataset. Here, the adversary can follow the same methods as for the white-box scenario, however, using real input data instead of dummy data. This in turn is expected to yield better estimations for the impact and offset. The goodness of the auxiliary data, i.e., the similarity of the content and class distribution to the original dataset, might play a role in the quality of the estimations. This aspect can be investigated in further research.

To demonstrate the quality of our offset estimation, we calibrate the gradients of Figure 7.3 (a) by subtracting the estimated offset and plot the results in Figure 7.3 (c). We can see how the gradients become mainly negative and strongly correlated with the label occurrences. To measure the correlation, we use the Pearson correlation coefficient $-1 \leqslant \rho \leqslant 1$ [16], which yields, for all the studied batch sizes, values of $|\rho| > 0.99$. The calibration process mitigates the

Algorithm 4: Label Leakage from Gradients (LLG)†

Data: $\mathbf{G} = [g_1, .., g_n]$: vector of gradients, m: impact, $\mathbf{S} = [s_1, .., s_n]$: vector of offsets, $\mathbf{D}$: data samples used to generate $\mathbf{G}$.
Result: E: list for extracted labels.

```
1  for g_i ∈ G do
2      if g_i < 0 then
3          append i to E;
4          g_i ← g_i − m;
5      end
6  end
7  G ← G − S;
8  while |E| < |D| do
9      Select g_i : g_i = min(G);
10     append i to E;
11     g_i ← g_i − m;
12 end
```

effect of the offset and makes the gradient values more consistent, thus, easier to be used for extracting the labels.

7.5.2 *Label leakage from gradients attack*

LLG extracts the ground-truth labels from gradients by exploiting Property 1 and 2 unfolded in the beginning of Section 7.4. The attack consists of three main steps summarized in Algorithm 4.

1 We start with extracting the labels based on the negative values of the gradients (Property 1). Thus, the corresponding label of each negative gradient is added to the list of the extracted labels E. As Property 1 holds firm in our problem setting, we can guarantee 100% correctness of the extracted labels in this step. In preparation for the next step, every time we add a label to E, we subtract the impact from the corresponding gradient following Property 2 (Lines 1-5).

2 We calibrate the gradients by subtracting the offsets. In case the offsets (elements of vector S) are not estimated, they are considered to be zeros. This step increases the correlation between the gradient values and label occurrences, which facilitates better label extraction based on these values (Line 7).

3 After calibration, the minimum gradient value (negative with maximum magnitude) is more likely to correspond to a label occurred in the batch (see Figure 7.3 (c)). Therefore, we select the minimum and add the corresponding label to the extracted labels. Each of the data samples used to generate the gradients has a ground-truth label, which we aim to extract. Thus, we repeat Step (3) until the size of the extracted labels list $|E|$ matches the number of the data samples $|D|$. Assuming that $|D|$ is known or can be guessed by the adversary (Lines 8-11).

Finally, the output of the LLG attack is the list of extracted labels E, precisely, the labels existing in the batch and how many times they occur.

7.6 EMPIRICAL EVALUATION

We evaluated the effectiveness of LLG with varying settings including: different FL algorithms, threat models, model architectures, and model convergence statuses. We also tested the robustness of LLG against two defense mechanisms, namely noisy and compressed gradients. For the sake of simplicity, we refer to $g_i = \mathbb{1}^T \cdot \nabla W_L^i$ as the gradient of label i in the rest of this section. Next, we describe the experimental setting, then we discuss our results. The source code of the experiments can be found on GitHub[2].

7.6.1 *Experimental setup*

DEFAULT MODEL. We used a CNN model with three convolutional layers (see Appendix C) as our default model for a classification task. The activation function is Sigmoid, and we used SGD as an optimizer with a learning rate 0.1 and cross-entropy as a loss function. We used batches of varying sizes $B = 2^k : k \in [0, 7]$. When applying the attack for FedSGD, we fed the model with one batch, and we used 10 batches for FedAvg. The label distribution in a batch can be *balanced* or *unbalanced*. For balanced data, the samples of the batch were selected randomly from the dataset. For unbalanced data, we selected 50% of the batch samples from one random label i and 25% from another label j. The remaining 25% of the batch was chosen randomly. We initialized the model with random weights and repeated each experiment 100 times, then reported the mean values for analysis and discussion.

DATASETS. We conducted our experiments on four widely used benchmark datasets: MNIST [145] consists of 70,000 grey-scale images for handwritten digits, with 10 classes in total. SVHN [188] has 99,289 color images of house numbers with 10 classes. CIFAR-100 [140] contains 60,000 color images with 100 classes. And CelebA [162] is a facial attributes dataset with 202,599 images. In our experiments, we consider only the hair color attribute with 5 classes.

THREAT MODEL. We assumed the users to train the model on real data and correct labels. The adversary has access to the shared gradients of only one target user. We considered three different scenarios for the observation capabilities of the adversary (see Section 7.2). Based on these scenarios, the estimation of the impact and offset parameters differs (see Section 7.5.1), while the same attack applies for all. We refer to the application of the attack under these different scenarios as follows:

1 *LLG* for accessing only the shared gradients scenario.

2 *LLG** for the white-box model, where we employed various dummy images to estimate the impact and offset. Empirically, we observed the dummy images with which the attack achieves better performance on each dataset. This resulted in using zeros (black) images for MNIST, random pixels for SVHN, ones (white) for CIFAR, and zeros (black) for CelebA.

3 *LLG+* for auxiliary knowledge, where it is assumed that the adversary has access to auxiliary data that contains 10 batches of images from each class.

2 https://github.com/tklab-tud/LLG

METRICS. To measure the attack effectiveness, we used the Attack Success Rate (ASR) metric [289], which is expressed as the ratio of the correctly extracted labels over the total number of the extracted labels. We also employed the Hellinger distance [46] to measure the distance between the distribution of the extracted labels $E = \{e_1, ..., e_n\}$ and the ground-truth $Q = \{q_1, ..., q_n\}$, where e_i is the ratio of extracted labels as class i to the total number of extracted labels. Similarly, we consider q_i for the ground truth labels. The Hellinger distance is measured as follows

$$H(E, Q) = \frac{1}{\sqrt{2}} \sqrt{\sum_{i=1}^{n} (\sqrt{e_i} - \sqrt{q_i})^2}. \tag{7.17}$$

However, during our experiments, we observed that both aforementioned metrics yielded very similar measurements, therefore, we present our results only with the ASR metric.

BASELINES. We compared LLG with two baselines. First, the DLG attack [316], which aims to reconstruct the training data and labels using an optimization approach. For our experiments, we ran DLG for 100 iterations and focus only on the label reconstruction results. We used the DLG implementation provided by Zhao et al. [313][3]. Second, we considered a uniform distribution-based random guess as a baseline. An adversary without any shared gradients might partially succeed in guessing the existing labels frequency. This can be achieved by assuming that the labels distribute uniformly, especially in the case of large balanced batches. The random guess serves as a risk assessment curve. If any attack performs better than the random guess, this means that there is information leakage.

7.6.2 *Attack success rate*

We ran our experiments under two FL algorithms, namely FedSGD and FedAvg [175]. For FedSGD, we passed one batch to the model and attacked the generated gradients, while for FedAvg, we fed the model with 10 batches and attack the aggregated gradients, i.e., the sum of the gradients over 10 iterations. During our experiments, we observed a very limited difference in the ASR of the LLG attacks for balanced and unbalanced batches, therefore, and because the unbalanced data is closer to real-world scenarios [175], we focus on presenting the results of the unbalanced data case.

FEDSGD. Figures 7.4 (a-d) illustrate the ASR scores (y-axis) with batches of different sizes (x-axis). We can see that all LLG variants show some level of ASR degradation when the batch size is increased. However, it appears to be stabilized to some extent for bigger batches, e.g., 64 and 128. That is due to the fact that the first step of the algorithm (see Section 7.5.2) is based on Property 1 and yields 100% correct labels. This step extracts a maximum of n labels. Thus, its

3 https://github.com/PatrickZH/Improved-Deep-Leakage-from-Gradients

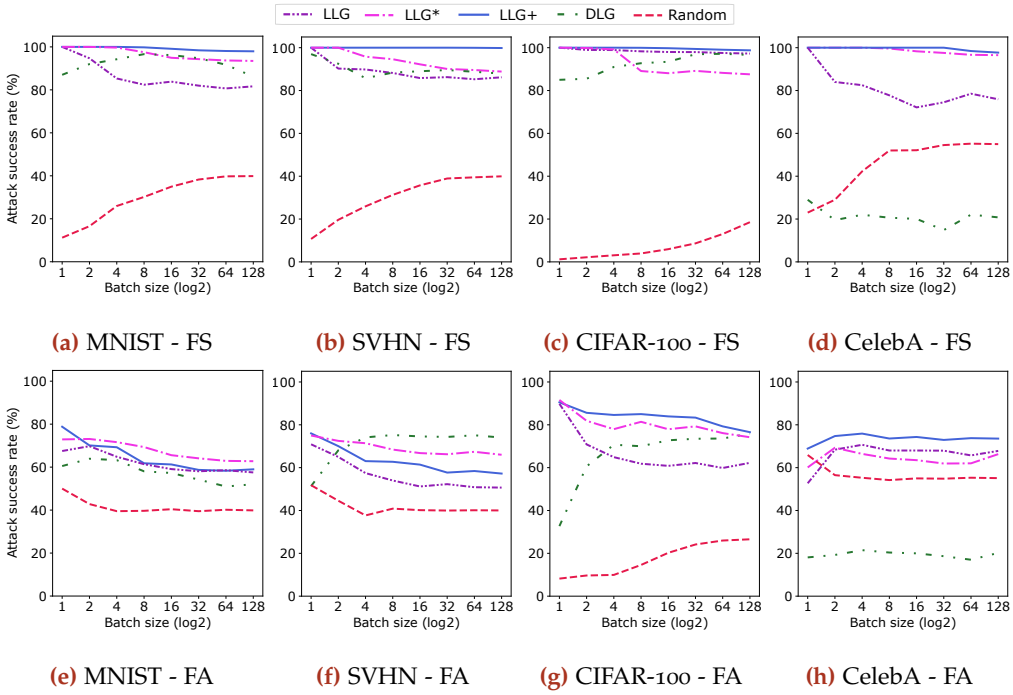


Figure 7.4: Attack success rate of (1) LLG with shared gradients, (2) LLG* with white-box model, (3) LLG+ with auxiliary knowledge, (4) DLG [316], and (5) random guess on MNIST, SVHN, CIFAR-100, and CelebA. Label extraction is based on gradients generated from passing (1) one batch for FedSGD (FS) (first row), and (2) 10 batches for FedAvg (FA) (second row), to a randomly initialized CNN. DLG runs for 100 iterations. LLG methods outperform the baselines in most of the cases†.

results are dominant when $B \leqslant n$. Where as if $B > n$, the second and third steps, which are based on heuristic estimations, contribute more to the final extracted labels. As a result, we notice a degradation of the ASR. However, the different batch sizes do not seem to massively affect the correctness of the results of these steps. This might be explained by the fact that the batch size B is always considered as a parameter in the heuristic estimations of the impact and offset.

Overall, LLG+ outperforms all the other LLG variants and DLG. The LLG and LLG* scores range from 100% to a minimum of 77% across the different datasets, whereas LLG+ remarkably exhibits a high level of stability for various batch sizes and numbers of classes (in datasets) with an ASR $> 98\%$. This mainly reflects the quality of our estimation methods for impact and offset.

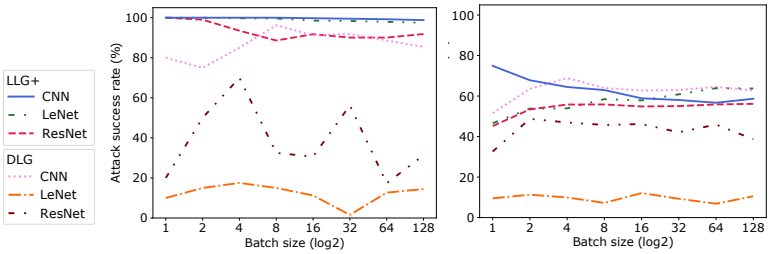
In contrast, DLG achieves varying accuracy scores. However, no clear behavior can be concluded w.r.t. the changes in the batch sizes. This might be due to the fact that DLG requires a training phase, which is highly sensitive to model initialization, i.e., it might fail to converge for some randomly initialized models or it might require different periods of time for reaching a specific accuracy. Unlike LLG, which yields more deterministic results, while at the same time being orders of magnitude faster. For example, the execution time of those experiments illustrated in Figure 7.4 (a) is as follows: LLG 54s, LLG* 32.2m, LLG+ 14.6m, DLG 17.4h, and Random 50s, using a Tesla GPU V100-SXM3-32GB. It is

worth mentioning that LLG* requires more time than LLG+ due to the dummy images generation.

The ASR of each LLG attack is similar to some extent on MNIST and SVHN respectively. This can be explained by the fact that both datasets have the same number of classes, i.e. 10. On CIFAR-100 (100 classes), interestingly, we notice that LLG performs quite closely to LLG+ (both have ASR $> 96\%$), as shown in Figure 7.4 (c), while it drops to around 75% on CelebA (5 classes). This observation suggests that LLG performs better for datasets with a bigger number of classes. This can be explained by the fact that LLG solely depends on the quality of the impact parameter, which is derived from Eq. (7.11) under the assumption that the untrained model performs poorly. This assumption is more valid when the number of classes is bigger, as we explained earlier in the proof of Property 2, Section 7.4. Therefore, the estimation of the impact yields better results leading to higher ASR.

LLG*, with its dummy data for the parameter estimation, shows a notable drop on CIFAR-100. It is known that the complexity of CIFAR-100 images is higher than the one of MNIST and SVHN. Therefore, we can conclude that the complexity of the dataset might influence LLG* in a negative way, while it has no observable effect on LLG and LLG+. For DLG, we notice in Figure 7.4 (d) a remarkable decrease in accuracy on CelebA. This can be due to the fact that the images are of higher dimensions (178×218), unlike the other datasets. Thus, the convergence of the attack is much more difficult.

