On the Robustness and Privacy of Distributed Machine Learning

by

Tian Liu

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
August 6, 2022

Keywords: privacy, robustness, distributed learning, federated learning, IoT

Approved by

Tao Shu, Chair, Associate Professor of Computer Science and Software Engineering
Gerry Dozier, Charles D. McCrary Eminent Chair Professor of Computer Science and Software Engineering
Bo Liu, Associate Professor of Computer Science and Software Engineering
David Umphress, COLSA Corporation Cyber Security and Information Assurance Professor of Computer Science and Software Engineering
Xiaowen Gong, Assistant Professor of Electrical and Computer Engineering

Abstract

Machine learning has recently gained tremendous interest due to its capabilities in producing predictive models in a wide variety of applications, such as objective detection and recommendation services. Meanwhile, the development of the Internet of Things (IoT), which enables the connection to the Internet and the computation capability to a wide range of devices, makes it possible for machine learning algorithms to gain insight from an aggregation of physically separated devices. However, due to its distributed nature, one cannot guarantee the legitimacy of the received data or parameters, which provides a venue for new attacks. Therefore, it is necessary to better understand the vulnerabilities and identify potential threats, so as to propose countermeasures to eliminate the impacts of such threats before applications are put into use.

This dissertation focuses on improving the robustness and privacy of distributed learning algorithms and covers both traditional distributed learning systems, in which a central server collects the data and performs the training, and the modern federated learning scheme, in which the training is performed on individual devices. In the background of the transition from traditional power grid to smart grid, the first proposed research studies the robustness of the artificial neural network (ANN) based state estimator by adversarial false data injection attacks. The state estimation of the grid can be misled by injecting noise-like data into a small portion of electricity meters. Focusing on the modern federated learning (FL) scheme, the second proposed research overcomes the ineffectiveness of the backdoor attacks on FL due to the dilution effect from normal users, by utilizing the information leakage from the shared model. The third proposed research provides a high-accuracy and low-cost solution for privacy preservation in mobile edge computing (MEC) systems, in which the key challenges come from computation and power constraints. This dissertation could help people better understand these vulnerabilities and design a safer and more efficient distributed learning system.

Acknowledgments

First, I express my special thanks to my advisor, Dr. Tao Shu. This dissertation would not have been possible without his tremendous support, patience, encouragement, and guidance during my Ph.D. study. Dr. Shu gave me the freedom to explore different topics and guided me toward the correct research direction. It was under Dr. Shu's generous help that I built up my research and learned how to conduct my research on cutting-edge topics.

I also would like to express my thanks to my committee members Dr. Gerry Dozier, Dr. Bo Liu, Dr. David Umphress, and Dr. Xiaowen Gong for their support and valuable comments to improve this dissertation. I am also gratefully acknowledge Dr. Diep Nyuen from University of Sydney Technology for the comments that I received.

Funding for my study has been supported by the Department of Computer Science and Software Engineering and National Science Foundation awards, CNS-2006998 and CNS-1837034. It is a great honor to work and collaborate with colleagues in the Wireless Networking and Security Lab (WINGS), particularly Drs. Li Sun, Jing Hou, and Jian Chen, Miss. Hairuo Xu, Mr. Xueyang Hu, and Mr. Amit Das.

My deepest gratitude goes to my family members, especially my dear husband Dr. Yecheng Xu. I would not have made it anywhere near here without their loving support.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and Motivation

By the end of 2022, there will be 18 billions Internet of Things (IoT) devices connected to the Internet to provide monitoring and computing services [10]. Meanwhile, machine learning applications have gained wide-spread prominence, particularly by the deployment of the powerful neural networks in various application domains, such as object detection, recommendation, natural language processing, and medicine. The explosion of IoT combined with the recent progress in machine learning makes it possible to learn from data on massive physically distributed devices. Currently, distributed learning applications thrive in the prediction of next words and emoji on smartphones [22], environmental monitoring [40], and aiding in medical diagnosis among hospitals [122].

Since IoT devices are usually physically separated, machine learning models can be trained in a traditional distributed or modern federated manner. In traditional distributed learning systems, data are collected by a cloud server or a data center, on which training is performed. However, with the tremendous growth of data generated/collected by IoT devices, offloading a huge amount of data to remote servers could be infeasible due to the required network resources and the incurred latency. Furthermore, the direct transmission of the data is at risk of privacy leakage. Recently, the concept of federated learning (FL) has emerged as a modern distributed learning scheme. Technically, FL is a distributed learning scheme that allows multiple devices to collaborate to train a high-accuracy model without sharing their actual datasets. Instead of sending the original data to a remote server and letting the server perform the training, FL training is performed individually on devices, and the devices only send the trained

model parameters to the central server, in which the model aggregation is performed. As a result, communication and latency are reduced, and privacy is preserved since only the model parameters are sent to the server.

Although more and more models learned from massive IoT devices are expected to be used in our daily lives, the vulnerabilities of distributed learning systems have not yet been well understood. Due to the ubiquitous IoT devices and their low costs, an attacker could easily pry users' privacy or tamper with the trained model by compromising a number of devices. Therefore, it is necessary to have a better understanding of their vulnerabilities, to identify threats, and propose countermeasures to eliminate the impacts of such threats before the models are put into use. Threats to distributed learning systems can be classified mainly into the following two classes:

- **Model robustness.** Robustness means that a model is resilient to small variations, such as outliers and small perturbations of inputs. Due to the nature of distributed learning systems and the inherent data non-i.i.d.-ness across all devices, the data or model parameters uploaded by a client can be different from others. It is difficult for a cloud server to validate the legitimacy/truthfulness of the received data or model parameters. As a result, the model parameters trained from extreme non-i.i.d. but normal data could be falsely rejected by the server, whereas an attacker can deliberately camouflage the malicious model/data to circumvent the detection mechanism. One of the famous robustness attacks is the adversarial example [38], in which the attacker injects a vector of well-coordinate perturbations to a data sample such that the tampered data sample is mis-predicted by the trained model. Another example is the backdoor attack [9], in which the adversary injects a pattern into the training data to corrupt the model during the training process such that the new model is equally accurate on the main training task, while performing well on a sub-task activated by some triggers.

- **Model/data privacy.** Contrary to the initial belief that FL is private because only the trained model updates are transmitted and no users' data is directly revealed, recent studies have found that shared FL model updates may unintentionally leak sensitive information about the data on which it was trained [28]. As pointed out by previous studies, using

FL scheme alone is insufficient in protecting the clients' local data privacy. For example, from the FL model, an adversary can infer if a given data sample was presented in the training data or not [79, 86], or recover representative data sample used in the training [34], or infer property information about the client's local training data [132].

The overarching goal of this dissertation is to obtain a comprehensive understanding on the security vulnerabilities of distributed learning systems, especially from the perspectives of model robustness and model/data privacy, and to develop a solid mathematical framework that can be used to characterize the vulnerabilities and improve the utility of existing learning algorithms and defense mechanisms. In the first work, it is examined whether the vulnerability of adversarial examples presented in the image classification problem also exists in the state estimation problem in the smart grid. In the second work, considering the ineffectiveness of single-shot backdoor attacks against FL due to the dilution effect from normal model updates especially in the early training stage, a novel information leakage assisted two-phase FL backdoor attack, which enhances the effectiveness of FL early-injected single-shot backdoor attack has been proposed. The third work focuses on the privacy-preserving solution for IoT devices, which is limited by computation capability, power supply, network connectivity, and participation flexibility. A low-cost (for both communication and computation overhead) adaptive noise perturbation privacy preserving scheme is then proposed, which does not sacrifice model accuracy for privacy, while enjoying differential privacy (DP) comparable privacy protection.

1.2 Overview of Research Contributions

1.2.1 Adversarial False Data Injection Attack against ANN-based State Estimation in Smart Grid

In this work, a new study of adversarial false data injection attacks against artificial neural network (ANN)-based state estimation is initiated. By injecting a deliberate attack vector into the measurements, the attacker can degrade the accuracy of an ANN-based state estimate while remaining undetected. Two algorithms to generate the attack vectors are proposed, one population-based algorithm (differential evolution or DE) and one gradient-based algorithm

(sequential least square quadratic programming or SLSQP). The researcher then evaluates these algorithms through simulations on IEEE 9-bus, 14-bus, and 30-bus systems. Simulation results show that DE is more effective than SLSQP on all simulation cases. The attack examples generated by the DE algorithm successfully degrade the accuracy of the ANN state estimation result with high probability (more than 80% in all simulation cases), despite having a small number of compromised meters and low injection strength. The potential defense strategy to mitigate such attacks is further discussed, which provides insight for robustness improvement in future research.

1.2.2 Assisting Backdoor FL with Whole Population Knowledge Alignment

In this work, the early-injected single-shot backdoor attack against FL is strengthened by utilizing the information leaked from the shared FL model. Theoretical analysis shows that FL convergence can be expedited if the client trains on a dataset that mimics the distribution and gradients of the whole population. On the basis of this observation, a two-phase backdoor attack is proposed, including a preliminary phase for the subsequent backdoor attack. In the preliminary phase, the attacker-controlled client first launches a whole population distribution inference attack and then trains on a locally crafted dataset that is aligned with both the gradient and the inferred distribution. Benefiting from the preliminary phase, the later injected backdoor achieves better effectiveness, as the backdoor effect will be less likely to be diluted by the normal model updates. Extensive experiments are conducted to evaluate the effectiveness of the proposed backdoor attack. The results show that the proposed backdoor outperforms existing backdoor attacks in both success rate and longevity, even when defense mechanisms are in place.

1.2.3 High-Accuracy Low-Cost Privacy-Preserving FL in IoT Systems via Adaptive Perturbation

In this work, the high accuracy of the FL model is retained while protecting user privacy by taking into account both the magnitude and direction of the additive perturbation. In particular, the magnitude of the additive noise is set to adaptively change with the magnitude of the local

mode updates. Then a direction-based filtering scheme is used to expedite the FL model convergence. A maximum tolerable variance of the additive noises is derived to maximize privacy protection at local clients, while the FL global model enjoys the same model accuracy and convergence rate as a result of the cancel-out effect presented in the aggregation of noises on the server by the central limit theorem. Theoretically, it is proven that FL with the proposed noise perturbation scheme retains the same accuracy and convergence rate of $\mathcal{O}(1/T)$ as that of a non-private FL (FL with no privacy preservation), in both convex and non-convex loss function scenarios. We also evaluate the performance of the proposed scheme in terms of convergence behavior, time and computation efficiency, and privacy protection against state-of-the-art privacy inference attacks on a real-world dataset. Experimental results show that FL with the proposed perturbation scheme outperforms DP in the accuracy and convergence rate of the FL model in both client dropout and non-client dropout scenarios. Compared to DP, the proposed scheme does not incur additional computation and communication overhead. This approach provides a DP-comparable or better effectiveness in defending against privacy attacks under the same FL model accuracy.

1.3 Publication Contributions

During my Ph.D. study, I have contributed to the following publications (listed chronologically).

[1] **T. Liu** and T. Shu. Adversarial false data injection attack against nonlinear ac state estimation with ANN in smart grid. In *International Conference on Security and Privacy in Communication Systems (SecureComm)*, Springer, 2019.

[2] X. Hu, **T. Liu**, and T. Shu. Fast and high-resolution NLOS beam switching over commercial off-the-shelf mmwave devices. In *IEEE Transactions on Mobile Computing (TMC)*, IEEE, 2021.

[3] **T. Liu** and T. Shu. On the security of ANN-based ac state estimation in smart grid. *Computers & Security*, Elsevier, 2021.

[4] J. Chen, **T. Liu**, and T. Shu. A survey on visible light communication standards. In *Get-Mobile: Mobile Computing and Communications*, 2021.

[5] **T. Liu**, X. Hu, H. Xu, T. Shu, and D. Nguyen. High-accuracy low-cost privacy-preserving federated learning in IOT systems via adaptive perturbation. In *Journal of Information and Security Applications*. Conditionally accepted.

[6] **T. Liu**, X. Hu and T. Shu. Assisting backdoor federated learning with whole population knowledge alignment in mobile edge computing. In *18th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, 2022. Forthcoming.

[7] X. Hu, **T. Liu** and T. Shu, (k, α)-coverage for RIS-aided mmWave directional communication. In *IEEE Transactions on Mobile Computing (TMC)*. Conditionally accepted.

1.4 Dissertation Overview

In the rest of this dissertation, three works are detailed with each addressing a set of problems, deepening the knowledge of the robustness and privacy protection of distributed learning algorithms against adversaries. Each chapter focuses on presenting one work, along with comprehensive evaluations and comparisons between the solutions to the state-of-the-art methods.

In Chapter 2, the robustness of the ANN-based state estimation in smart grids is studied by designing a false data injection attack, which is capable of misleading the state estimate by injecting false noise-like data into meter readings. The designed attack on the IEEE test systems is then evaluated with a defense mechanism to defend against the proposed attacks coming up.

In Chapter 3, the study is presented to improve the early injection of a single-shot backdoor attack against FL by utilizing information leaked from the shared FL model. The attacker facilitates the convergence of the FL model so as to strengthen the effectiveness of the later injected single-shot backdoor. Numerical experiments are then conducted to show the effectiveness of the proposed backdoor attack.

In Chapter 4, a low-cost high-accuracy perturbation-based FL privacy-preserving mechanisms is proposed. The proposed scheme takes both perturbation direction and magnitude into

consideration, so that the perturbations are canceled out on the server and the direction of descent is preserved. Theoretical proofs and numerical results against state-of-the-art attacks are provided.

Finally, future work is discussed in Chapter 5.

Chapter 2

On the Security of ANN-based AC State Estimation in Smart Grid

2.1 Introduction

With the increase in residential and industrial power demand, nowadays a regional or nation-wide power outage often leads to catastrophic consequences in the matter of public safety. After the US Northeast Blackout in 2003, the US and Canada reached a consensus to transition to a smart grid system, which would be cleaner and more efficient, reliable, resilient, and responsive than a traditional grid. The smart grid is a complex system that integrates a traditional power grid and information technologies to enable inter-networking over power grids. Although transferring from the traditional power grid to the smart grid provides many new attractive features such as remote and automatic grid monitoring, control, and pricing, it has also raised serious security challenges by opening up the traditional power system to many potential attacks in cyber space. For example, in the 2015 Ukraine power outage [66, 61], the hacker successfully compromised the information systems of three energy distribution companies and caused power failure to more than 225,000 customers lasting 1 to 6 hours. Since then, cyber attacks on smart grids have caught public attention and have become a realistic and growing concern for governments, vendors, and customers.