FEDAVG. In Figures 7.4 (e-h), we can see that the ASRs of all the LLG variants considerably decrease compared to FedSGD, ranging between 55% and 90%. This is expected as the shared gradients are generated from multiple iterations (10 batches). Thus, the correlation between the gradient values and label occurrences is less prominent. In other words, the gradients are accumulated several times over iterations, such that the correlation (demonstrated through Property 1 and 2) becomes more difficult to detect and exploit. However, the LLG attacks achieve higher ASRs than the random guess on all the datasets, thus, they are still posing a serious threat. The superiority of LLG+ is maintained on CIFAR-100 and CelebA, while on MNIST and SVHN, LLG* performs better. This observation raises a question about the influence of the quality of the data (images) used for parameter estimation. Although it would be expected that using images similar to the original dataset (in LLG+) would lead to better estimates than using dummy images (in LLG*), this result shows the opposite. This could be related to the process of image selection for LLG+. Finally, we notice that on SVHN, DLG outperforms the LLG variants. In some cases, the random initialization could allow the model in DLG to converge quicker so that the attack performs better. These cases could be the reason for this result. However, further investigation is needed to find a concrete explanation.



(a) FS **(b)** FA

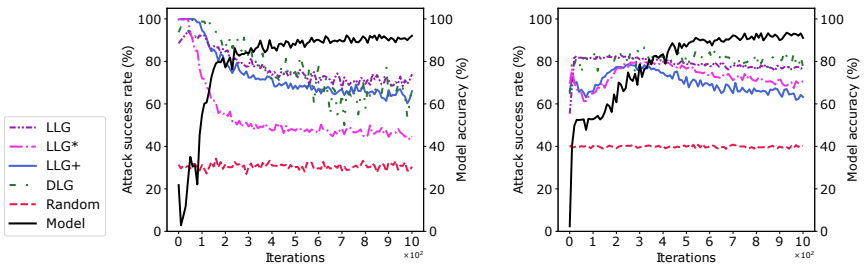
Figure 7.5: Attack success rate of LLG+ and DLG on unbalanced batches of different sizes from MNIST with different model architectures: CNN, LeNet, ResNet20. For FedSGD (FS), LLG+ achieves around 100% accuracy on CNN and LeNet while its accuracy slightly decreases on ResNet20. DLG achieves $> 80\%$ for most batch sizes on CNN, while it drops remarkably on the more complex architectures, LeNet and ResNet20. For FedAvg (FA), the ASR of LLG+ is slightly different form architecture to another, while DLG shows higher sensitivity to the architectures[†].

7.6.3 *Model architecture*

Here, we study the influence of the model architecture on the studied attacks; for that, we considered two models besides our default CNN: (1) LeNet [145], a basic CNN that contains 3 convolutional layers with 2 maximum pooling layers as shown in Appendix C. (2) ResNet20 [109], a successful residual architecture with convolutions, which introduces the concept of "identity shortcut connection" that skips one or more layers to avoid the problem of vanishing gradients in deep neural architectures. ResNet20 contains 20 layers in total: 9 convolutional layers, 9 batch normalization, and 2 linear layers. Both aforementioned architectures, alongside their principal components, namely convolutions and residual blocks, have achieved and contributed to the state-of-the-art results on several classification tasks.

The two main conditions for Property 1 to hold are: (1) using the cross-entropy loss and (2) having a non-negative activation function in the last layer before the output. Thus, we assumed that the labels extracted in the first step of the attack (see Algorithm 4, Line 1-5) based on this property are correct regardless of the rest of the model architecture. On contrast, the next steps of LLG are based on the impact, offset, and their estimations, which might be of different accuracy from one model architecture to another. To run our analysis, we used MNIST with varying batch sizes and measured the ASR of LLG+ and DLG.

FEDSGD. As we can see in Figure 7.5 (a), LLG+ performs best on CNN and LeNet, achieving approximately 100% of success rate, while a degradation starts from batches with size > 2 for ResNet20. This might be due to the residual blocks in the ResNet20 architecture that prevents the vanishing gradients problem in deep neural networks. In other words, ResNet20 implicitly alters and controls the range of the gradient values in order to not let them vanish (gradients tending towards zero) or explode (gradients tending towards infinity) during training. This conflicts with our definitions of the impact and offset parameters in Eq. (7.12) and (7.13) and thus has direct implications on the attack performance.



(a) FS **(b)** FA

Figure 7.6: Influence of model convergence status on ASR of LLG, LLG*, LLG+, DLG, and random guess for CNN with unbalanced batches from MNIST dataset. On the left y-axis it is plotted the attack success rate, while on the right y-axis, it is plotted the model test accuracy. The number of training iterations ($\times 10^3$) is on the x-axis. All different LLG methods achieve remarkable success rates even if the models are well-trained and gradients become less informative[†].

On the other hand, DLG shows much higher sensitivity towards the model architecture. As we can see, it achieves $> 80\%$ for most batch sizes on CNN, while it drops remarkably on the more complex models, LeNet and ResNet20. A strong influence of the model architecture complexity on DLG is highly likely since DLG includes an optimization phase, where optimizing complex models typically requires many more iterations.

FEDAVG. In Figure 7.5 (b), we observe that under small batch sizes $B \leqslant 16$, the ASR of LLG+ is higher for CNN. While for bigger batches, LLG+ only slightly differs from one model to another. This supports the (grounded) hypothesis that the model architecture has limited effect on LLG+. In contrast, DLG shows again higher sensitivity with bigger variance of the ASR over the different architectures.

7.6.4 *Model convergence status*

The gradients guide the model towards a local minimum of the loss function. As the model converges to this minimum, the information included in the gradients becomes less prominent. Therefore, we expect the convergence status of the model to have a strong influence on the attack effectiveness. All the previous experiments are conducted in one communication round, i.e., the gradients are generated and shared with the server only once. In this section, we went further with training the model and observed the implications on the attack.

We trained the model in a federated setting, where the data of MNIST was distributed among 750 users, each having 80 unbalanced data samples. The server randomly selected 100 users for every communication round to train the global model locally and share their gradients. The CNN model was trained with batches of size 8 for 10^3 iterations. We chose the batch size 8 to be able to apply the DLG attack in its most effective setting $B \leqslant 8$ [316]. In every communication round, we attacked the shared gradients of one target user (victim) with DLG and LLG variants, where the impact and offset were estimated dynamically.

Figures 7.6 (a,b) depict the attacks ASR (on the left y-axis) versus the model accuracy at testing time (on the right y-axis), while the x-axis represents the number of training iterations.

FEDSGD. We can see in Figure 7.6 (a) that the growth of the model accuracy incurs a notable decrease of the ASR for all LLGs. However, although the model converges to close to 90% accuracy, LLG and LLG+ keep achieving ASR $> 60\%$, considerably higher than the random guess which is around 32%. Meaning, the attacks are still able to take advantage of the reduced information in gradients over the course of the whole training process. Similarly, DLG shows degradation in accuracy, yet it remains effective for well-trained models.

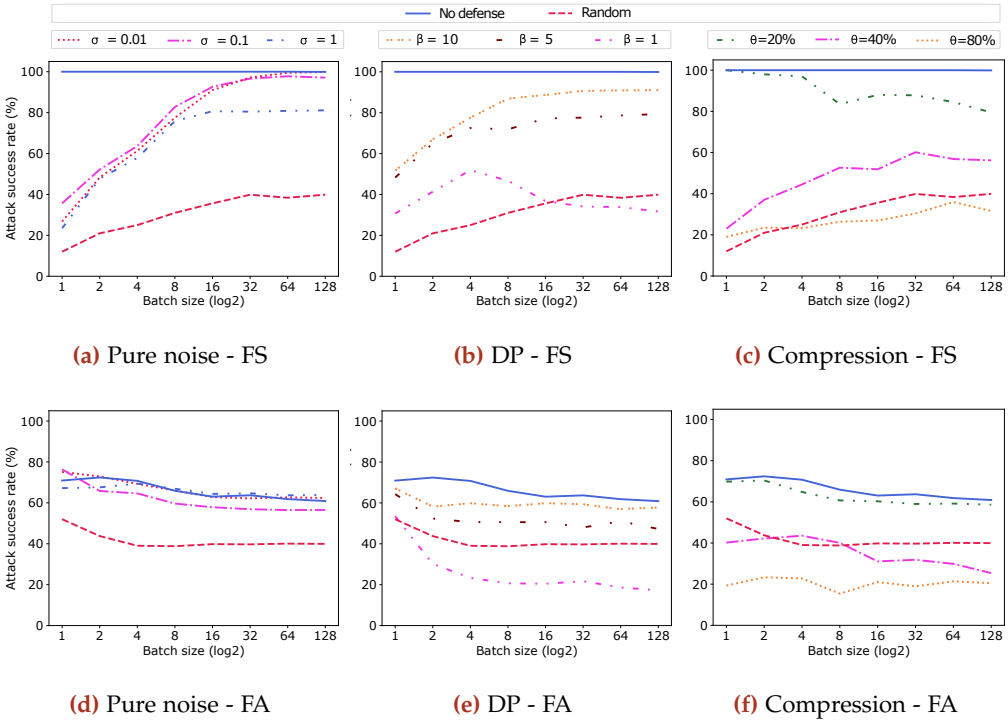
FEDAVG. Figure 7.6 (b) shows more stability of ASR over the training process, especially for LLG and DLG. Even in the early stages of the training, the multiple local iterations in FedAvg improve the model accuracy making the accumulative gradients less informative. Therefore, the attacks start with lower ASR compared to FedSGD. However, this leads also to mitigating the notable degradation of ASR observed in Figure 7.6 (a). LLG* and LLG+ exhibit volatile behavior in the early iterations, where they have an increasing success rate between iteration 100 and 300. Then, they decrease again from 80% to close to 70% and 60%, respectively. Interestingly, DLG maintains a high success rate (around 80%) outperforming the LLG variants in most parts of the training process. This shows that DLG is less sensitive to convergence status under FedAvg and thus can cope with the decreasing amount of information in the gradients. Overall, the attacks stay effective with ASR $> 40\%$, which is the random guess success rate.

7.6.5 *Defense mechanisms*

As LLG is mainly based on the gradients, thus, sensitive to changes in their values, obfuscating them can be a direct mitigation mechanism. In this section, we used two obfuscation techniques: noisy gradients and gradients compression. We applied these techniques on the user side before sharing the gradients with the server and thus, protecting against external eavesdroppers and curious servers. Then, we attacked the gradients of one target user in one communication round for a randomly initialized CNN model (untrained). In general, applying obfuscation techniques incurs a loss in the model accuracy. To cover this aspect, we trained the model to convergence under conditions similar to those in Section 7.6.4 while applying the defenses, and reported its accuracy.

7.6.5.1 *Noisy gradients*

Many researchers consider adding noise to gradients as the de facto standard for privacy-preserving ML [149]. In this experiment, we evaluated LLG+ against two techniques of noise addition: (1) Pure noise: we added noise to gradients before sharing, similar to [289, 316], where no formal privacy properties are guaranteed. (2) Differential privacy: following differentially private FL [86], we clipped the gradients to bound their sensitivity, then, we added noise to them.



(a) Pure noise - FS **(b)** DP - FS **(c)** Compression - FS

(d) Pure noise - FA **(e)** DP - FA **(f)** Compression - FA

Figure 7.7: Effectiveness of different defenses against LLG+ on an unbalanced batch from MNIST with a randomly initialized CNN: (a) defense by adding Gaussian distributed noise to gradients with variances $\sigma \in \{0.01, 0.1, 1\}$, (b) defense by user-side differential privacy with $\sigma = 0.1$ and clipping bound $\beta \in \{1, 5, 10\}$, (c) defense by pruning gradients with varying compression ratios $\theta \in \{20\%, 40\%, 80\%\}$. Pure noise is not successful in eliminating the risk completely, since LLG+ maintains a higher ASR than the random guess. While, differential privacy mitigates the attack with $\beta = 1$ for FedAvg, and for FedSGD when batch size $B \geqslant 16$. Gradient compression is effective in FedSGD when a high compression ratio ($\geqslant 80\%$) is used with $B \geqslant 4$. For FedAvg, even the compression ratio 40% with $B \geqslant 8$ is an effective defense†.

The clipping is defined as $\nabla \boldsymbol{W} \leftarrow \nabla \boldsymbol{W} / \max\left(1, \frac{\|\nabla \boldsymbol{W}\|_2}{\beta}\right)$, where β is the gradient norm bound. In both noise addition techniques, we used the Gaussian noise distribution. For pure noise, the standard deviation of the noise distribution is $\sigma \in \{0.01, 0.1, 1\}$ with central 0. For differential privacy, we used $\sigma = 0.1$ and varying norm bound $\beta \in \{1, 5, 10\}$. We tracked the privacy loss for the model trained with differential privacy using the moments accountant [2]. For 100 communication rounds and $\delta = 10^{-5}$, the privacy budget is estimated to be $\epsilon \approx 11.5$.

FEDSGD. In Figure 7.7 (a), we can see that the higher the magnitude of the noise the less accurate the attack. This is expected as the attack partially uses the magnitude of the gradients to infer the ground-truth labels following Property 2. Interestingly, we observed that the noise has less effect on the attack when the batch size is increasing. We investigated this observation further by inspecting the values of the gradients before and after noise addition. Our empirical analysis showed earlier in Figure 7.3 (a, b) that the majority of gradients g_i have values close to zero when they correspond to labels not present in the batch. Adding noise to such small gradient values might lead to flipping their

sign, and consequently, disrupting Property 1, which is one of the basis of the attack. The signs of the gradients can be flipped more often by the noise when the batches are of smaller sizes $B < n$ (n is the number of classes) since not all the labels will be present in the batch. Therefore, a clear influence of the noise on the success rate of the attack can be seen. In bigger batches $B \geqslant n$, it is more likely to have more differing labels, thus, their gradients values are not close to zero. As a result, adding a small amount of noise does not lead to sign flipping. This also explains the stability of ASR values when $B \geqslant n$. Overall, adding noise does not eliminate the risk completely while reducing the model accuracy (see Table 7.1). As we can see, LLG+ maintains higher ASRs than the random guess for all the test noise variances.

Figure 7.7 (b) shows that adding noise of $\sigma = 0.1$ with clipping bound $\beta = 1$ is an effective defense against LLG+ for batch sizes $B > 16$, where the ASR drops beyond the random guess. However, this leads to a significant drop in the model accuracy (52.4%) as shown in Table 7.1.