One of the key mechanisms in ensuring normal operation of a smart grid is state estimation, which provides the current status of the grid for the control center operators to take corrective action in order to prevent an accident from happening. State estimation aims to compute the states of the system (the complex voltages at all buses [117]) that are not directly measurable, based on the grid's topology and the meter's power usage measurements collected

from the *supervisory control and data acquisition* (SCADA) system. Conventionally, state estimation is formulated as a non-linear *weighted least square* (WLS) problem that minimizes the distance between actual measurements and computed measurements from the estimated state. Such methods have several limitations. First, solvers to the problem, such as Gauss-Newton, are computationally heavy, sensitive to initial values, and may encounter convergence issues. In addition, the state estimation has to be computed periodically for every set of meter measurements collected in each meter reading cycle (typically a 15-minute period) in order to obtain the current system status. Furthermore, a prior observability analysis is often required to ensure that the system is overdetermined. This state estimation scheme is further challenged by the growing grid scale and unprecedented system dynamics caused by the increasing deployment of new elements in the smart grid, such as renewable generators, electric vehicles, and dynamic pricing.

In light of the above issues in existing state estimation methods, *artificial neural networks* (ANN) have received a lot of interest as a new approach to smart grid state estimation for mainly two reasons: (1) the computation cost can be ignored once the model is trained. In particular, once the ANN state estimation model is trained offline based on historical or simulated data, such a model can provide accurate estimation online at minimal computation cost, eliminating the need for carrying out observability analysis prior to running the state estimation. (2) ANNs naturally fit into the non-linear nature of the state estimation problem. So far, several efforts have been made to adopt ANNs for state estimation. It has been established that ANN-based state estimation provides results much faster, and the accuracy is comparable to or higher than that of conventional state estimations.

Although state estimation plays an important role in ensuring the normal operation of the smart grid, it is well known that conventional state estimation methods are vulnerable to *false data injection* (FDI) attacks [74], which is a data integrity cyber-attack and has been proven to be a real threat to the smart grid system. In particular, an adversary can corrupt the state variable by injecting carefully coordinated false data to meter measurements while evading the bad data detection. The injected false data may result in generation re-dispatch [68] or trigger a

branch outage sequence that involves multiple branches and ultimately leads to a system failure [21].

Although FDI attacks against conventional state estimation methods have been well understood in the literature, little is known about FDI attacks against ANN-based state estimation. As ANN-based state estimation is expected to receive more and more applications for the smart grid in the near future, and because the smart grid is a critical infrastructure of society, it is necessary to gain a better understanding of the vulnerabilities of this new state estimation method of FDI attacks, so as to identify possible threats and propose countermeasures to eliminate such threats before this new method can be applied in practice on a larger scale. Therefore, we can reduce the potential loss and increase the confidence of the society in the security feature of the new method.

In contrast to existing FDI attacks that mainly rely on a linear *direct current* (DC) power flow model, FDI attacks against an ANN-based state estimation must accommodate a nonlinear *alternating current* (AC) power flow model, as the nonlinearity is a fundamental feature of the ANN state estimation. As ANN becomes a popular technique in the power system, several works demonstrate the effectiveness of adversarial attacks on power system applications [25, 26, 63]. Unfortunately, little work has been done to analyze the vulnerabilities and robustness of the ANN-based state estimation model.

Meanwhile, in the area of image classification, researchers noticed that ANNs can easily be fooled by well-coordinated samples with small perturbations. This discovery has spurred many efforts to explore the vulnerabilities of ANN by designing adversarial attacks.

In this work, we are interested in examining whether the above vulnerability of ANN present in the image classification problem also exists in the state estimation problem in the smart grid. We create an FDI attack customized for the ANN-based state estimation model. This attack can also be used to construct an upper bound on the robustness of the model. Furthermore, we try to develop algorithms that can systematically generate contaminated measurements that maximize the ANN-based state estimation error while eluding detection by a bad data detector. By doing so, we aim to establish a new understanding of the security vulnerabilities of the latest high-accuracy ANN-based state estimator. To our knowledge, our work

is the first in the literature to study the vulnerabilities and robustness of the ANN-based state estimator by FDI attacks.

Compared with its image classification counterpart, solving our problem faces new and significant challenges. In addition to the obvious difference in the application model, our problem presents the following three novel features in its structure. Firstly, our problem has an optimization nature in the sense that we seek the optimal attack vector that maximizes the attack outcomes. In contrast, the goal of the image-classification counterpart is simply to find a feasible attack vector. Secondly, the attack model in our problem considers the attacker's access and resource constraints, in which the attacker only has access to and can only manipulate a certain number of meters. The attacker's injection is also subject to physical constraints on the smart grid system. In contrast, the image-classification problem does not have such constraints, and the attacker can change any pixel of the image. Lastly, the output of state estimator is a vector of continuous values, whereas that of the image-classification is discrete and covers a limited number of pre-defined cases. Due to these fundamental structural differences, the existing results from the image classification ANN are not directly applicable to our problem, and therefore new solutions need to be developed.

In this work, we study the robustness of ANN-based state estimators by constructing adversarial FDI attacks. We first create ANN-based state estimators as our target models, followed by evaluating both model accuracy and bad data rate to ensure that the target models are sufficiently strong. We then use the idea of an adversarial example to formulate an optimization-based FDI attack. In this model, an attacker attempts to maximize the state estimation error without being reported by the bad data detector, subject to given resource and meter access constraints. Subsequently, two algorithms are proposed to solve the optimization above to find the best false data injection vector: *differential evolution* (DE) and *sequential least square quadratic programming* (SLSQP). We extensively evaluate our proposed attacks based on simulations on IEEE 9-bus, 14-bus, and 30-bus system models under various scenarios to verify their effectiveness.

The main contribution of our work includes the following fivefold:

- In creating the target ANN state estimator for large-scale grid systems (e.g., 30-bus and above), a novel penalty term is proposed for the loss function, which significantly improves the accuracy of the ANN in modeling the voltage phase angle for large-scale grids.

- An optimization-based FDI attack formulation is proposed for the ANN-based AC state estimation model, which can accommodate various practical constraints on the attacker, including their resource and meter accessibility.

- We adapt two algorithms, DE and SLSQP, to solve the above optimization, targeting two different attack scenarios. DE generates attack vectors for the scenario, in which the attacker can compromise any k meters, while both DE and SLSQP can accommodate the scenario, in which the attacker has only access to specific k meters.

- The effectiveness of the proposed attack models is verified by extensive simulations on IEEE 9-bus, 14-bus, and 30-bus systems under various attack scenarios. Our results show that the DE attack is successful with high probability (more than 80% in all simulated cases), despite having a small number of compromised meters and low false injection level.

- We adopt adversarial training to defend against the above attacks. It turns out that adversarial training could lower the attack success rate, but would slightly impair the model accuracy.

The proposed algorithms provide a practical way for systematically identifying key meters whose readings have a higher weight in the state estimation and thus may serve as a guide to the utility company to reach a more focused/concentrated protection against these key meters under resource and budget constraints. Furthermore, our defense strategy encourages the building of more robust ANN-based state estimation models in the future.

The remainder of the chapter is organized as follows. In Section 2.2, we survey the ANN-based state estimation, false data injection attack, and adversarial example. We then provide a preliminary for state estimation and bad data detection in Section 2.3. We construct ANN-based

state estimation models as our attack targets and evaluate their performance in Section 2.4. Subsequently, we introduce our adversary model and attack formulation in Section 2.6. Our two attack algorithms, the DE and SLSQP algorithms, are presented in Section 6. The experimental analysis and the proposed defense are presented in Sections 2.7 and 2.8, respectively.

2.2 Related Work

2.2.1 ANN-based State Estimation

Various neural network architectures are explored for state estimation in the smart grid, such as the feed-forward neural network [3], radial basis function neural network [105], the counter propagation network, and the functional link network [58]. In [88], Onwuachumba *et al.* proposed a reduced ANN-based state estimation model, which uses fewer measurements and no prior observability analysis is required. To adapt to the new features emerging in smart grid, such as renewable generators and dynamic pricing, the ANN-based state estimation for real-time and distributed power systems is studied in [84, 81, 128, 127].

2.2.2 False Data Injection Attack

Existing results on FDI attacks against conventional state estimations are inapplicable to ANN-based state estimation for the following two reasons. First, most previous work on FDI attacks is based on the DC power flow model [74, 99, 33, 50], which is a linear approximation of the real-world AC power flow model and is usually used as a simplified version of the AC power flow model. FDI attacks against AC models are more complicated and hence require a more sophisticated attacker than DC models. The FDI attacks derived from DC models may be ill-suited for AC models [95]. In addition, the works on constructing FDI attacks against AC models are mainly focused on WLS state estimators [50, 53, 110, 114, 67], thus they cannot be directly applied to ANN-based state estimators.

A considerable number of works have been proposed to defend against FDI attacks. The authors in [12] approached the issue by identifying and protecting a set of critical meters to detect FDI attacks. The authors in [19, 62, 100] approached the issue using a statistical method combined with physical laws of the power system. Data-driven and machine learning-based

approaches were proposed in [32, 45, 39, 125, 129]. A Kalman filter-based detector was developed in [76]. Liu *et al.* developed a detection by using the sparsity of attacks [71]. The authors in [65] proposed a sequential detector, and the authors in [49] proposed an adaptive CUSUM algorithm, in order to accelerate the detection process.

2.2.3 Adversarial Examples

Szegedy *et al.* were the first to propose an adversarial attack against deep neural networks [109]. After that, various attack algorithms are proposed, such as the Fast Gradient Sign Method (FGSM) [38], Fast Gradient Value (FGV) [98] and DeepFool [83]. Especially in [107], the deep learning model can be fooled by adding one pixel perturbation to the image. Furthermore, perturbations are shown to be transferable among ANN models, even if they are trained on different data sets, and preserve different architectures [59, 73, 111, 119].

Another branch of research studies defense against adversarial examples. Papernot *et al.* used a distillation network to extract knowledge to improve robustness [89]. In adversarial training, adversarial examples are generated in every training step, then they are injected into the training data set [38, 48, 75]. And in the classifier robustifying, authors in [14, 2] put emphasis on how to design a robust architecture of the ANN.

2.3 Preliminaries

In this section, we briefly introduce the state estimation and the detection of bad data. All notation used is defined in Table 2.1.

2.3.1 State Estimation

In the AC power flow model, measurements are non-linearly dependent on state variables, as characterized by the following equation:

$$\mathbf{z} = h(\mathbf{x}) + \mathbf{e},$$

Table 2.1: Notation and definitions.

Notation	Definitions
n, m	Number of state variables/measurements
$\mathbf{x}, \mathbf{x_a}, \hat{\mathbf{x}}$	Natural/compromised/estimated state variables, including voltage magnitude $\|V_i\|$ and phase angle θ_i at all buses, $i = 1, .., n$
P_i ,Q_i	Real and reactive power injection at bus i.
P_{ij}, Q_{ij}	Real and reactive power injection at branch connecting bus i to bus j
$\mathbf{z}, \mathbf{z_a}$	Natural/compromised measurements, including real and reactive power injection of buses P_i and Q_i and branches P_{ij} and Q_{ij}
$h(\cdot)$	A set of non-linear, deterministic functions that relate states to measurements $h : \mathbf{x} \rightarrow \mathbf{z}$
$f(\cdot)$	ANN-based state estimator that eliminates errors in measurements and output
a	Attack vector that injects to a given measurement **z**
$G_{ij} + jB_{ij}$	The ij-th element of the complex bus admittance matrix
$g_{ij} + jb_{ij}$	The admittance of the series branch connecting busses i and j
$g_{sj} + jb_{sj}$	The admittance of the shunt branch connected at bus i

where $\mathbf{z}$ and $\mathbf{x}$ denote a N_m-dimension measurement vector and a N_n-dimension state vector, respectively, and **e** denotes a N_m-dimension vector of normally distributed measurement errors. $h(\mathbf{x})$ denotes a set of non-linear functions, by which the measurements are related to state variables, according to Kirchhoff's circuit law:

$$P_i = V_i \sum_{j=1}^{N} |V_j|(G_{ij}cos\theta_{ij} + B_{ij}sin\theta_{ij}), \tag{2.1}$$

$$Q_i = V_i \sum_{j=1}^{N} |V_j|(G_{ij}sin\theta_{ij} - B_{ij}cos\theta_{ij}), \tag{2.2}$$

$$P_{ij} = |V_i|^2(g_{si} + g_{ij}) - |V_iV_j|(g_{ij}cos\theta_{ij} + b_{ij}sin\theta_{ij}, \tag{2.3}$$

$$Q_{ij} = -|V_i|^2(b_{si} + b_{ij}) - |V_iV_j|(g_{ij}sin\theta_{ij} - b_{ij}cos\theta_{ij}). \tag{2.4}$$

In an overdetermined case, where we have more measurements than state variables ($N_m > N_n$), the state variables are determined from the WLS optimization over a residual function $J(\mathbf{x})$ [117]:

$$\hat{\mathbf{x}} = \arg\min_{x} J(\mathbf{x}), \text{where } J(\mathbf{x}) = (\mathbf{z} - h(\mathbf{x}))^T \mathbf{W} (\mathbf{z} - h(\mathbf{x})). \tag{2.5}$$

Here, the weight matrix $\mathbf{W}$ is defined as $diag\{\sigma_1^{-2}, \sigma_2^{-2}, ..., \sigma_{N_m}^{-2}\}$, and σ_i^2 is the variance of the i-th measurement ($i = 1, ..., N_m$). $\mathbf{W}$ is introduced to emphasize trusted measurements while de-emphasizing less trusted ones.

2.3.2 Bad Data Detection

Meter measurements may contain errors due to various reasons, such as transmission error, wiring failure, or malicious attack. Therefore, for data quality control purposes, a bad data detection is usually introduced to identify measurements whose error exceeds a pre-defined threshold. Integration of state estimation and bad data detection can largely suppress the presence of bad data and ensure that the state estimation is based only on good data. Most bad data detection schemes rely on the residual $J(\hat{x})$ as a decision variable. In particular, given the assumption that $\mathbf{e}$ is normally distributed, it is shown that $J(x)$ follows a $\chi^2(K)$ distribution, where $K = N_m - N_n$ is the degree of freedom. Any measurements with a residual greater than the pre-determined threshold τ is recognized as bad data:
z is identified as bad data, if

$$J(\hat{\mathbf{x}}) = (\mathbf{z} - h(\hat{\mathbf{x}}))^T \mathbf{W} (\mathbf{z} - h(\hat{\mathbf{x}})) > \tau. \tag{2.6}$$

The threshold τ can be determined by a significant level α in hypothesis testing, indicating that false alarms would occur with probability α.

2.4 ANN-based AC State Estimation

The main difficulty in utilizing Eq.(2.5) directly to estimate the AC state is that it requires solving a non-linear optimization problem. Instead of making any particular assumption on the structure of $h(\cdot)$, we adopt an empirical methodology to characterize the non-linear state estimation function. In particular, based on a sufficient number of empirical state-measurement readings, we attempt to train an ANN model that can accurately represent the states as a non-linear function of the measurements. In the operational phase, this ANN is expected to directly output a state estimate $\hat{\mathbf{x}}$ for each input of the measurements $\mathbf{z}$, without the need to solve the

nonlinear optimization in Eq.(2.5). In the following, we present our procedure for generating the training data, defining the loss function, training the ANNs, and testing the accuracy of the trained ANN state estimators.