FEDAVG. Unlike in FedSGD, the magnitude of the pure noise does not have a clear effect on ASR for FedAvg as shown in Figure 7.7 (d). That is due to the fact that the shared gradients are generated from 10 batches. Thus, the gradient values reflect $10 \times B$ labels, which is always greater or equal to n for MNIST, where $n = 10$. Therefore, it is likely that most of the labels appear in one of the batches at least, consequently, no gradient values will be close to zero. As a result, the pure noise does not impact the ASR remarkably, and LLG+ remains effective. In Figure 7.7 (e), we notice that noise $\sigma = 0.1$ with bound of $\beta = 1$ is able to mitigate LLG+, reducing its success rate to close to 20% for bigger batch sizes. However, the model accuracy degrades remarkably to 52.5%. Additionally, other differential privacy approaches, e.g., DP-SGD [2] can also be applied and investigated as a defense.

7.6.5.2 *Gradient compression*

One of the main motivations for FL is reducing the communication cost by maintaining the user data local and sharing only the gradients. However, some models might contain hundreds of millions of parameters, and sending the gradients of these parameters introduces again a very significant communication overhead. Gradient compression is one proposal to mitigate this issue [154, 263], where mainly gradients with small magnitudes are pruned to zero, while further measures are taken to avoid information loss, thus, ensuring to reach the potential model accuracy.

Pruning some gradients reduces the information that the attack exploits to extract the labels. In this set of experiments, we evaluated LLG+ under various gradient compression ratios $\theta \in \{20\%, 40\%, 80\%\}$, i.e., θ denotes the percentage of the gradients to be discarded in each communication round with the server. We used the sparsification approach proposed in [154], where users send only the prominent gradients, i.e., with a magnitude larger than a specific threshold. The threshold was calculated dynamically based on the desired compression

	FedSGD (Acc. = 93.3%)			FedAvg (Acc. = 94.5%)		
PN (σ)	0.01	0.1	1	0.01	0.1	1
Acc. (%)	93.4	89.9	$\leqslant$ 10.1	94.6	91.4	$\leqslant$ 13.5
DP (β)	10	5	1	10	5	1
Acc. (%)	89	86.1	$\leqslant$ 52.4	91.2	90.5	52.5
GC (θ%)	20	40	80	20	40	80
Acc. (%)	93.4	93.7	91.9	92.8	91.6	89.3

Table 7.1: Model accuracy while applying the defense mechanisms. PN: pure noise, DP: differential privacy, GC: gradient compression[†].

ratio. The small gradients were accumulated across multiple communication rounds and sent only when they are large enough.

FEDSGD. Figure 7.7 (c) illustrates that when the compression ratio is $\leqslant$ 20%, there is only a slight effect on the success rate of the attack. When the ratio is 80%, we notice that LLG+ becomes completely ineffective for $B \geqslant 4$, dropping beyond the random guess. Notably, the model accuracy is maintained high at 91.9% in this case. Consequently, gradient compression with $\theta > 80\%$ can practically defend against the attack while producing accurate models.

FEDAVG. Similar to FedSGD, we observe a limited effect of the ratio $\theta \leqslant 20\%$ in Figure 7.7 (f), whereas the ratio of $\theta = 40\%$ with $B \geqslant 8$ can mitigate the risk of LLG+, as well as $\theta \geqslant 80\%$ for any batch size. Under both compression ratios, the model converges at high accuracy scores, 91.6% and 89.3%, respectively. Additional improvements on the accuracy can be achieved by applying error compensation techniques, such as momentum correction and local gradient clipping, which are proposed in [154].

7.6.5.3 *Concluding considerations*

Results show that adding moderated amounts of noise to gradients alone cannot protect against our attack LLG+, while applying differentially private FL mitigates the attack, but at the cost of a remarkable loss in model accuracy. In contrast, gradient compression with high compression ratios can reduce the success rate of the attack beyond the random guess, rendering it ineffective while maintaining high model accuracy. This demonstrates the importance of gradient compression not only as an approach to efficient communication, but also as a defense mechanism against our label leakage attack.

We want to explicitly stress the fact that in addition to the aforementioned defenses, cryptography-based approaches exist [8, 22, 103, 314], which can protect gradients from external eavesdroppers and even curious servers. However, besides the computation and communication overhead introduced by these approaches, they prevent the server from evaluating the benignity of users' updates (see Section 5.3).

7.7 CONCLUSION

As discussed in the beginning of this chapter, the users' ground-truth labels can contain sensitive information, e.g., a patient's disease. Therefore, investigating the information leakage threats to the ground-truth labels is of a high importance. In this chapter, we identified and formalized two properties of the gradients of the last layer in deep neural network models trained with cross-entropy loss for a classification task. These properties reveal a correlation between gradients and label occurrences in the training batch. We proposed Label Leakage from Gradients (LLG), a novel attack that exploits this correlation and extracts the ground-truth labels from shared gradients in the FedSGD and FedAvg algorithms. We demonstrated the validity of LLG through mathematical proofs and empirical analysis.

Results demonstrated the scalability of LLG to arbitrary batch sizes and numbers of classes. Moreover, we showed the success rate of LLG on various model architectures and in different stages of training. The effectiveness of noisy gradients and gradient compression as defenses was also investigated. Findings suggested the gradient compression to be an efficient technique to prevent the attack while maintaining relatively high model accuracy. With this work, we hope to raise the awareness of the privacy risks associated with gradients sharing schemes, encouraging the community and service providers to give careful consideration to security and privacy measures in this context.

In this and the previous chapters, we elaborated on threats in FL as one of the emerging techniques for distributed machine learning. Similar threats can also be found in other distributed machine learning techniques, e.g., label leakage in split learning [148]. We discussed also several defense mechanisms against these threats, e.g., differential privacy, which is not limited to FL, but used widely in many other analytics techniques, e.g., decision trees [310]. Thus, although we focused on FL in our last two chapters, relevant aspects of privacy risks in distributed analytics in general were covered.

Part V

EPILOGUE

8 CONCLUSION

This chapter recapitulates the work of this thesis. Section 8.1 summarizes the main contributions and findings. Then, Section 8.2 discusses the future research directions.

8.1 SUMMARY OF CONTRIBUTIONS

This thesis tackled the challenge of applying privacy-enhanced distributed analytics in Online Social Networks (OSNs). First, we contributed a novel approach to privacy-friendly OSNs. Second, we presented two approaches to rendering prominent analytics techniques privacy-friendly. Third, we investigated threats against one of the most emerging distributed analytics techniques. Next, we summarize our contributions in these three research areas.

Chapter 2 provided the *Hybrid Online Social Network (HOSN)* approach, which combines the user experience and inevitable network effects of wide-spread Centralized Online Social Networks (COSNs) with the privacy-related benefits of Decentralized Online Social Networks (DOSNs). HOSN allows users to store sensitive data in a distributed fashion beyond the reach of service providers. That is achieved by leveraging P2P networks and an access control mechanism applied by the users themselves. We proved the viability of the approach via Hushtweeht, an Android prototype app building on top of Twitter and using available distributed storing systems. Hushtweet enables users to store their private tweets and anonymous *likes* on a private network, which only their followers can access. Simultaneously, Hushtweet allows users to enjoy the regular Twitter experience. With this approach, we showed that it is possible to overcome the lock-in effect of COSNs and still enhance user privacy in several aspects.

To ensure wide acceptance and efficient development of our HOSN concept and thus the Hushtweet application, understanding the privacy concerns of users and how these concerns influence their conceptions is essential. Chapter 3 studied, in the context of OSNs, the relationships between privacy concerns, trusting beliefs, risk beliefs, and willingness to use. We showed in this chapter that mitigating privacy concerns with software features in the user interface of Hushtweet made a positive impact on the trustworthiness of the application. Furthermore, we studied the effect of the user variables on these relationships. We found that the software features particularly affected older people and those with less experience regarding privacy-related issues. Interestingly, we found that the lack of awareness of privacy practices is one of the top concerns of users. Also, we showed that addressing this concern can lead to users having a better sense of control. Our detailed study showed that to alleviate users'

concerns and gain their trust, it requires software features that are carefully designed to clearly demonstrate the privacy practices of the OSN applications.

After establishing the HOSN approach, which allows users to keep their data distributed, we moved on to applying analytics on this data. In particular, Chapter 4 introduced an approach to enable users to collaboratively mine frequent itemsets in a distributed and privacy-enhanced manner. This approach is based on a combination of distributed sampling and mining algorithms. First, we minimize the data collected from users by sampling using a privacy-enhanced version of the Metropolis-Hasting Random Walk method. Then, we applied the distributed FP-Growth algorithm under a privacy-aware setting. We evaluated our approach on three large-scale real-world datasets. Results showed that users can mine very high-quality frequent itemsets, especially in well-connected networks, while maintaining the decentralized nature and privacy advantages of their social network. In comparison to other traversal-based sampling mechanisms, our Metropolis-Hasting Anonymous Random Walk approach achieves a better quality of frequent itemsets while also reducing the communication overhead. Our approach showed that it is possible for users to co-create one of the most important components of many recommender systems, frequent itemsets, based on distributed data while maintaining a good level of privacy.

The second analytics technique we studied in this thesis is based on neural networks. More precisely, we elaborated on the emerging technique for training the models in a distributed manner, Federated Learning (FL). FL allows users to train a joint model collaboratively without sharing their data with other parties. Instead, users train the model locally and share only the model updates with a central server, which coordinates the training process. Chapter 5 discussed a number of privacy issues that are exacerbated by the centralized coordination, such as centralization of control and constrained defenses. We emphasized a furthering of the FL approach towards hierarchical architectures (coined Hierarchical Federated Learning (HFL)) that was recently discussed and investigated in a few publications. We contributed to HFL in several ways, particularly by investigating various measures for improved and more targeted privacy protection. HFL allows flexible distribution of the FL functionality across the hierarchy, such that other nodes (users or group servers) in the network can also participate in coordinating the training process besides the central server. Furthermore, the hierarchy facilitates applying known defense and verification methods more effectively and efficiently by placing them in certain parts of the hierarchy when needed. Finally, with HFL, users can leverage their trust in each other to mitigate several threats in different application scenarios. With this contribution, we ultimately contributed to the application of FL in a more distributed fashion that better matches emerging technologies such as fog computing. Also, we showed the implications of the underlay architecture of FL on privacy issues.

Despite the privacy advantages brought by FL, its distributed nature enables a novel set of attacks and threats to user data. Chapter 6 presented a systematic mapping study on recent publications that address attacks in FL. This chapter structured the publications according to multiple classification schemes regard-

ing the attacks' properties and evaluation setups. The study showed that the majority of works focus on the classification function and on neural network models, Convolutional Neural Network models in particular, while overlooking other machine learning algorithms. We identified several issues in the assumptions made in several papers, which lead to the restricted applicability of the proposed attacks in real-world scenarios. For example, some attacks are effective only under specific uncommon values of hyper-parameters. Our study also identified multiple fallacies in the evaluation of the attacks. We argued that these fallacies can affect the validity and generalizability of the results. For instance, the usage of oversimplified models or datasets was found in the majority of the publications. To alleviate these fallacies, we contributed several actionable recommendations, such as using modern models and FL-oriented datasets. Overall, this contribution helps illustrate the big picture of attacks in FL and shed light on the suboptimal evidence we have in the literature on their effectiveness in real-world scenarios.

Our study on attacks identified several gaps in the research field. One of these gaps is the lack of investigations about the risk of leaking the ground-truth labels of users' data. The ground-truth labels are the annotations generated on the user side that specify the correct context of the data w.r.t. the machine learning task. Chapter 7 contributed Label Leakage from Gradients (LLG), a novel attack that lies in this gap. LLG exploits two properties of gradients of the last layer in a neural network model to extract the label occurrences in the training batches. We demonstrated the severity of the attack on a variety of datasets, models, and two FL algorithms. Results showed that LLG is effective for arbitrary batch sizes and numbers of classes. LLG maintained a high success rate on various model architectures and in different stages of training. As mitigations, we tested noisy gradients and gradient compression. Findings indicated that gradient compression can be an effective technique to prevent the attack while preserving the model accuracy. With this contribution, we identified and demonstrated the existence of a vulnerability in FL and that can help researchers and practitioners better assess the risk of FL. We also showed the importance of applying particular defense mechanisms to reduce this risk. Ultimately, this work contributes to the application of FL in a more privacy-enhanced manner.

8.2 OUTLOOK

Several research directions can complement the work in this thesis. We showed in Chapter 2 that HOSN still partially counts on the infrastructure of the underlying COSN to connect users. A satisfactory compensation for this usage should be offered to the service provider of COSN. In this thesis, we discussed offering limited statistics on trends and also elaborated on building data models from distributed data. Yet, further research is needed on how to incorporate these statistics and models effectively in the business model of the service provider.

For improving the trustworthiness of HOSN (Chapter 3), more investigations on how the software features need to be chosen to maximize their impact on

the privacy concerns and trust beliefs. Additional variables can be considered to better understand the users' perceptions of HOSN, such as users' risk awareness.

With regard to user sampling and Association Rule Mining (ARM) (Chapter 4), our analysis of the sample sizes and the quality of the frequent itemsets can be a base for advancement in progressive sampling methods for ARM, e.g., [43]. More variants of random walks can also be incorporated in this research to test their impact on the sample quality, e.g., rejection-controlled Metropolis-Hastings [151].

We discussed in Chapter 5 potential privacy advantages of HFL. Future work can investigate this direction by implementing and evaluate our proposals regarding functionality and defenses distribution across the hierarchy. In addition, it is important to study the impact of this architecture, in particular, the multiple aggregations across the layers, on the model accuracy under different sizes and shapes of the hierarchy. These multiple aggregations also highlight the importance of improvements to the standard aggregation method (weighted average in FedAvg), which can be suboptimal in some real-world scenarios.

On the anti-privacy research side (Chapter 7), further improvements to the LLG attack under the FedAvg algorithm and against trained models can be investigated. Also, extending the attack to other than the classification tasks and other loss functions can be a promising research direction. Another interesting direction is investigating the implications of combining LLG with the DLG attack on the overall accuracy of the data reconstruction. Interestingly, several researchers from different universities have contacted us, expressing their interest in developing the attack and continuing this line of research.

Overall, our thesis contributes a considerable step towards empowering users with more privacy-enhanced OSNs and the ability to take an active role in applying analytics on their data without sacrificing their privacy.