2.4.1 Model Training

Although it would be more convincing to use actual data from a real power grid, power companies use their own proprietary data format, in which most of them are not accessible. Therefore, lacking actual state-measurement data from a real power grid, we follow the convention to present our results based on computer simulations, as in previous studies (e.g. [74, 21, 23]). Simulation-based evaluation would give valid results because the simulation data are generated according to realistic grid typologies and well-established physical laws/mechanics that govern the operation of the grids. In addition, simulation data provide a wider range of the operational condition coverage. In particular, real-meter data can only cover a limited set of operational conditions of the grids under which these actual data are recorded, while the simulation data have a much wider coverage on the grids' operation conditions, as these data can be generated on demand for any operation condition of interest.

The training and testing cases in our study are generated by simulations on IEEE test systems (9-bus, 14-bus, 30-bus). A Matlab package, MATPOWER [133], is used for data generation and power flow analysis. Note that the use of simulation data in training does not affect the validity of the proposed ANN model. One can simply replace the simulation data by actual data once they become available and then retrain the ANN by the same procedure.

Our state-measurement data are generated as follows. The state variable, consisting of the magnitude $|V_i|$ and phase angle θ_i of the bus voltages, is a function of the load of the power system and changes within a small range. To account for this dynamic behavior, we consider a series on loads of the power grid ranging from 80% to 120%. For each load instance, the state is calculated by power flow analysis. According to the American National Standard for Code for Electricity Metering [6], class 2 accuracy applies for power grid measurements, which tolerates a $\pm 2\%$ error in a measurement reading. In accordance with this specification, we add an independent Gaussian noise ϵ to each measurement reading ψ, so that the simulated

measurement reading becomes $(1+\epsilon)\psi$, where $\epsilon \sim N(0, 0.67\%^2)$. For each of the test systems, 10,000 and 1,000 state-measurement pairs are generated for training and testing, respectively. Note that all constant values are excluded from measurements and state variables.

An ANN-based state estimation model is trained for each of the test systems. Following [3, 84, 52, 80], each ANN state estimation model possesses a *multi-layered perceptron* (MLP) architecture, consisting of one input layer, one or more hidden layers, and one output layer. We use the mean WLS error as the loss function:

$$loss(\mathbf{z}, \mathbf{x}) = \frac{1}{N} \sum_{i=1}^{N} (\mathbf{z} - h(\mathbf{x}))^T \mathbf{W} (\mathbf{z} - h(\mathbf{x})), \tag{2.7}$$

where N is the number of training samples.

Our experiments show that the accuracies of both voltage magnitude and phase angle are satisfactory, yet the phase angle accuracy is lower. There are several reasons behind this phenomenon. First, the loss function only narrows the difference between the actual and estimated measurements. Being different from conventional machine learning problems, the state estimation requires the error to be minimized from both measurement and state sides. Second, the voltage magnitudes are strictly confined to a small range to provide a stable and consistent power supply.

These trained models serve as targets for our proposed attacks. The inaccuracy in the state estimation, i.e., the deviation of the estimated state from the actual state, overlays the goal of the FDI attack. So any estimation inaccuracy would be counted as an attack success in the attack evaluation. To eliminate such effect, we revise the loss function in order to achieve high accuracies on both voltage magnitude and phase angle. A new penalty term of the *mean square error* (MSE) between the actual state and the estimated state is added in Eq.(2.7), leading to a new loss function in Eq.(2.8) specially designed for large-scale systems. In this new loss function, a small constant c is added to balance both error terms so that the gradient descent works on both terms simultaneously:

$$loss(\mathbf{z}, \mathbf{x}) = \frac{1}{N} \sum_{i=1}^{N} (\mathbf{z} - h(\mathbf{x}))^T \mathbf{W} (\mathbf{z} - h(\mathbf{x})) + c \frac{1}{N} \sum_{i=1}^{N} (\mathbf{x} - \hat{\mathbf{x}})^2. \tag{2.8}$$

Empirically, we investigate the value of c uniformly spaced (on a logarithmic scale) from $c = 1 \times 10^1$ to $c = 1 \times 10^5$, and choose a c that provides the best estimate precision. Our experiments show that by adding this new penalty term, the voltage phase angle estimation accuracy increases to an equivalent level as that of the voltage magnitude. The proposed ANNs are implemented in Python, using the *TensorFlow* package with *Keras* as back-end. The model architectures and parameters are given in Table 2.2.

Table 2.2: ANN-based state estimator architectures and parameters.

	9-bus	**14-bus**	**30-bus**
Architecture			
Input Size	42	103	204
Fully Connected + ReLU	64	128	256
Output Size	14	22	53
Parameter			
Learning Rate	0.001	0.001	0.001
Decay Rate	1×10^{-5}	1×10^{-5}	1×10^{-5}
Batch Size	64	64	64
Epochs	300	500	1000

2.4.2 Model Evaluation

After the models are trained, we use testing data to evaluate their performance. A good state estimation model should have the following two properties: First, it should be able to provide accurate state estimation irrespective of the noise in the measurements; second, regular measurement noises should not trigger bad data alarms (i.e., low false alarm rate). Accordingly, we evaluate the accuracy of the estimation of the ANNs by *maximum absolute error* (MAE) and *maximum absolute relative error* (MARE) between the true and the estimated values, where MARE is simply MAE normalized w.r.t. the magnitude of the true value. An estimate is considered accurate if the MARE of the voltage magnitude and the voltage phase angle do not exceed 1% and 5%, respectively. To evaluate the false alarm rate, we use a significant level of bad data $\alpha = 0.01$. Table 3 and Table 4 summarize the performance evaluation of trained ANN models. It is clear from these tables that the proposed ANN models are able to estimate states accurately and have a low false alarm rate under regular measurement noise.

Table 2.3: Evaluation of the voltage magnitude of the model.

Test System	MAE (p.u.)	MARE	Bad Data(%)	Accuracy(%)
9-bus	2.2×10^{-5}	2.4×10^{-5}	0	100
14-bus	5.8×10^{-5}	5.6×10^{-3}	3	100
30-bus	6.3×10^{-5}	6.5×10^{-5}	5	100

Table 2.4: Evaluation of the voltage angle of the model.

Test System	MAE (rad)	MARE	Accuracy(%)
9-bus	1.0×10^{-4}	1.6×10^{-2}	96
14-bus	6.1×10^{-3}	2.6×10^{-2}	99
30-bus	1.2×10^{-4}	1.3×10^{-2}	98

2.5 Adversarial Model and Attack Formulation

In this section, we present a detailed adversarial model against the ANN-based state estimator, following [126]. This model characterizes the adversary by their goal, knowledge of the data and target system, and resource and meter accessibility constraints. Based on this model, we formulate an optimization problem that the attacker can use to decide their best attack strategy.

2.5.1 Adversarial Model

It is realistic and practical for an attacker to have the ability to compromise meters, given the fact that the meters are physically distributed and lack protection. The goal of the attacker is to launch an FDI attack, in which the attacker aims to inject a manipulated measurement vector, whose ultimate goal is to maximize the state estimation error while remaining undetected. The false data are injected to the compromised meters, then collected by the SCADA system, and eventually sent to the state estimation application.