Part VI

APPENDIX

APPENDIX OF CHAPTER 3

A.1 SOFTWARE FEATURES

In this section, we present the software features that were implemented in the Hushtweet mockups. For each privacy concern, three features were implemented as shown in Tables A.1, A.2, A.3, A.4, and A.5, A.6.

A.2 SURVEY QUESTIONNAIRE

The questions were adopted mainly from Malhotra et al. [171] and adjusted for the Hushtweet application. Instead of measuring the privacy concerns, we measured the mitigation of privacy concerns. Thus, the questions were modified to suggest that Hushtweet is actively mitigating the privacy concerns.

Privacy Concerns

Control

1 Privacy is really a matter of the right of Hushtweet users to exercise control and autonomy over decisions about how their information is collected, used, and shared.

2 In Hushtweet, user control of personal information lies at the heart of user privacy.

3 My online privacy is invaded when control is lost or unwillingly reduced as a result of sharing personal information with Hushtweet.

Awareness

1 Hushtweet discloses the way data are collected, processed and used.

2 Hushtweets' privacy policy has a clear and conspicuous disclosure.

3 I am aware and knowledegable about how my personal information is used by Hushtweet.

Collection

1 Hushtweet asks me for personal information.

2 When Hushtweet asks me for personal information, I sometimes think twice before providing it.

3 I would give personal information to Hushtweet.

Goal	Facet	Requirement	Feature
Clarity of privacy practices	Transparency	Providing an overview of the privacy practices	FAQ: "How does Hushtweet protect my privacy?"
Clarity of privacy practices	Transparency	Informing users on the privacy practice for private tweets	Alert messages on tweeting (1) privately: "This tweet will be encrypted", and (2) publicly: "Twitter has access to this data"
Clarity of privacy practices	Transparency, Provider integrity	Informing users on the legally binding commitments of Hushtweet regarding privacy	"Privacy Policy" page that informs users on data collection and its purpose.

Table A.1: Features for lacking awareness†.

Goal	Facet	Requirement	Feature
Fairness	Completeness, Transparency, Provider integrity	Showing users the statistical information that they are part of. Clarifying their gain from the collection of this information.	"My data" page that contains: (1) description of the statistical information and its purpose, and (2) a list of the statistical information that the user is part of.
Awareness	Transparency, Provider integrity, Provider predictability	Informing users on their data usage by Twitter and Hushtweet, and the services they receive in return	FAQ: "How does Hushtweet and Twitter use my data?" FAQ: "What is my benefit from Hushtweet services in comparison to Twitter services?"
Awareness	Transparency, Provider integrity	Informing users on their data usage by Twitter and Hushtweet	Alert messages on tweeting (1) privately: "Twitter can't use contained data for targeted ads", and (2) publicly: "Twitter might use contained data for targeted ads"

Table A.2: Features for collection†.

4 Hushtweet is collecting too much personal information about me.

Errors

1 All the personal information in the distributed databases used by Hushtweet are double-checked for accuracy – no matter how much this costs.

2 Hushtweet makes sure that the personal information in their files is accurate.

3 Hushtweet has procedures to correct errors in personal information.

4 Hushtweet devotes time and effort to verifying the accuracy of the personal information in the distributed databases.

Unauthorized secondary use

1 Hushtweet does not use personal information for any purpose unless it has been authorized by the individuals who provided information.

Goal	Facet	Requirement	Feature
Data control	Privacy (control)	Allowing users to decide how their data is shared	A toggle button to change the user's posted tweet status between private and public at any time.
Data control	Privacy (control)	Allowing users to delete all their data	FAQ: "How can I delete my data?" "My data" page that contains a button for deleting all user data (private tweets and anonymous likes)
Procedure control	Privacy (control)	Allowing users to decide what data is used for statistical information	"My data" page that contains: (1) a list of the statistical information that the user is part of, and (2) a toggle button for each item of this information with which the user can opt-in/-out of collection.

Table A.3: Features for insufficient control†.

Goal	Facet	Requirement	Feature
Data accuracy	Data integrity, Data reliability, Data validity	Verifying the correctness of the data	An alert message on tweeting privately: "Data is correctly and safely stored".
Data accuracy	Data integrity, Data reliability, Datav validity	Verifying the correctness of the data	FAQ: "How does Hushtweet ensure the correctness and integrity of my data?" FAQ: "Does Hushtweet modify my data?"
Data accuracy	Failure tolerance	Maintaining the data accuracy on network disconnection	On tweeting while the network is disconnected, the tweet is stored locally and can be posted when the network is connected again. Informing the user with an alert message: "Don't worry, your tweet is saved locally. Just retry when your network connection is back"

Table A.4: Features for errors†.

2 When people give personal information to Hushtweet for some reason, Hushtweet does not use the information for any other reason.

3 Hushtweet does not sell the personal information in the distributed databases to other companies.

4 Hushtweet does not share personal information with other companies unless it has been authorized by the individuals who provided the information.

Improper access

1 Hushtweet devotes time and effort to preventing unauthorizing access to the personal information.

Goal	Facet	Requirement	Feature
Technical access control	Confidentiality	Protecting user data from unauthorized users or parties	An alert message on tweeting privately: "Data is correctly and safely stored".
Technical access control	Traceability, Transparency	Showing users who had access to their data	"Access History" page that displays time of (1) user login, (2) data access for calculating statistical information by Hushtweet, and (3) profile view by other users
Organizational access control	Provider integrity	Clarifying the Hushtweet policy in regard to the restricted access of developers to user data	FAQ: "How does Hushtweet protect my data from improper access?"

Table A.5: Features for improper access†.

Goal	Facet	Requirement	Feature
Authorization	Transparency	Informing users on their data usage by Hushtweet. Requesting authorization for data access and usage.	"Authorization" page that contains: (1) description of data access and usage by Hushtweet, and (2) authorization button with which users authorize Hushtweet to access their data on Twitter and calculate statistical information.
Clarity of intent	Provider integrity, Provider benevolence	Informing users on the intent of Hushtweet as a research project	"About us" page that informs users on the purpose of the research project of Hushtweet, which is protecting users' privacy, and also listing the involved universities and researchers
Clarity of data use purpose	Transparency, Provider integrity	Informing users on how hustweet uses the statistical information and which parties have access to it	FAQ: "For what purpose is my data used?" FAQ: "Is my data shared with third parties?"

Table A.6: Features for unauthorized secondary use†.

2 Distributed databases that contain personal information are protected from unauthorized access by Hushtweet – no matter what it costs.

3 Hushtweet makes sure that unauthorized people cannot access personal information in the distributed databases.

Global Information Privacy concern

1 All things considered, the Internet would cause serious privacy problems.

2 Compared to others, I am more sensitive about the way online companies handle my personal information.

3 To me, it is the most important thing to keep my privacy intact from online companies.

4 I believe other people are too much concerned with online privacy issues.

5 Compared with other subjects on my mind, personal privacy is very important.

6 I am concerned about threats to my personal privacy today.

Trust beliefs

1 Hushtweet is trustworthy in handling personal information.

2 Hushtweet tells the truth and fullfill promises related to personal information provided by me.

3 I trust that Hushtweet keeps my best interests in mind when dealing with personal information.

4 Hushtweet is in general predictable and consistent regarding the usage of personal information.

5 Hushtweet is always honest with customers when it comes to using personal information that I provide.

Risk beliefs

1 In general, it is risky to give personal information to HushTweet.

2 There is high potential for loss associated with giving personal information to HushTweet.

3 There is too much uncertainty associated with giving personal information to HushTweet.

4 Providing HushTweet with personal information involves many unexpected problems.

5 I feel safe giving personal information to HushTweet.

Willingness to use the app

1 I am interested in using Hushtweet.

2 I am willing to use Hushtweets' private tweeting functionality.

3 I would rather use Hushtweets' anonymous like than liking publicly on Twitter.

4 I prefer Hushtweet over Twitter.

5 I would download Hushtweet.

6 I am willing to use Hushtweet anonymous liking functionality.

7 I would tell my friends about Hushtweet.

8 I would rather use Hushtweets' private tweet than tweeting publicly on Twitter.

B

APPENDIX OF CHAPTER 4

As mentioned in Section 4.4.1, a user u_k sends $\frac{|\mathcal{F}(u_k)|}{p}$ to a candidate u_{k+1}. Having an estimation error of β in p, i.e., $p' = p \pm \beta p$, leads to the same error in $|\mathcal{F}(u_k)|$, i.e., $|\mathcal{F}(u_k)|' = |\mathcal{F}(u_k)| \pm \beta|\mathcal{F}(u_k)|$. In case of adding noise to $|\mathcal{F}(u_k)|$, the estimation error will be a combination of the error comes from the noise α and the error comes from the random variable β. In the following, we calculate the final error based on α and β. For simplicity, we denote $x = |\mathcal{F}(u_k)|$ and $y = p$.

$$
\begin{aligned}
z &= x''/y' \\
z &= x''/(y + \beta y) \\
z &= x' + (\beta x')/(y + \beta y) \\
x' &= x + \alpha x \\
z &= \frac{(x + \alpha x) + \beta(x + \alpha x)}{y + \beta y} \\
x'' &= x + (\alpha + \beta + \alpha\beta)x
\end{aligned}
$$

Thus, $\alpha + \beta + \alpha\beta$ is the final estimation error.

C

APPENDIX OF CHAPTER 7

The model architectures used in the experiments are as shown in Tables C.1 and C.2.

Layer	Size	Activation function
(input)	-	-
Conv 2D	channels x 12	Sigmoid
Conv 2D	12 x 12	Sigmoid
Conv 2D	12 x 12	Sigmoid

Table C.1: Architecture of CNN, the default model in the experimental setting†.

Layer	Size	Activation function
(input)	-	-
Conv 2D	1 x 6	ReLU
Maxpool	2 x 2	-
Conv 2D	6 x 16	ReLU
Maxpool	2	-
Linear	16 x 6	ReLU
Linear	120 x 84	ReLU
Linear	84 x 10	ReLU

Table C.2: Architecture of LeNet network, a very common architecture adopted in computer vision†.

BIBLIOGRAPHY

1. Abad, G. L. & Orón, L. C. How social networks and data brokers trade with private data. *Redes. com: revista de estudios para el desarrollo social de la comunicación,* 84–103 (2016).
2. Abadi, M., McMahan, H. B., Chu, A., Mironov, I., Zhang, L., Goodfellow, I. & Talwar, K. *Deep learning with differential privacy* in *Proceedings of the ACM Conference on Computer and Communications Security* (2016).
3. Agrawal, R., Srikant, R., *et al. Fast algorithms for mining association rules* in *Proc. 20th int. conf. very large data bases, VLDB* **1215** (1994), 487–499.
4. Aiken, L. S., West, S. G. & Reno, R. R. *Multiple regression: Testing and interpreting interactions* (sage, 1991).
5. Aledhari, M., Razzak, R., Parizi, R. M. & Saeed, F. Federated learning: A survey on enabling technologies, protocols, and applications. *IEEE Access* **8,** 140699–140725 (2020).
6. Andrew, G., Thakkar, O., McMahan, H. B. & Ramaswamy, S. Differentially private learning with adaptive clipping. *arXiv preprint arXiv:1905.03871* (2019).
7. Anguita, D., Ghio, A., Oneto, L., Parra, X. & Reyes-Ortiz, J. L. *A public domain dataset for human activity recognition using smartphones.* in *Esann* **3** (2013), 3.
8. Aono, Y., Hayashi, T., Wang, L., Moriai, S., *et al.* Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security* **13,** 1333–1345 (2017).
9. Aono, Y., Hayashi, T., Wang, L., Moriai, S., *et al. Privacy-preserving deep learning: Revisited and enhanced* in *International Conference on Applications and Techniques in Information Security* (2017), 100–110.
10. Augenstein, S., McMahan, H. B., Ramage, D., Ramaswamy, S., Kairouz, P., Chen, M., Mathews, R., *et al.* Generative models for effective ML on private, decentralized datasets. *arXiv preprint arXiv:1911.06679* (2019).
11. Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D. & Shmatikov, V. How to backdoor federated learning. *arXiv preprint arXiv:1807.00459* (2018).
12. Balaban, S. Deep learning and face recognition: the state of the art. *CoRR* **abs/1902.03524** (2019).
13. Balle, B., Kairouz, P., McMahan, H. B., Thakkar, O. & Thakurta, A. Privacy Amplification via Random Check-Ins. *arXiv preprint arXiv:2007.06605* (2020).

14. Bar-Yossef, Z., Berg, A., Chien, S., Fakcharoenphol, J. & Weitz, D. *Approximating aggregate queries about web pages via random walks* in *VLDB* (2000), 535–544.

15. Baruch, M., Baruch, G. & Goldberg, Y. A little is enough: Circumventing defenses for distributed learning. *arXiv preprint arXiv:1902.06156* (2019).

16. Benesty, J., Chen, J., Huang, Y. & Cohen, I. in *Noise reduction in speech processing* 1–4 (Springer, 2009).

17. Benet, J. Ipfs-content addressed, versioned, p2p file system. *arXiv preprint arXiv:1407.3561* (2014).

18. Bhagoji, A. N., Chakraborty, S., Mittal, P. & Calo, S. *Model poisoning attacks in federated learning* in *Proc. Workshop Secur. Mach. Learn.(SecML) 32nd Conf. Neural Inf. Process. Syst.(NeurIPS)* (2018).

19. Bhagoji, A. N., Chakraborty, S., Mittal, P. & Calo, S. *Analyzing federated learning through an adversarial lens* in *36th International Conference on Machine Learning, ICML 2019* (2019).

20. Bittau, A., Erlingsson, U., Maniatis, P., Mironov, I., Raghunathan, A., Lie, D., Rudominer, M., Kode, U., Tinnes, J. & Seefeld, B. *Prochlo: Strong privacy for analytics in the crowd* in *Proceedings of the 26th Symposium on Operating Systems Principles* (2017), 441–459.

21. Boer, D. & Kramer, S. Secure Sum Outperforms Homomorphic Encryption in (Current) Collaborative Deep Learning. *arXiv preprint arXiv:2006.02894* (2020).

22. Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A. & Seth, K. Practical secure aggregation for federated learning on user-held data. *arXiv preprint arXiv:1611.04482* (2016).

23. Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A. & Seth, K. *Practical secure aggregation for privacy-preserving machine learning* in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (2017), 1175–1191.

24. Borchert, A., Diaz Ferreyra, N. E. & Heisel, M. *Building trustworthiness in computer-mediated introduction: A facet-oriented framework* in *Int. Conf. on Social Media and Society* (2020), 39–46.

25. Borchert, A., Ferreyra, N. E. D. & Heisel, M. *A Conceptual Method for Eliciting Trust-related Software Features for Computer-mediated Introduction.* in *ENASE* (2020), 269–280.

26. Borchert, A., Wainakh, A., Krämer, N., Mühlhäuser, M. & Heisel, M. *Mitigating Privacy Concerns by Developing Trust-related Software Features for a Hybrid Social Media Application* in *ENASE* (2021), 269–280.

27. Borchert, A., Wainakh, A., Krämer, N., Mühlhäuser, M. & Heisel, M. *The Relevance of Privacy Concerns, Trust, and Risk for Hybrid Social Media* in *Communications in Computer and Information Science (CCIS)* (2022).

28. Buchegger, S., Schiöberg, D., Vu, L.-H. & Datta, A. *PeerSoN: P2P social networking: early experiences and insights* in *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems* (2009), 46–52.

29. Buczak, A. L. & Guven, E. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications surveys & tutorials* **18,** 1153–1176 (2015).

30. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P. & Maxwell, G. *Bulletproofs: Short proofs for confidential transactions and more* in *2018 IEEE Symposium on Security and Privacy (SP)* (2018), 315–334.

31. Burhhalter, L. *Robust Secure Aggregation for Privacy-Preserving Federated Learning with Adversaries* in *2019 cheeeeck* (2019).

32. Caldas, S., Wu, P., Li, T., Konecny, J., McMahan, H. B., Smith, V. & Talwalkar, A. LEAF: A Benchmark for Federated Settings. *CoRR* **abs/1812.01097** (2018).

33. Cao, D., Chang, S., Lin, Z., Liu, G. & Sun, D. *Understanding distributed poisoning attack in federated learning* in *2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)* (2019), 233–239.

34. Cao, Q., Shen, L., Xie, W., Parkhi, O. M. & Zisserman, A. *Vggface2: A dataset for recognising faces across pose and age* in *2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018)* (2018), 67–74.

35. Catanese, S. A., De Meo, P., Ferrara, E., Fiumara, G. & Provetti, A. *Crawling facebook for social network analysis purposes* in *Proceedings of the international conference on web intelligence, mining and semantics* (2011), 1–8.

36. Chahar, H., Keshavamurthy, B. N. & Modi, C. Privacy-preserving distributed mining of association rules using Elliptic-curve cryptosystem and Shamir's secret sharing scheme. *Sadhana - Academy Proceedings in Engineering Sciences* **42,** 1997–2007 (2017).

37. Chakaravarthy, V. T., Pandit, V. & Sabharwal, Y. *Analysis of sampling techniques for association rule mining* in *Proceedings of the 12th international conference on database theory* (2009), 276–283.

38. Chen, C.-L., Golubchik, L. & Paolieri, M. Backdoor attacks on federated meta-learning. *arXiv preprint arXiv:2006.07026* (2020).

39. Chen, J., Zhang, J., Zhao, Y., Han, H., Zhu, K. & Chen, B. *Beyond Model-Level Membership Privacy Leakage: an Adversarial Approach in Federated Learning* in *2020 29th International Conference on Computer Communications and Networks (ICCCN)* (2020), 1–9.

40. Chen, L., Wang, H., Charles, Z. & Papailiopoulos, D. Draco: Byzantine-resilient distributed training via redundant gradients. *arXiv preprint arXiv:1803.09877* (2018).

41. Cheng, J. Brain tumor dataset. *Distributed by Figshare* (2017).

42. Choi, J. P. Herd behavior, the" penguin effect," and the suppression of informational diffusion: an analysis of informational externalities and payoff interdependency. *The Rand Journal of Economics,* 407–425 (1997).

43. Chuang, K.-T., Chen, M.-S. & Yang, W.-C. *Progressive sampling for association rules based on sampling error estimation* in *Pacific-Asia conference on knowledge discovery and data mining* (2005), 505–515.

44. Codella, N., Rotemberg, V., Tschandl, P., Celebi, M. E., Dusza, S., Gutman, D., Helba, B., Kalloo, A., Liopyris, K., Marchetti, M., *et al.* Skin lesion analysis toward melanoma detection 2018: A challenge hosted by the international skin imaging collaboration (isic). *arXiv preprint arXiv:1902.03368* (2019).

45. Cohen, G., Afshar, S., Tapson, J. & Van Schaik, A. *EMNIST: Extending MNIST to handwritten letters* in *2017 International Joint Conference on Neural Networks (IJCNN)* (2017), 2921–2926.

46. Cramér, H. *Mathematical methods of statistics* (Princeton university press, 1999).

47. Cutillo, L. A., Molva, R. & Strufe, T. Safebook: A privacy-preserving online social network leveraging on real-life trust. *IEEE Communications Magazine* **47,** 94–101 (2009).

48. Dacrema, M. F., Cremonesi, P. & Jannach, D. *Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches* in *Proceedings of the 13th ACM Conference on Recommender Systems* (Association for Computing Machinery, Copenhagen, Denmark, 2019), 101–109.

49. Daley, M. F., Goddard, K., McClung, M., Davidson, A., Weiss, G., Palen, T., Nyirenda, C., Platt, R., Courtney, B. & Reichman, M. E. Using a handheld device for patient data collection: a pilot for medical countermeasures surveillance. *Public Health Reports* **131,** 30–34 (2016).

50. Data, D., Song, L. & Diggavi, S. Data encoding for Byzantine-resilient distributed optimization. *arXiv preprint arXiv:1907.02664* (2019).

51. Daubert, J., Bock, L., Kikirasy, P., Mühlhauser, M. & Fischer, M. *Twitterize: Anonymous micro-blogging* in *2014 IEEE/ACS 11th International Conference on Computer Systems and Applications (AICCSA)* (2014), 817–823.

52. Daubert, J., Fischer, M., Grube, T., Schiffner, S., Kikiras, P. & Mühlhäuser, M. AnonPubSub: Anonymous publish-subscribe overlays. *Computer Communications* **76,** 42–53 (2016).

53. Davenport, C. *Gboard passes one billion installs on the Play Store* https://www.androidpolice.com/2018/08/22/gboard-passes-one-billion-installs-play-store/. Online; accessed 23.06.2021. 2018.

54. De Cock, M., Dowsley, R., Nascimento, A. C., Railsback, D., Shen, J. & Todoki, A. High performance logistic regression for privacy-preserving genome analysis. *BMC Medical Genomics* **14,** 1–18 (2021).

55. De Cristofaro, E. An Overview of Privacy in Machine Learning (2020).

56. Dean, B. *Social Network Usage & Growth Statistics: How Many People Use Social Media in 2021?* https://backlinko.com/social-media-users. Online; accessed 24.06.2021. 2021.

57. Delis, A., Verykios, V. S. & Tsitsonis, A. A. *A data perturbation approach to sensitive classification rule hiding* in *Proceedings of the 2010 ACM Symposium on Applied Computing* (2010), 605–609.

58. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. & Fei-Fei, L. *Imagenet: A large-scale hierarchical image database* in *2009 IEEE conference on computer vision and pattern recognition* (2009), 248–255.

59. Diaz, C. Anonymity and privacy in electronic services. *Heverlee: Katholieke Universiteit Leuven. Faculteit Ingenieurswetenschappen* (2005).

60. Dictionary, O. *Oxford definition of analytics* https://www.lexico.com/en/definition/analytics. Online; accessed 16.10.2021.

61. Donath, J. in *CHI'14 Extended Abstracts on Human Factors in Computing Systems* 1057–1058 (2014).

62. Dowling, G. R. Perceived risk: the concept and its measurement. *Psychology & Marketing* **3,** 193–210 (1986).

63. Dowling, G. R. & Staelin, R. A model of perceived risk and intended risk-handling activity. *Journal of consumer research* **21,** 119–134 (1994).

64. Dua, D. & Graff, C. *UCI Machine Learning Repository* 2017. http://archive.ics.uci.edu/ml.

65. Duane, S., Tandan, M., Murphy, A. W. & Vellinga, A. Using mobile phones to collect patient data: lessons learned from the SIMPle study. *JMIR research protocols* **6,** e61 (2017).

66. Dwyer, C., Hiltz, S. & Passerini, K. Trust and privacy concern within social networking sites: A comparison of Facebook and MySpace. *AMCIS 2007 proceedings,* 339 (2007).

67. Ebert, N., Ackermann, K. A. & Heinrich, P. *Does Context in Privacy Communication Really Matter?—A Survey on Consumer Concerns and Preferences* in *Proc. of the 2020 CHI Conf. on Human Factors in Computing Systems* (2020), 1–11.

68. Elliott, A. C. & Woodward, W. A. *Statistical analysis quick reference guidebook: With SPSS examples* (Sage, 2007).

69. Enthoven, D. & Al-Ars, Z. An Overview of Federated Deep Learning Privacy Attacks and Defensive Strategies. *arXiv preprint arXiv:2004.04676* (2020).

70. Enthoven, D. & Al-Ars, Z. Fidel: Reconstructing Private Training Samples from Weight Updates in Federated Learning. *arXiv preprint arXiv:2101.00159* (2021).

71. Evfimievski, A., Srikant, R., Agrawal, R. & Gehrke, J. Privacy preserving mining of association rules. *Information Systems* **29,** 343–364 (2004).

72. Evtimov, I., Cui, W., Kamar, E., Kiciman, E., Kohno, T. & Li, J. Security and Machine Learning in the Real World. *arXiv preprint arXiv:2007.07205* (2020).

73. Facebook. *Facebook Q4 2018 Results* https://s21.q4cdn.com/399680738/files/doc_financials/2018/Q4/Q4-2018-Earnings-Presentation.pdf. Online, accessed 21.03.2019. 2019.

74. Fang, M., Cao, X., Jia, J. & Gong, N. Z. Local model poisoning attacks to Byzantine-robust federated learning. *arXiv preprint arXiv:1911.11815* (2019).

75. Farke, F. M., Balash, D. G., Golla, M., Dürmuth, M. & Aviv, A. J. *Are Privacy Dashboards Good for End Users? Evaluating User Perceptions and Reactions to Google's My Activity* in *30th USENIX Security Symposium (USENIX Security 21)* (2021), 483–500.

76. Fayyad, U., Piatetsky-Shapiro, G. & Smyth, P. The KDD process for extracting useful knowledge from volumes of data. *Communications of the ACM* **39,** 27–34 (1996).

77. Fernandes, K., Vinagre, P. & Cortez, P. *A proactive intelligent decision support system for predicting the popularity of online news* in *Portuguese Conference on Artificial Intelligence* (2015), 535–546.

78. Finkle, J. *Twitter urges all users to change passwords after glitch* https://www.reuters.com/article/us-twitter-passwords/twitter-urges-all-users-to-change-passwords-after-glitch-idUSKBN1I42JG. Online; accessed 16.10.2021.

79. Flores, M., Dayan, I., Roth, H., Zhong, A., Harouni, A., Gentili, A., Abidin, A., Liu, A., Costa, A., Wood, B., *et al.* Federated Learning used for predicting outcomes in SARS-COV-2 patients. *Research Square* (2021).

80. Fung, C., Yoon, C. J. & Beschastnikh, I. Mitigating sybils in federated learning poisoning. *arXiv preprint arXiv:1808.04866* (2018).

81. Ganju, K., Wang, Q., Yang, W., Gunter, C. A. & Borisov, N. *Property inference attacks on fully connected neural networks using permutation invariant*

representations in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security* (2018), 619–633.

82. Gao, C. & Elzarka, H. The use of decision tree based predictive models for improving the culvert inspection process. *Advanced Engineering Informatics* **47,** 101203 (2021).

83. Garg, V., Singh, A. & Singh, D. A survey of association rule hiding algorithms. *Proceedings - 2014 4th International Conference on Communication Systems and Network Technologies, CSNT 2014,* 404–407 (2014).

84. Gedeon, J., Heuschkel, J., Wang, L. & Mühlhäuser, M. Fog computing: Current research and future challenges. *KuVS-Fachgespräch Fog Comput* **1,** 1–4 (2018).

85. Geiping, J., Bauermeister, H., Dröge, H. & Moeller, M. *Inverting Gradients - How easy is it to break privacy in federated learning?* in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual* (eds Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.-F. & Lin, H.-T.) (2020).

86. Geyer, R. C., Klein, T. & Nabi, M. Differentially Private Federated Learning: A Client Level Perspective. *ArXiv e-prints* (2017).

87. Ghosh, S. & Reilly, D. L. *Credit card fraud detection with a neural-network* in *System Sciences, 1994. Proceedings of the Twenty-Seventh Hawaii International Conference on* **3** (1994), 621–630.

88. Gjoka, M., Kurant, M., Butts, C. T. & Markopoulou, A. *Walking in facebook: A case study of unbiased sampling of osns* in *2010 Proceedings IEEE Infocom* (2010), 1–9.

89. Gjoka, M., Kurant, M., Butts, C. T. & Markopoulou, A. Practical recommendations on crawling online social networks. *IEEE Journal on Selected Areas in Communications* **29,** 1872–1892 (2011).

90. Goldfarb, A. & Tucker, C. Shifts in privacy concerns. *American Economic Review* **102,** 349–53 (2012).

91. Goodfellow, I., Bengio, Y. & Courville, A. *Deep Learning* `http://www.deeplearningbook.org` (MIT Press, 2016).

92. Goodfellow, I. J., Erhan, D., Carrier, P. L., Courville, A., Mirza, M., Hamner, B., Cukierski, W., Tang, Y., Thaler, D., Lee, D.-H., *et al. Challenges in representation learning: A report on three machine learning contests* in *International conference on neural information processing* (2013), 117–124.

93. Graffi, K., Podrajanski, S., Mukherjee, P., Kovacevic, A. & Steinmetz, R. *A distributed platform for multimedia communities* in *2008 Tenth IEEE International Symposium on Multimedia* (2008), 208–213.

94. Grazioli, S. & Jarvenpaa, S. L. Perils of Internet fraud: An empirical investigation of deception and trust with experienced Internet consumers.

IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans **30,** 395–410 (2000).