It is assumed that the attacker has complete knowledge of the topology and configuration of the power grid, such as the nodal admittance matrix. This information can be accessed or estimated from a public database or historical records. In addition, it is also assumed that the attacker knows everything about the ANN-based state estimation model, including the architecture and parameters. These information could be obtained by an attacker either by breaking

into the information system of the power grid (similar to the 2015 Ukraine case) or by training a shadow ANN that mimics the real ANN-based state estimator on a substitute data set. We assume that the attacker also knows the threshold of the bad data detector.

Although these assumptions render a strong attacker that may not always represent the practical cases, they enable us to evaluate the robustness and vulnerabilities of the ANN-based state estimators under the worst-case scenario, which provides an upper bound on the impact of FDI attacks against the ANN-based state estimation.

In addition to the bad data detection threshold, the adversary also faces other constraints, including the set of meters to which they have access, the maximum number of meters they can compromise, and the maximum amount of errors they can inject into the actual measurements without being detected.

Note that in this work, we only consider the FDI attacks that happen during the operational phase of the ANN-based state estimator. In other words, the adversary is only able to alter the measurement inputs after the ANN model is trained. It is not allowed to perturb either the training data or the trained model. Investigating training data or model poisoning is out of the scope of this work and will be studied in our future work.

2.5.2 Attack Formulation

Let $\mathbf{z}_a$ be the measurement vector in the presence of FDI attack, then $\mathbf{z}_a$ can be described as follows:

$$\mathbf{z_a} = \mathbf{z} + \mathbf{a} = h(\mathbf{x}) + \mathbf{a}, \tag{2.9}$$

where $\mathbf{a}$ is a N_m-dimension non-zero attack vector. Given the input of a manipulated measurement $\mathbf{z_a}$, the state estimation output of the ANN-based state estimator f is as follows:

$$\hat{\mathbf{x}}_\mathbf{a} = f(\mathbf{z_a}) = f(\mathbf{z} + \mathbf{a}). \tag{2.10}$$

According to Eq.(2.6), an adversary intending to elude bad data detection must satisfy the following condition:

$$J(\hat{\mathbf{x}}_\mathbf{a}) = (\mathbf{z_a} - h(\hat{\mathbf{x}}_\mathbf{a}))^T \mathbf{W} (\mathbf{z_a} - h(\hat{\mathbf{x}}_\mathbf{a})) \leq \tau. \tag{2.11}$$

The error injected into the state estimation hence can be calculated by:

$$\hat{\mathbf{x}}_\mathbf{a} - \hat{\mathbf{x}} = f(\mathbf{z_a}) - f(\mathbf{z}). \tag{2.12}$$

With the above notation, the problem of finding the best adversarial injection $\mathbf{a}$ for a given measurement $\mathbf{z}$ can be formulated as a constrained optimization.

$$\begin{aligned} \underset{\mathbf{a}}{\text{maximize}} \quad & \|\hat{\mathbf{x}}_\mathbf{a} - \hat{\mathbf{x}}\|_p \\ \text{subject to} \quad & (\mathbf{z_a} - h(\hat{\mathbf{x}}_\mathbf{a}))^T \mathbf{W} (\mathbf{z_a} - h(\hat{\mathbf{x}}_\mathbf{a})) < \tau, \\ & \|\mathbf{a}\|_0 \leq L, \\ & a_i^l \leq a_i \leq a_i^u, i = 1, ..., N_m, \\ & z_i^{min} \leq z_{a_i} \leq z_i^{max}, i = 1, ..., N_m, \end{aligned} \tag{2.13}$$

where L is the maximum number of meters that the attacker can compromise (so that they can alter the measurement of the reported meter), and $[a_i^l, a_i^u]$ provides the lower and upper limits of modification to the measurement of each compromised meter, and $[z_i^{min}, z_i^{max}]$ denotes the valid range for each measurement, ensuring that the manipulated measurement is still within the permitted range on that particular unit. The strength of the measurement modification/manipulation depends on the attacker's resource and meter accessibility constraints, which have not been considered in previous work. In our work, by limiting the measurement manipulation to a subset of meters, the attacker can avoid injecting excessive errors, which can easily be detected by a univariate analysis. In addition, if the adversary knows where the high precision meters are located, they can avoid injecting too much error into those meters and instead allocate the resource to other meters to improve the overall attack outcome.

The objective function in the optimization Eq.(2.13) requires some distance metric $\|\cdot\|_p$ to quantify the impact of the attack. In this work, we evaluate the ANN-based state estimation by examining whether the state estimation is misled by an injection vector whose values are limited to a noise level. The injection is tiny itself, and its impact will be further cracked by the non-linearity of the AC power model. Therefore, this distance metric must be chosen carefully. In reality, the voltage magnitude is always limited in a tight range in order to ensure stable electricity supply, whereas the voltage phase angle varies in a relatively large range. Hence, an erroneous estimation of the latter may seriously affect the consistent operation of the power grid, but cannot be easily detected. Therefore, instead of targeting the total difference contributed by both voltage magnitudes and the voltage phase angles, we define the adversary's objective function as the maximum change to the voltage phase angles θ:

$$\|\hat{\mathbf{x}}_{\mathbf{a}} - \hat{\mathbf{x}}\|_\infty = \max(|\hat{\theta}_{a_1} - \hat{\theta}_1|, ..., |\hat{\theta}_{a_n} - \hat{\theta}_n|). \tag{2.14}$$

2.6 Attack Methodology

In this section, we present two algorithms, DE and SLSQP, to solve the proposed optimization Eq.(2.13).

2.6.1 Solving the Proposed Attack with DE

As a population-based stochastic optimization algorithm, the DE algorithm was first proposed in 1996 by Rainer *et al.* [106]. The population is randomly initialized within the variable bounds. The main optimization process consists of three operations: mutation, crossover, and selection. In each generation, a mutant vector is produced by adding a target vector (father) with a weighted difference of the other two randomly chosen vectors. Then a crossover parameter mixes the father and mutant vectors to form a candidate solution (child). A pair-wise comparison is drawn between fathers and children, and whichever is better will enter the next generation.

We follow [107] to encode our measurement attack vector into an array, which contains a fixed number of perturbations, and each perturbation contains two values: the compromised meter index and the amount into inject to that meter.

The use of DE and encoding has the following three advantages for generating attack vectors.

- **Higher probability of finding global optimum** - In every generation, the diversity introduced by the mutation and crossover operations ensures that the solution does not get stuck in a local optimum, leading to a higher probability of finding the global optimum [107, 106].

- **Adaptability for multiple attack scenarios** - DE can adapt to different attack scenarios using our encoding method. On the one hand, by specifying the number of meters to compromise, DE can search for both meter indices and injection amount. On the other hand, by fixing the meter indices, DE can only search for injection amount to these specified meters.

- **Parallelizibility to shorten attack time** - The function evaluation of an ANN is computationally demanding. As the scale of the smart grid increases, generating an attack vector may take seconds to minutes. An attacker must complete the generation and injection of the attack vector before the next state estimation takes place. DE algorithm is parallelization-friendly, as it is based on a vector population. DE operations can be mounted on a computer cluster to significantly expedite the computation of the attack vector.

Next, we present how we adapt the DE algorithm to our proposed attack:

- **Deal with duplicate meter indices** - In our work, instead of outputting the exact meter value, we select to output the injection vector to shrink the search space. We use two approaches to ensure the uniqueness of meter indices in the solution. First, we generate meter indices without replacement in population initialization. Second, we add a filter in the crossover operation. This filter keeps the meter indices unchanged if the newly selected meter index is repetitive.