95. Greff, K., Srivastava, R. K., Koutnik, J., Steunebrink, B. R. & Schmidhuber, J. LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems* **28,** 2222–2232 (2016).

96. Grewal, P. *Suspending Cambridge Analytica and SCL Group From Facebook* https://newsroom.fb.com/news/2018/03/suspending-cambridge-analytica/. Online, accessed 21.03.2019. 2018.

97. Guardian, T. *Facebook to contact 87 million users affected by data breach* https://www.theguardian.com/technology/2018/apr/08/facebook-to-contact-the-87-million-users-affected-by-data-breach. Online; accessed 09.05.2019. 2018.

98. Guardian, T. *Huge Facebook breach leaves thousands of other apps vulnerable* https://www.theguardian.com/technology/2018/oct/02/facebook-hack-compromised-accounts-tokens. Online; accessed 09.05.2019. 2018.

99. Gürses, S. & Diaz, C. Two tales of privacy in online social networks. *IEEE Security & Privacy* **11,** 29–37 (2013).

100. Hamilton, W. L., Ying, R. & Leskovec, J. Inductive representation learning on large graphs. *arXiv preprint arXiv:1706.02216* (2017).

101. Hamner, B. *Popular Datasets Over Time* https://www.kaggle.com/benhamner/popular-datasets-over-time/code. Online; accessed 31.05.2021. 2017.

102. Han, J., Pei, J. & Yin, Y. Mining frequent patterns without candidate generation. *ACM sigmod record* **29,** 1–12 (2000).

103. Hao, M., Li, H., Luo, X., Xu, G., Yang, H. & Liu, S. Efficient and privacy-enhanced federated learning for industrial artificial intelligence. *IEEE Transactions on Industrial Informatics* **16,** 6532–6542 (2019).

104. Hard, A., Rao, K., Mathews, R., Ramaswamy, S., Beaufays, F., Augenstein, S., Eichner, H., Kiddon, C. & Ramage, D. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604* (2018).

105. Hargreaves, T. *Is it Time to Ditch the MNIST Dataset?* https://www.ttested.com/ditch-mnist/. Online; accessed 01.06.2021. 2020.

106. Hayes, A. F. *Introduction to mediation, moderation, and conditional process analysis: A regression-based approach* (Guilford publications, 2017).

107. Hayes, J., Melis, L., Danezis, G. & De Cristofaro, E. LOGAN: Membership Inference Attacks Against Generative Models. *Proceedings on Privacy Enhancing Technologies* (2019).

108. He, C., Li, S., So, J., Zhang, M., Wang, H., Wang, X., Vepakomma, P., Singh, A., Qiu, H., Shen, L., *et al.* FedML: A Research Library and Bench-

mark for Federated Machine Learning. *arXiv preprint arXiv:2007.13518* (2020).

109. He, K., Zhang, X., Ren, S. & Sun, J. *Deep residual learning for image recognition* in *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), 770–778.

110. Henningsen, S., Florian, M., Rust, S. & Scheuermann, B. *Mapping the interplanetary filesystem* in *2020 IFIP Networking Conference (Networking)* (2020), 289–297.

111. Herzberg, A. & Leibowitz, H. *Can Johnny finally encrypt?: evaluating E2E-encryption in popular IM applications* in *ACM Workshop on Socio-Technical Aspects in Security and Trust (STAST)* (2016).

112. Hettich, S. Kdd cup 1999 data. *The UCI KDD Archive* (1999).

113. Hipp, J., Güntzer, U. & Nakhaeizadeh, G. Algorithms for association rule mining—a general survey and comparison. *ACM sigkdd explorations newsletter* **2,** 58–64 (2000).

114. Hitaj, B., Ateniese, G. & Perez-Cruz, F. *Deep Models under the GAN: Information leakage from collaborative deep learning* in *Proceedings of the ACM Conference on Computer and Communications Security* (2017).

115. Hitaj, B., Ateniese, G. & Perez-Cruz, F. *Deep models under the GAN: information leakage from collaborative deep learning* in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (2017), 603–618.

116. Holmes, A. *533 million Facebook users' phone numbers and personal data have been leaked online* https://www.businessinsider.com/stolen-data-of-533-million-facebook-users-leaked-online-2021-4?r=US&IR=T. Online; accessed 24.06.2021. 2021.

117. Hootsuite. *General Twitter stats* https://blog.hootsuite.com/twitter-statistics/. Online; accessed 09.05.2019.

118. Hosseinalipour, S., Azam, S. S., Brinton, C. G., Michelusi, N., Aggarwal, V., Love, D. J. & Dai, H. Multi-stage hybrid federated learning over large-scale wireless fog networks. *arXiv preprint arXiv:2007.09511* (2020).

119. Howard, P. & Feyman, Y. *Yelp for Health: Using the Wisdom of Crowds To Find High-Quality Hospitals* 2017.

120. Hu, P. & Lau, W. C. A survey and taxonomy of graph sampling. *arXiv preprint arXiv:1308.5865* (2013).

121. Hu, L.-t. & Bentler, P. M. Cutoff criteria for fit indexes in covariance structure analysis: Conventional criteria versus new alternatives. *Structural equation modeling: a multidisciplinary journal* **6,** 1–55 (1999).

122. Huang, G., Liu, Z., Van Der Maaten, L. & Weinberger, K. Q. *Densely connected convolutional networks* in *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), 4700–4708.

123. Inc., Y. *Yelp Dataset* 2011. https://www.yelp.com/dataset.

124. Jaques, E. In praise of hierarchy. *Markets, hierarchies and networks: The coordination of social life*, 48–52 (1991).

125. Jarvenpaa, S. L., Tractinsky, N. & Saarinen, L. Consumer trust in an Internet store: A cross-cultural validation. *Journal of Computer-Mediated Communication* **5,** JCMC526 (1999).

126. Jayaraman, B., Wang, L., Evans, D. & Gu, Q. Revisiting membership inference under realistic assumptions. *arXiv preprint arXiv:2005.10881* (2020).

127. Jere, M. S., Farnan, T. & Koushanfar, F. A Taxonomy of Attacks on Federated Learning. *IEEE Security Privacy*, 0–0 (2020).

128. Jiang, W., Wang, G. & Wu, J. Generating trusted graphs for trust evaluation in online social networks. *Future generation computer systems* **31,** 48–58 (2014).

129. Jiang, Z., Heng, C. S. & Choi, B. C. Research note—privacy concerns and privacy-protective behavior in synchronous online social interactions. *Information Systems Research* **24,** 579–595 (2013).

130. Jochems, A., Deist, T. M., El Naqa, I., Kessler, M., Mayo, C., Reeves, J., Jolly, S., Matuszak, M., Ten Haken, R., van Soest, J., *et al.* Developing and validating a survival prediction model for NSCLC patients through distributed learning across 3 countries. *International Journal of Radiation Oncology* Biology* Physics* **99,** 344–352 (2017).

131. Jose, P. E. *Doing statistical mediation and moderation* (Guilford Press, 2013).

132. Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., *et al.* Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977* (2019).

133. Kantarcioglu, M. & Clifton, C. Privacy-preserving distributed mining of association rules on horizontally partitioned data. *IEEE Transactions on Knowledge and Data Engineering* **16,** 1026–1037 (2004).

134. Kasiviswanathan, S. P., Lee, H. K., Nissim, K., Raskhodnikova, S. & Smith, A. What can we learn privately? *SIAM Journal on Computing* **40,** 793–826 (2011).

135. Kim, Y., Park, W., Roh, M.-C. & Shin, J. *Groupface: Learning latent groups and constructing group-based representations for face recognition* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020), 5621–5630.

136. Kitchenham, B. & Charters, S. Guidelines for performing systematic literature reviews in software engineering (2007).

137. Konečny, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T. & Bacon, D. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492* (2016).

138. Kotsiantis, S. & Kanellopoulos, D. Association Rules Mining: A Recent Overview. *Greece - Science* **32,** 71–82 (2006).

139. Kozlowska, I. Facebook and data privacy in the age of Cambridge Analytica. *Seattle, WA: The University of Washington. Retrieved August* **1,** 2019 (2018).

140. Krizhevsky, A., Hinton, G., *et al.* Learning multiple layers of features from tiny images. *MIT* (2009).

141. Kumar, N., Berg, A. C., Belhumeur, P. N. & Nayar, S. K. *Attribute and simile classifiers for face verification* in *2009 IEEE 12th international conference on computer vision* (2009), 365–372.

142. Lake, B. M., Salakhutdinov, R. & Tenenbaum, J. B. Human-level concept learning through probabilistic program induction. *Science* **350,** 1332–1338 (2015).

143. Larson, S. *Every single Yahoo account was hacked - 3 billion in all* `http://money.cnn.com/2017/10/03/technology/business/yahoo-breach-3-billion-accounts/index.html`. Online; accessed 09.05.2019. 2017.

144. Larson, S. *Fitness app that revealed military bases highlights bigger privacy issues* `http://money.cnn.com/2018/01/29/technology/strava-privacy-data-exposed/index.html`. Online; accessed 09.05.2019. 2018.

145. LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* **86,** 2278–2324 (1998).

146. Leskovec, J. & Faloutsos, C. *Sampling from large graphs* in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining* (2006), 631–636.

147. Li, H., Wang, Y., Zhang, D., Zhang, M. & Chang, E. Y. *Pfp: parallel fp-growth for query recommendation* in *Proceedings of the 2008 ACM conference on Recommender systems* (2008), 107–114.

148. Li, O., Sun, J., Gao, W., Zhang, H., Yang, X., Xie, J. & Wang, C. Label Leakage and Protection in Two-party Split Learning. *NeurIPS 2020 Workshop on Scalability, Privacy, and Security in Federated Learning (SpicyFL)* (2020).

149. Li, Q., Wen, Z. & He, B. Federated learning systems: Vision, hype and reality for data privacy and protection. *arXiv preprint arXiv:1907.09693* (2019).

150. Li, Q., Wen, Z., Wu, Z., Hu, S., Wang, N. & He, B. A Survey on Federated Learning Systems: Vision, Hype and Reality for Data Privacy and Protection.

151. Li, R.-H., Yu, J. X., Qin, L., Mao, R. & Jin, T. *On random walk based graph sampling* in *2015 IEEE 31st international conference on data engineering* (2015), 927–938.

152. Li, S., Yi, D., Lei, Z. & Liao, S. *The casia nir-vis 2.0 face database* in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops* (2013), 348–353.

153. Lin, T., Stich, S. U., Patel, K. K. & Jaggi, M. Don't Use Large Mini-Batches, Use Local SGD. *arXiv preprint arXiv:1808.07217* (2018).

154. Lin, Y., Han, S., Mao, H., Wang, Y. & Dally, W. J. Deep gradient compression: Reducing the communication bandwidth for distributed training. *arXiv preprint arXiv:1712.01887* (2017).

155. Little, M., McSharry, P., Roberts, S., Costello, D. & Moroz, I. Exploiting nonlinear recurrence and fractal scaling properties for voice disorder detection. *Nature Precedings,* 1–1 (2007).

156. Liu, D. C. & Nocedal, J. On the limited memory BFGS method for large scale optimization. *Mathematical programming* **45,** 503–528 (1989).

157. Liu, K., Dolan-Gavitt, B. & Garg, S. *Fine-pruning: Defending against backdooring attacks on deep neural networks* in *International Symposium on Research in Attacks, Intrusions, and Defenses* (2018), 273–294.

158. Liu, K. S., Xiao, C., Li, B. & Gao, J. *Performing co-membership attacks against deep generative models* in *Proceedings - IEEE International Conference on Data Mining, ICDM* (2019).

159. Liu, L., Zhang, J., Song, S. & Letaief, K. *Client-edge-cloud hierarchical federated learning* in *Proc. IEEE Int. Conf. Commun.(ICC), to be published* (2019).

160. Liu, L., Zhang, J., Song, S. & Letaief, K. B. Edge-assisted hierarchical federated learning with non-iid data. *arXiv preprint arXiv:1905.06641* (2019).

161. Liu, X., Xie, L., Wang, Y., Zou, J., Xiong, J., Ying, Z. & Vasilakos, A. V. Privacy and Security Issues in Deep Learning: A Survey. *IEEE Access* (2020).

162. Liu, Z., Luo, P., Wang, X. & Tang, X. *Deep learning face attributes in the wild* in *Proceedings of the IEEE international conference on computer vision* (2015), 3730–3738.

163. Long, Y., Bindschaedler, V. & Gunter, C. A. Towards measuring membership privacy. *arXiv preprint arXiv:1712.09136* (2017).

164. Lu, H., Liu, C., He, T., Wang, S. & Chan, K. S. Sharing Models or Coresets: A Study based on Membership Inference Attack. *arXiv preprint arXiv:2007.02977* (2020).

165. Luo, J., Wu, X., Luo, Y., Huang, A., Huang, Y., Liu, Y. & Yang, Q. Real-world image datasets for federated learning. *arXiv preprint arXiv:1910.11089* (2019).

166. Luo, W., Xie, Q. & Hengartner, U. *Facecloak: An architecture for user privacy on social networking sites* in *Computational Science and Engineering, 2009. CSE'09. International Conference on* **3** (2009), 26–33.

167. Luo, W., Xie, Q. & Hengartner, U. *FaceCloak Download* https://crysp.uwaterloo.ca/software/facecloak/download.html. Online, accessed 22.03.2019. 2011.

168. Luo, X., Wu, Y., Xiao, X. & Ooi, B. C. Feature Inference Attack on Model Predictions in Vertical Federated Learning. *arXiv preprint arXiv:2010.10152* (2020).

169. Lyu, L., Yu, H. & Yang, Q. Threats to Federated Learning: A Survey. *arXiv preprint arXiv:2003.02133* (2020).

170. Mahloujifar, S., Mahmoody, M. & Mohammed, A. *Universal multi-party poisoning attacks* in *36th International Conference on Machine Learning, ICML 2019* (2019).

171. Malhotra, N. K., Kim, S. S. & Agarwal, J. Internet users' information privacy concerns (IUIPC): The construct, the scale, and a causal model. *Information systems research* **15,** 336–355 (2004).

172. Mao, Y., Zhu, X., Zheng, W., Yuan, D. & Ma, J. *A Novel client Membership Leakage Attack in Collaborative Deep Learning* in *2019 11th International Conference on Wireless Communications and Signal Processing (WCSP)* (2019), 1–6.

173. Mayer, R. C., Davis, J. H. & Schoorman, F. D. An integrative model of organizational trust. *Academy of management review* **20,** 709–734 (1995).

174. McCandless, D. *World's Biggest Data Breaches & Hacks* https://informationisbeautiful.net/visualizations/worlds-biggest-data-breaches-hacks. Online; accessed 18.06.2021. 2021.

175. McMahan, H. B., Moore, E., Ramage, D., Hampson, S., *et al.* Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629* (2016).

176. McMahan, H. B., Ramage, D., Talwar, K. & Zhang, L. Learning differentially private language models without losing accuracy. *arXiv preprint arXiv:1710.06963* (2017).

177. Melis, L., Song, C., De Cristofaro, E. & Shmatikov, V. *Exploiting unintended feature leakage in collaborative learning* in *Proceedings - IEEE Symposium on Security and Privacy* (2019).

178. Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. & Teller, E. Equation of state calculations by fast computing machines. *The journal of chemical physics* **21,** 1087–1092 (1953).

179. Mirshghallah, F., Taram, M., Vepakomma, P., Singh, A., Raskar, R. & Esmaeilzadeh, H. Privacy in deep learning: A survey. *arXiv preprint arXiv:2004.12254* (2020).

180. Mislove, A., Marcon, M., Gummadi, K. P., Druschel, P. & Bhattacharjee, B. *Measurement and analysis of online social networks* in *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement* (2007), 29–42.

181. Mitchell, T. Machine learning (1997).

182. Mo, F., Borovykh, A., Malekzadeh, M., Haddadi, H. & Demetriou, S. Layer-wise Characterization of Latent Information Leakage in Federated Learning. *arXiv preprint arXiv:2010.08762* (2020).

183. Mohammadi, N. G., Paulus, S., Bishr, M., Metzger, A., Koennecke, H., Hartenstein, S. & Pohl, K. *An Analysis of Software Quality Attributes and Their Contribution to Trustworthiness.* in *CLOSER* (2013), 542–552.

184. Montaner, M., López, B. & De La Rosa, J. L. A taxonomy of recommender agents on the internet. *Artificial intelligence review* **19,** 285–330 (2003).

185. Moro, S., Cortez, P. & Rita, P. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems* **62,** 22–31 (2014).

186. Nasr, M., Shokri, R. & Houmansadr, A. Comprehensive privacy analysis of deep learning: Stand-alone and federated learning under passive and active white-box inference attacks. *arXiv preprint arXiv:1812.00910* (2018).

187. Nasr, M., Shokri, R. & Houmansadr, A. *Comprehensive privacy analysis of deep learning* in *2019 ieee symposium on security and privacy* (2019).

188. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B. & Ng, A. Y. *Reading digits in natural images with unsupervised feature learning* in (2011).

189. Newberry, C. *47 Facebook Stats That Matter to Marketers in 2021* https://blog.hootsuite.com/facebook-statistics/. Online; accessed 11.10.2021.

190. Nguyen, T. D., Marchal, S., Miettinen, M., Fereidooni, H., Asokan, N. & Sadeghi, A.-R. *DÏoT: A federated self-learning anomaly detection system for IoT* in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)* (2019), 756–767.

191. Nguyen, T. D., Rieger, P., Miettinen, M. & Sadeghi, A.-R. Poisoning Attacks on Federated Learning-based IoT Intrusion Detection System (2020).

192. Ni, J., Li, J. & McAuley, J. *Justifying recommendations using distantly-labeled reviews and fine-grained aspects* in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint*

Conference on Natural Language Processing (EMNLP-IJCNLP) (2019), 188–197.

193. Nov, O. & Wattal, S. *Social computing privacy concerns: antecedents and effects* in *Proceedings of the SIGCHI conference on human factors in computing systems* (2009), 333–336.

194. Noyes, D. *Distribution of Twitter users worldwide as of April 2021, by age group* https://www.statista.com/statistics/283119/age-distribution-of-global-twitter-users/. Online; accessed 17.01.2022. 2021.

195. Noyes, D. *Distribution of Twitter users worldwide as of January 2021, by gender* https://www.statista.com/statistics/828092/distribution-of-users-on-twitter-worldwide-gender/. Online; accessed 10.02.2021. 2021.

196. Noyes, D. *Leading countries based on number of Twitter users as of October 2021* https://www.statista.com/statistics/242606/number-of-active-twitter-users-in-selected-countries/. Online; accessed 18.01.2022. 2021.

197. Noyes, D. *Percentage of U.S. adults with select education background who use Twitter as of February 2021* https://www.statista.com/statistics/265648/share-of-us-internet-users-who-use-twitter-by-educational-degree/. Online; accessed 17.01.2022. 2021.

198. Of TFF, A. *TensorFlow Federated* https://www.tensorflow.org/federated. Online; accessed 02.03.2021. 2019.

199. Olivero, N. & Lunt, P. Privacy versus willingness to disclose in e-commerce exchanges: The effect of risk awareness on the relative role of trust and control. *Journal of economic psychology* **25,** 243–262 (2004).

200. Osadchiy, T., Poliakov, I., Olivier, P., Rowland, M. & Foster, E. Recommender system based on pairwise association rules. *Expert Systems with Applications* **115,** 535–542 (2019).

201. Pallis, G., Zeinalipour-Yazti, D. & Dikaiakos, M. D. Online social networks: status and trends. *New directions in web data management 1,* 213–234 (2011).

202. Pan, Y., Huo, Y., Tang, J., Zeng, Y. & Chen, B. Exploiting relational tag expansion for dynamic user profile in a tag-aware ranking recommender system. *Information Sciences* **545,** 448–464 (2021).

203. Papernot, N., McDaniel, P., Sinha, A. & Wellman, M. P. *Sok: Security and privacy in machine learning* in *2018 IEEE European Symposium on Security and Privacy (EuroS&P)* (2018), 399–414.

204. Park, D. *Analysis vs. Analytics: Past vs. Future* https://www.eetimes.com/analysis-vs-analytics-past-vs-future/. Online; accessed 16.10.2021.

205. Parthasarathy, S. *Efficient progressive sampling for association rules* in *null* (2002), 354.

206. Paul, T., Famulari, A. & Strufe, T. A survey on decentralized online social networks. *Computer Networks* **75,** 437–452 (2014).

207. *Peepeth-stats-Free peeping* `https://peepeth.com/stats` and `https://peepeth.com/a/free`. Online, accessed 29.01.2019.

208. Petersen, K., Feldt, R., Mujtaba, S. & Mattsson, M. *Systematic mapping studies in software engineering* in *12th International Conference on Evaluation and Assessment in Software Engineering (EASE) 12* (2008), 1–10.

209. Petersen, K., Vakkalanka, S. & Kuzniarz, L. Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology* **64,** 1–18 (2015).

210. Pfitzmann, A. & Hansen, M. *A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management* 2010.

211. Pogorelov, K., Randel, K. R., Griwodz, C., Eskeland, S. L., de Lange, T., Johansen, D., Spampinato, C., Dang-Nguyen, D.-T., Lux, M., Schmidt, P. T., *et al. Kvasir: A multi-class image dataset for computer aided gastrointestinal disease detection* in *Proceedings of the 8th ACM on Multimedia Systems Conference* (2017), 164–169.

212. Prabhu, V. U. Kannada-MNIST: A new handwritten digits dataset for the Kannada language. *arXiv preprint arXiv:1908.01242* (2019).

213. Pustozerova, A. & Mayer, R. *Information Leaks in Federated Learning* in *Proceedings of the Network and Distributed System Security Symposium* (2020).

214. Qian, J. & Hansen, L. K. What can we learn from gradients? *arXiv preprint arXiv:2010.15718* (2020).

215. Qian, J., Nassar, H. & Hansen, L. K. Minimal conditions analysis of gradient-based reconstruction in Federated Learning. *arXiv preprint arXiv:2010.15718* (2020).

216. Rezgui, Y. & Marks, A. Information security awareness in higher education: An exploratory study. *Computers & security* **27,** 241–253 (2008).

217. Ribeiro, B. & Towsley, D. *Estimating and sampling graphs with multidimensional random walks* in *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement* (2010), 390–403.

218. Rigaki, M. & Garcia, S. A Survey of Privacy Attacks in Machine Learning. *arXiv preprint arXiv:2007.07646* (2020).

219. Al-Rubaie, M. & Chang, J. M. Privacy-Preserving Machine Learning: Threats and Solutions. *IEEE Security and Privacy* (2019).

220. Russakovsky, O. *et al.* ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)* **115,** 211–252 (2015).

221. Ryffel, T., Trask, A., Dahl, M., Wagner, B., Mancuso, J., Rueckert, D. & Passerat-Palmbach, J. A generic framework for privacy preserving deep learning. *arXiv preprint arXiv:1811.04017* (2018).

222. Salem, A., Zhang, Y., Humbert, M., Berrang, P., Fritz, M. & Backes, M. *ML-Leaks: Model and Data Independent Membership Inference Attacks and Defenses on Machine Learning Models* in (2019).

223. Salganik, M. J. & Heckathorn, D. D. Sampling and estimation in hidden populations using respondent-driven sampling. *Sociological methodology* **34,** 193–240 (2004).

224. Salzberg, M. *Kickstarter Pitch* https://web.archive.org/web/20110814222702/http://blog.joindiaspora.com/2010/04/27/kickstarter-pitch.html. Online, accessed 21.03.2019. 2010. https://web.archive.org/web/20110814222702/http://blog.joindiaspora.com/2010/04/27/kickstarter-pitch.html.

225. Samaria, F. S. & Harter, A. C. *Parameterisation of a stochastic model for human face identification* in *Proceedings of 1994 IEEE workshop on applications of computer vision* (1994), 138–142.

226. Schaeffer, R. National information assurance (ia) glossary. *CNSS Secretariat, NSA, Ft. Meade* (2010).

227. Scheaffer, R. L., Mendenhall III, W., Ott, R. L. & Gerow, K. G. *Elementary survey sampling* (Cengage Learning, 2011).

228. Schneider, F., Feldmann, A., Krishnamurthy, B. & Willinger, W. *Understanding online social network usage from a network perspective* in *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement* (2009), 35–48.

229. Schulz, S. & Strufe, T. *d 2 Deleting Diaspora: Practical attacks for profile discovery and deletion* in *2013 IEEE International Conference on Communications (ICC)* (2013), 2042–2046.

230. Shakimov, A., Varshavsky, A., Cox, L. P. & Cáceres, R. *Privacy, cost, and availability tradeoffs in decentralized osns* in *Proceedings of the 2nd ACM workshop on Online social networks* (2009), 13–18.

231. Shejwalkar, V., Houmansadr, A., Kairouz, P. & Ramage, D. Back to the drawing board: A critical evaluation of poisoning attacks on federated learning. *arXiv preprint arXiv:2108.10241* (2021).

232. Shen, M., Wang, H., Zhang, B., Zhu, L., Xu, K., Li, Q. & Du, X. Exploiting Unintended Property Leakage in Blockchain-Assisted Federated Learning for Intelligent Edge Computing. *IEEE Internet of Things Journal* (2020).

233. Shen, S., Tople, S. & Saxena, P. *Auror: Defending against poisoning attacks in collaborative deep learning systems* in *Proceedings of the 32nd Annual Conference on Computer Security Applications* (2016), 508–519.

234. Shi, X., Chen, S. & Yang, H. *DFPS: Distributed FP-growth algorithm based on Spark* in *2017 IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)* (2017), 1725–1731.

235. Shokri, R. & Shmatikov, V. *Privacy-preserving deep learning* in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security* (2015), 1310–1321.

236. Shokri, R., Stronati, M., Song, C. & Shmatikov, V. *Membership Inference Attacks Against Machine Learning Models* in *Proceedings - IEEE Symposium on Security and Privacy* (2017).

237. Shokri, R., Stronati, M., Song, C. & Shmatikov, V. *Membership inference attacks against machine learning models* in *2017 IEEE Symposium on Security and Privacy (SP)* (2017), 3–18.

238. Simonyan, K. & Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

239. Sivanathan, A., Gharakheili, H. H., Loi, F., Radford, A., Wijenayake, C., Vishwanath, A. & Sivaraman, V. Classifying IoT devices in smart environments using network traffic characteristics. *IEEE Transactions on Mobile Computing* **18,** 1745–1759 (2018).

240. Smith, H. J., Milberg, S. J. & Burke, S. J. Information privacy: measuring individuals' concerns about organizational practices. *MIS quarterly,* 167–196 (1996).

241. So, J., Güler, B. & Avestimehr, A. S. Turbo-Aggregate: Breaking the Quadratic Aggregation Barrier in Secure Federated Learning. *IEEE Journal on Selected Areas in Information Theory,* 1–1 (2021).

242. Song, C., Ristenpart, T. & Shmatikov, V. *Machine learning models that remember too much* in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (2017), 587–601.

243. Song, M., Wang, Z., Zhang, Z., Song, Y., Wang, Q., Ren, J. & Qi, H. Analyzing user-level privacy attack against federated learning. *IEEE Journal on Selected Areas in Communications* **38,** 2430–2444 (2020).

244. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* **15,** 1929–1958 (2014).

245. Srivastava, Y., Murali, V. & Dubey, S. R. *A Performance Evaluation of Loss Functions for Deep Face Recognition* in *National Conference on Computer Vision, Pattern Recognition, Image Processing, and Graphics* (2019), 322–332.

246. Statista. *Device usage of Facebook users* https://www.statista.com/statistics/377808/distribution-of-facebook-users-by-device/. Online; accessed 09.05.2019.

247. Statista. *Number of monthly active Facebook users worldwide as of 2nd quarter 2021* https://www.statista.com/statistics/264810/number-of-monthly-active-facebook-users-worldwide/. Online; accessed 11.10.2021.

248. Statista. *Number of monthly active Twitter users worldwide from 1st quarter 2010 to 1st quarter 2019* https://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/. Online; accessed 11.10.2021.

249. Stutzbach, D., Rejaie, R., Duffield, N., Sen, S. & Willinger, W. *Sampling techniques for large, dynamic graphs* in *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications* (2006), 1–6.

250. Stutzbach, D., Rejaie, R., Duffield, N., Sen, S. & Willinger, W. On unbiased sampling for unstructured peer-to-peer networks. *IEEE/ACM Transactions on Networking* **17,** 377–390 (2008).