- **Ensure the measurement after injection is within range** - A valid measurement reading must satisfy $z_i^{\min} \leq z_i + a_i \leq z_i^{\max}$, where $z_i^{\min}$ and $z_i^{\max}$ are the lower and upper limit power permitted on z_i. We use an intuitive approach by replacing $\mathbf{z_a} = \mathbf{z} + \mathbf{a}$ with $\mathbf{z_a} = \min(\max(\mathbf{z_a}, \mathbf{z}^{\min}), \mathbf{z}^{\max})$, where $\min$ and $\max$ are element-wise operations.

- **Deal with the overall constraint** - In addressing the constraints, adding a penalty term to the original objective function has been one of the popular approaches. However, they do not always yield satisfactory solutions since the appropriate multiplier for the penalty term is difficult to choose and the objective function may be distorted by the penalty term. Therefore, we use a heuristic constraint handling method proposed in [27]. A pair-wise comparison is drawn between fathers and children in order to differentiate feasible solutions from infeasible ones. The three criteria of the pairwise comparison are the following:

  1. If both vectors are feasible, the one with the best objective function value is preferred.
  2. If one vector is feasible and the other one is not, the feasible one is preferred.
  3. If both vectors are infeasible, the one with the smaller constraint violation is preferred.

  Essentially, the above comparison handles constraints in two steps: first, the comparison among feasible and infeasible solutions provides a search direction towards the feasible region; then, the crossover and mutation operations keep the search near the global optimum, while maintaining the diversity among feasible solutions. The pseudo code for the proposed attack using DE is presented in Algorithm 1.

2.6.2 Solving the Proposed Attack with SLSQP

In some gradient-based attack algorithms in image classification([109, 17]), a logistic function is added to the objective function as a penalty term and the multiplier for the penalty term is chosen by a line search. These algorithms aim to find a feasible solution, not the optimal one.

Algorithm 1 DE attack

Input: measurement **z**, GEN_{MAX} {maximum number of generations}, N {population size}, f {objective function}, g {constraint function}, CR {crossover rate}
Output: injection vector **a**

$g = 0$
Population initialization $\mathbf{a}_{i,0}$ for $i = 1, ..., N$. Meter indices are randomly select without replacement and injection amounts are randomly select within the univariate bound.
Evaluate the $f(\mathbf{a}_{i,g})$ and constraint violation $CV(\mathbf{a}_{i,g}) = \max(g(\mathbf{a}_{i,g}), 0)$, for $i = 1, ..., N$
for $g = 1 : MAX_{GEN}$ **do**
  for $i = 1 : N$ **do**
    Randomly select r_1 and r_2
    $j_{rand} = randint(1, N_m)$
    for $j = 1 : D$ **do**
      if $(rand_j[0, 1) < CR$ **or** $j = j_{rand})$ **and** the meter index not repetitive with previous meter indices **then**
        $u^j_{i,g+1} = x^j_{best,G} + F(x^j_{r_1,g} - x^j_{r_2,g})$
      else
        $u^j_{i,g+1} = x^j_{i,G}$
      end if
    end for
    Evaluate $f(\mathbf{u}_{i,g+1})$ and $CV(\mathbf{u}_{i,g+1})$
    Update the population if the child $\mathbf{u}_{i,g+1}$ is better than the father $\mathbf{x}_{i,g}$ by above three criteria
  end for
end for

Therefore, we use a conventional optimization algorithm (SLSQP) [56]. SLSQP is a variation on the SQP algorithm for non-linearly constrained gradient-based optimization. In our SLSQP attack, we encode the solution to a N_m-dimension vector, in which the i-th element denotes the injection amount to the i-th meter. This encoding allows the attacker to generate attack vectors for a set of specified meters by placing upper and lower bounds on the corresponding elements in the attack vector. To solve the proposed optimization problem, we first construct the Lagrangian function.

$$\mathcal{L}(\mathbf{a}, \lambda) = f(\mathbf{a}) + \lambda \cdot g(\mathbf{a}), \tag{2.15}$$

where

$$\begin{cases} f(\mathbf{a}) = \|\hat{\mathbf{x}}_{\mathbf{a}} - \hat{\mathbf{x}}\|_\infty \\ g(\mathbf{a}) = (\mathbf{z} - h(\hat{\mathbf{x}}_{\mathbf{a}}))^T \mathbf{W} (\mathbf{z}_{\mathbf{a}} - h(\hat{\mathbf{x}}_{\mathbf{a}})) < \tau. \end{cases} \tag{2.16}$$

In each iteration k, the above problem can be solved by transferring to a linear least square sub-problem in the following form:

$$\begin{aligned} \max_{\mathbf{d}} \quad & \|(\mathbf{D^k})^{1/2}(\mathbf{L^k})^T\mathbf{d} + ((\mathbf{D^k})^{-1/2}(\mathbf{L^k})^{-1}\nabla(\mathbf{a}^k)\| \\ \text{subject to} \quad & \nabla g(\mathbf{a}^k)\mathbf{d} + g(\mathbf{a}^k) \geq 0, \end{aligned} \tag{2.17}$$

where $L^k D^k (L^k)^T$ is a stable factorization of the chosen search direction $\nabla^2_{zz}\mathbf{L}(\mathbf{z}, \lambda)$ and is updated by BFGS method.

By solving the QP sub-problem for each iteration, we can get the value of $\mathbf{d}^k$, i.e., the update direction for $\mathbf{z}^k$:

$$\mathbf{z}^{k+1} = \mathbf{z}^k + \alpha\mathbf{d}^k, \tag{2.18}$$

where α is the step size, which is determined by solving an additional optimization. The step size $\psi(\alpha) := \phi(\mathbf{a}^k + \alpha d^k)$ with $\mathbf{x}^k$ and d^k are fixed, and can be obtained by minimization:

$$\phi(\mathbf{a}^k; r) := f(\mathbf{a^k}) + \max(r \cdot g(\mathbf{a}), 0), \tag{2.19}$$

with r being updated by:

$$r^{k+1} := \max(\frac{1}{2}(r^k + |\lambda|, |\lambda|)). \tag{2.20}$$

The limit on the injection amount is achieved by setting a bound to the optimizing variable. The physical constraint for branch limit is ensured by performing an element-wise min-max operation as in the DE attack.

2.7 Attack Evaluation

In this section, we evaluate both FDI attacks on three IEEE test systems: 9-bus, 14-bus, and 30-bus systems. The implementation of our attacks is done in Python, using packages *TensorFlow* and *SciPy*. We run the experiments on a computer equipped with a 3.5 GHz CPU and 16 GB memory.

Attack Scenarios: Depending on the attacker's capabilities and practical constraints, the attacker can launch an attack in different scenarios. Inspired by [74], we consider the following two attack scenarios to facilitate evaluation.

- **Any k-meter attack** - The attacker can access all meters, but the number of compromised meters is limited by k. In this scenario, the attacker may want to wisely allocate the resource, by selecting meters and injection amounts that maximize the impact of the attack.

- **Specific k-meter attack** - The attacker has the access to k specific meters. For example, the attacker may access only meters in a confined region. In this case, the attacker needs to determine the injection amount to each meter to maximize the attack impact.

We perform the experiments as follows. To fairly compare the attack performance on different test systems, we choose the percentage of compromised meters, R, to be 5%, 10% and 20%. For each R, we explore the attack performance under different injection levels: 2%, 5% and 10%. The injection level is defined as the maximum injection strength in terms of proportion to the measurement. Each experiment runs on 1,000 measurement instances and is repeated for 10 times to reduce randomness.