251. Subramanian, L., Agarwal, S., Rexford, J. & Katz, R. H. *Characterizing the Internet hierarchy from multiple vantage points* in *Proceedings. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies* **2** (2002), 618–627.

252. Sun, G., Cong, Y., Dong, J., Wang, Q. & Liu, J. Data Poisoning Attacks on Federated Machine Learning. *arXiv preprint arXiv:2004.10020* (2020).

253. Sun, Z., Kairouz, P., Suresh, A. T. & McMahan, H. B. Can You Really Backdoor Federated Learning? *arXiv preprint arXiv:1911.07963* (2019).

254. Tan, M. & Le, Q. *Efficientnet: Rethinking model scaling for convolutional neural networks* in *International Conference on Machine Learning* (2019), 6105–6114.

255. Tang, H., Lian, X., Yan, M., Zhang, C. & Liu, J. D^2: Decentralized Training over Decentralized Data. *arXiv preprint arXiv:1803.07068* (2018).

256. Tassa, T. Secure mining of association rules in horizontally distributed databases. *IEEE Transactions on Knowledge and Data Engineering* **26,** 970–983 (2014).

257. *The Federation-a statistics hub-Diaspora* stats* https://the-federation.info/diaspora. Online, accessed 12.10.2021.

258. Toivonen, H. *Sampling large databases for association rules* in *VLDB* **96** (1996), 134–145.

259. Tolpegin, V., Truex, S., Gursoy, M. E. & Liu, L. *Data Poisoning Attacks Against Federated Learning Systems* in *European Symposium on Research in Computer Security* (2020), 480–501.

260. Tomsett, R., Chan, K. & Chakraborty, S. *Model poisoning attacks against distributed machine learning systems* in *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications* **11006** (2019), 110061D.

261. Truex, S., Liu, L., Gursoy, M. E., Yu, L. & Wei, W. Demystifying Membership Inference Attacks in Machine Learning as a Service. *IEEE Transactions on Services Computing* (2019).

262. Tschandl, P., Rosendahl, C. & Kittler, H. The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific data* **5,** 1–9 (2018).

263. Tsuzuku, Y., Imachi, H. & Akiba, T. Variance-based gradient compression for efficient distributed deep learning. *arXiv preprint arXiv:1802.06058* (2018).

264. Tufekci, Z. & King, B. *We Can't Trust Uber* https://www.nytimes.com/2014/12/08/opinion/we-cant-trust-uber.html. 2014.

265. Tulabandhula, T., Vaya, S. & Dhar, A. Privacy-preserving Targeted Advertising. *arXiv preprint arXiv:1710.03275* (2017).

266. Twitter. *Q4 2018 Earnings Report* https://s22.q4cdn.com/826641620/files/doc_financials/2018/q4/Q4-2018-Slide-Presentation.pdf. Online, accessed 21.03.2019. 2019. https://s22.q4cdn.com/826641620/files/doc_financials/2018/q4/Q4-2018-Slide-Presentation.pdf.

267. Twitter. *Facebook, Inc. (FB) Fourth Quarter 2020 Results Conference Call* https://s21.q4cdn.com/399680738/files/doc_financials/2020/q4/FB-Q4-2020-Conference-Call-Transcript.pdf. Online, accessed 11.10.2021. 2021. https://s21.q4cdn.com/399680738/files/doc_financials/2020/q4/FB-Q4-2020-Conference-Call-Transcript.pdf.

268. Twitter. *Get started with the Twitter developer platform* https://developer.twitter.com/en/docs/basics/getting-started. Online, accessed 22.03.2019.

269. Urabe, S., Wang, J., Kodama, E. & Takata, T. A high collusion-resistant approach to distributed privacy-preserving data mining. *Information and Media Technologies* **2,** 821–834 (2007).

270. Verhoeven, B. & Daelemans, W. *CLiPS Stylometry Investigation (CSI) corpus: A Dutch corpus for the detection of age, gender, personality, sentiment and deception in text.* in *LREC* (2014), 3081–3085.

271. Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K. & Wierstra, D. Matching networks for one shot learning. *arXiv preprint arXiv:1606.04080* (2016).

272. Wagner, I. Privacy Policies Across the Ages: Content and Readability of Privacy Policies 1996–2021. *arXiv preprint arXiv:2201.08739* (2022).

273. Wainakh, A., Grube, T., Daubert, J., Porth, C. & Mühlhäuser, M. *Tweet beyond the Cage: A Hybrid Solution for the Privacy Dilemma in Online Social Networks* in *2019 IEEE Global Communications Conference (GLOBECOM)* (2019), 1–6.

274. Wainakh, A., Grube, T., Daubert, J. & Mühlhäuser, M. *Efficient privacy-preserving recommendations based on social graphs* in *Proceedings of the 13th ACM Conference on Recommender Systems* (2019), 78–86.

275. Wainakh, A., Grube, T. & Max, M. *Tweet beyond the Cage: A Hybrid Solution for the Privacy Dilemma in Online Social Networks* in *IEEE Global Communications Conference* (2019).

276. Wainakh, A., Guinea, A. S., Grube, T. & Mühlhäuser, M. *Enhancing privacy via hierarchical federated learning* in *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)* (2020), 344–347.

277. Wainakh, A., Müßig, T., Grube, T. & Mühlhäuser, M. *Label leakage from gradients in distributed machine learning* in *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)* (2021), 1–4.

278. Wainakh, A., Strassheim, A., Grube, T., Daubert, J. & Mühlhäuser, M. *Enabling Privacy-Preserving Rule Mining in Decentralized Social Networks* in *The 16th International Conference on Availability, Reliability and Security* (2021), 1–11.

279. Wainakh, A., Ventola, F., Müßig, T., Keim, J., Cordero, C. G., Zimmer, E., Grube, T., Kersting, K. & Mühlhäuser, M. *User-Level Label Leakage from Gradients in Federated Learning* in *Proceedings on Privacy Enhancing Technologies (PETS)* [to appear] (2022).

280. Wainakh, A., Zimmer, E., Subedi, S., Keim, J., Grube, T., Karuppayah, S., Guinea, A. S. & Mühlhäuser, M. *Federated Learning Attacks Revisited: A Critical Discussion of Gaps, Assumptions, and Evaluation Setups* in *IEEE Access* [under review] (2022).

281. Wang, L., Xu, S., Wang, X. & Zhu, Q. Eavesdrop the Composition Proportion of Training Labels in Federated Learning. *arXiv preprint arXiv:1910.06044* (2019).

282. Wang, T., Chen, Y., Zhang, Z., Sun, P., Deng, B. & Li, X. *Unbiased sampling in directed social graph* in *Proceedings of the ACM SIGCOMM 2010 conference* (2010), 401–402.

283. Wang, T., Chen, Y., Zhang, Z., Xu, T., Jin, L., Hui, P., Deng, B. & Li, X. *Understanding graph sampling algorithms for social network analysis* in *2011 31st international conference on distributed computing systems workshops* (2011), 123–128.

284. Wang, X., Peng, Y., Lu, L., Lu, Z., Bagheri, M. & Summers, R. M. *Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised*

classification and localization of common thorax diseases in *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), 2097–2106.

285. Wang, Y., Deng, J., Guo, D., Wang, C., Meng, X., Liu, H., Ding, C. & Rajasekaran, S. SAPAG: A Self-Adaptive Privacy Attack From Gradients. *arXiv preprint arXiv:2009.06228* (2020).

286. Wang, Y. & Chen, Y. *A new association rules mining method based on ontology theory* in *2012 IEEE Fifth International Conference on Advanced Computational Intelligence (ICACI)* (2012), 287–291.

287. Wang, Z., Song, M., Zhang, Z., Song, Y., Wang, Q. & Qi, H. *Beyond Inferring Class Representatives: client-Level Privacy Leakage from Federated Learning* in *Proceedings - IEEE INFOCOM* (2019).

288. Wang, Z., Song, M., Zhang, Z., Song, Y., Wang, Q. & Qi, H. *Beyond inferring class representatives: User-level privacy leakage from federated learning* in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications* (2019), 2512–2520.

289. Wei, W., Liu, L., Loper, M., Chow, K.-H., Gursoy, M. E., Truex, S. & Wu, Y. *A Framework for Evaluating Client Privacy Leakages in Federated Learning* in *European Symposium on Research in Computer Security* (2020), 545–566.

290. Whitten, A. & Tygar, J. D. *Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0* in *Proceedings of the 8th USENIX Security Symposium* (1999).

291. Wieringa, R., Maiden, N., Mead, N. & Rolland, C. Requirements engineering paper classification and evaluation criteria: a proposal and a discussion. *Requirements engineering* **11,** 102–107 (2006).

292. Wilson, C., Steinbauer, T., Wang, G., Sala, A., Zheng, H. & Zhao, B. Y. *Privacy, availability and economics in the polaris mobile social network* in *Proceedings of the 12th Workshop on Mobile Computing Systems and Applications* (2011), 42–47.

293. Wu, F. *PLFG: A Privacy Attack Method Based on Gradients for Federated Learning* in *International Conference on Security and Privacy in Digital Economy* (2020), 191–204.

294. Xiao, H., Rasul, K. & Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747* (2017).

295. Xie, C., Koyejo, O. & Gupta, I. *Fall of empires: Breaking Byzantine-tolerant SGD by inner product manipulation* in *Uncertainty in Artificial Intelligence* (2020), 261–270.

296. Xu, G., Li, H., Liu, S., Yang, K. & Lin, X. VerifyNet: Secure and verifiable federated learning. *IEEE Transactions on Information Forensics and Security* **15,** 911–926 (2019).

297. Xu, M. & Li, X. *Subject Property Inference Attack in Collaborative Learning* in *2020 12th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)* **1** (2020), 227–231.

298. Xu, X., Wu, J., Yang, M., Luo, T., Duan, X., Li, W., Wu, Y. & Wu, B. *Information Leakage by Model Weights on Federated Learning* in *Proceedings of the 2020 Workshop on Privacy-Preserving Machine Learning in Practice* (2020), 31–36.

299. Yang, D., Zhang, D. & Qu, B. Participatory cultural mapping based on collective behavior data in location-based social networks. *ACM Transactions on Intelligent Systems and Technology (TIST)* **7,** 1–23 (2016).

300. Yang, H. Young American consumers' prior negative experience of online disclosure, online privacy concerns, and privacy protection behavioral intent. *Journal of Consumer Satisfaction, Dissatisfaction & Complaining Behavior* **25,** 179–202 (2012).

301. Yang, Q., Liu, Y., Cheng, Y., Kang, Y., Chen, T. & Yu, H. Federated learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning* **13,** 1–207 (2019).

302. Yang, T., Andrew, G., Eichner, H., Sun, H., Li, W., Kong, N., Ramage, D. & Beaufays, F. Applied federated learning: Improving google keyboard query suggestions. *arXiv preprint arXiv:1812.02903* (2018).

303. Yates, A., Nogueira, R. & Lin, J. *Pretrained Transformers for Text Ranking: BERT and Beyond* in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining* (2021), 1154–1156.

304. Yin, D., Chen, Y., Ramchandran, K. & Bartlett, P. Byzantine-robust distributed learning: Towards optimal statistical rates. *arXiv preprint arXiv:1803.01498* (2018).

305. Zhang, H., Chen, W., Tian, J., Wang, Y. & Jin, Y. Show, Attend and Translate: Unpaired Multi-Domain Image-to-Image Translation with Visual Attention. *arXiv preprint arXiv:1811.07483* (2018).

306. Zhang, J., Chen, B., Cheng, X., Binh, H. T. T. & Yu, S. PoisonGAN: Generative Poisoning Attacks against Federated Learning in Edge Computing Systems. *IEEE Internet of Things Journal* (2020).

307. Zhang, J., Chen, J., Wu, D., Chen, B. & Yu, S. *Poisoning attack in federated learning using generative adversarial nets* in *Proceedings - 2019 18th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/13th IEEE International Conference on Big Data Science and Engineering, TrustCom/BigDataSE 2019* (2019).

308. Zhang, J., Li, C., Ye, J. & Qu, G. *Privacy Threats and Protection in Machine Learning* in *Proceedings of the 2020 on Great Lakes Symposium on VLSI* (2020), 531–536.

309. Zhang, J., Zhang, J., Chen, J. & Yu, S. *GAN Enhanced Membership Inference: A Passive Local Attack in Federated Learning* in *ICC 2020-2020 IEEE International Conference on Communications (ICC)* (2020), 1–6.

310. Zhang, N., Li, M. & Lou, W. *Distributed data mining with differential privacy* in *2011 IEEE international conference on Communications (ICC)* (2011), 1–5.

311. Zhang, N., Paluri, M., Taigman, Y., Fergus, R. & Bourdev, L. *Beyond frontal faces: Improving person recognition using multiple cues* in *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015), 4804–4813.

312. Zhang, Z., Pedrycz, W. & Huang, J. Efficient frequent itemsets mining through sampling and information granulation. *Engineering Applications of Artificial Intelligence* **65,** 119–136 (2017).

313. Zhao, B., Mopuri, K. R. & Bilen, H. iDLG: Improved Deep Leakage from Gradients. *arXiv preprint arXiv:2001.02610* (2020).

314. Zhu, H., Li, Z., Cheah, M. & Goh, R. S. M. Privacy-preserving weighted federated learning within oracle-aided MPC framework. *arXiv preprint arXiv:2003.07630* (2020).

315. Zhu, J. & Blaschko, M. R-GAP: Recursive Gradient Attack on Privacy. *arXiv preprint arXiv:2010.07733* (2020).

316. Zhu, L., Liu, Z. & Han, S. *Deep leakage from gradients* in *Advances in Neural Information Processing Systems* (2019), 14747–14756.

317. Zhu, Y., Huang, L., Yang, W. & Yuan, X. Efficient collusionresisting secure sum protocol. *Chinese Journal of Electronics* **20,** 407–413 (2011).

DECLARATION

I hereby confirm that the submitted thesis with the title "On Privacy-Enhanced Distributed Analytics in Online Social Networks" has been done independently and without use of others than the indicated aids. I assure that I have not previously or concurrently applied for the opening of a promotion procedure with the doctoral thesis submitted here.

Darmstadt, February 7, 2022

Aidmar Wainakh,
February 7, 2022