We consider the following four metrics to evaluate the effectiveness of attacks. We measure the MAE and MARE that are injected into the voltage phase angle. We also report the success rate, where success is defined as an attack that produces more than 5% MARE to the voltage phase angle. Moreover, since the smart grid is assumed to be a quasi-static system and the state changes slowly over time, we want to investigate if the time between two state estimations allows an adversary to mount the FDI attack on the smart grid.

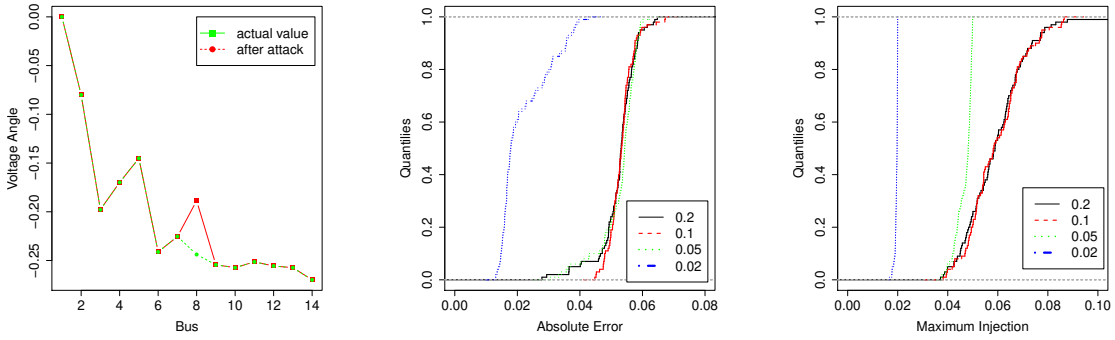
2.7.1 Any k Meter Attack

Under this scenario, the attacker can access all meters and has the freedom to choose any k meters to compromise. The way we encode the attack vector in DE enables the search for better target meters in every generation. In contrast, SLSQP only allows us to put constraints on specific meter indices. Therefore, only DE can be used to find the attack vector in any

k-meter attack. $DE/x/y/z$ denotes a DE variant, in which x specifies that the vector to be mutated is chosen by "random" or "best", and y denotes the number of difference vectors and z denotes the crossover scheme. We implement three DE variants in our experiments: *DE/best/1/bin*, *DE/current to best/1/bin* and *DE/current to rand/1/bin*, where *bin* denotes the binomial. These DE variants differ in the way of how the father vector is selected and how the differential variation is formed. We find that there are no significant differences among them. Hence, $DE/best/1/bin$ is used in all experiments:

$$u_{i,G+1} = x_{best,G} + F(x_{r_1,G} - x_{r_2,G}),$$

where $x_{r_1,G}, x_{r_2,G}$ are integers drawn from the current population, and $x_{best,G}$ denotes the best individual in terms of the value of the objective function in the current population. F is a real and constant factor $\in [0.5, 1]$, which controls the intensity of the mutant.



(a) Voltage phase angle before and after the attack.

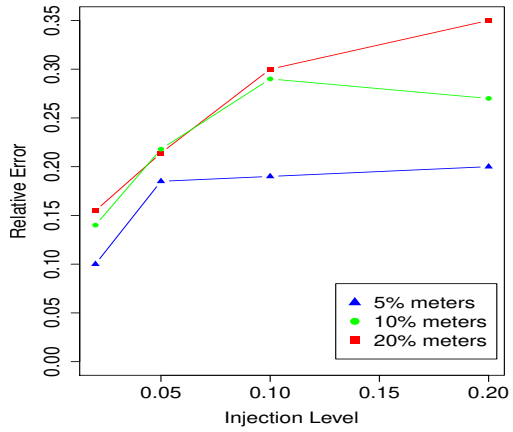
(b) CDF of absolute error in voltage phase angle.

(c) CDF of maximum injection.

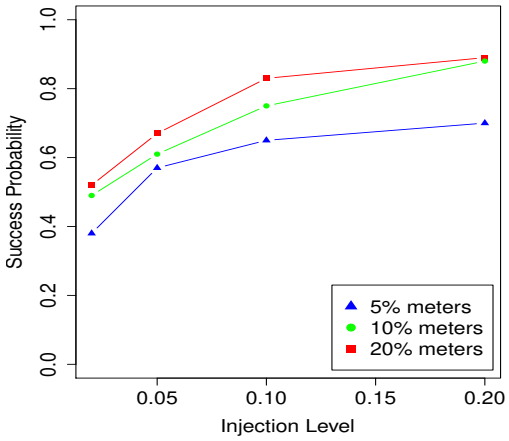
Figure 2.1: An example of a 5-meter attack on the 14-bus system.

Figure 2.1 shows an example of a 5%-meter attack on the 14-bus system. Our DE attack injects error into one of voltage phase angles, while others remain unchanged. In Figures 2.1 (b) and (c), for injection levels 10% and 20%, the maximum injections are condensed at 5% and rarely exceed 10%, due to the overall constraint on bad data detection.

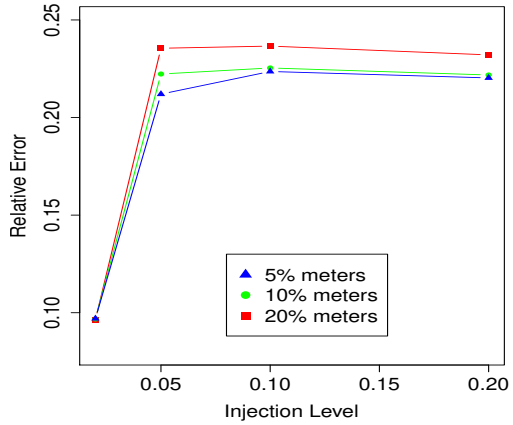
Figure 2.2 shows the impact of the attack with the change of R and the injection level. In general, the success probability and attack impact increase as the attacker controls more



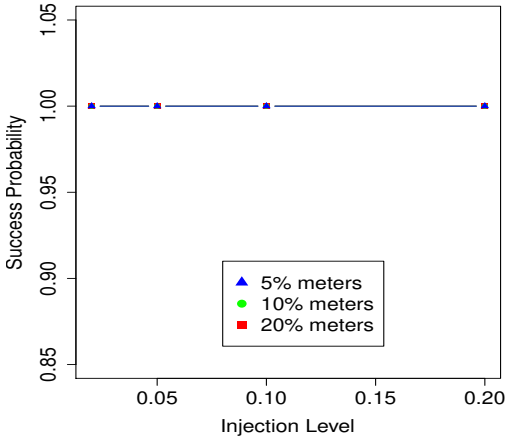
(a) 9-bus relative error.



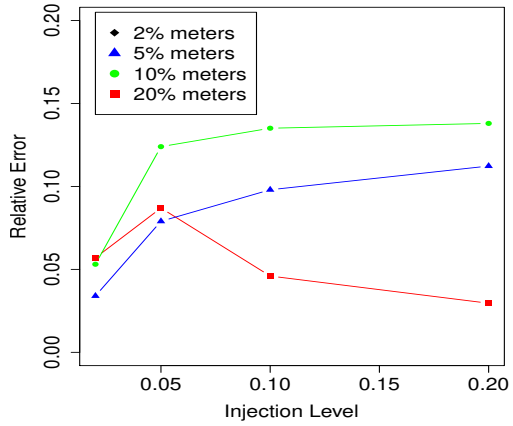
(b) 9-bus success rate.



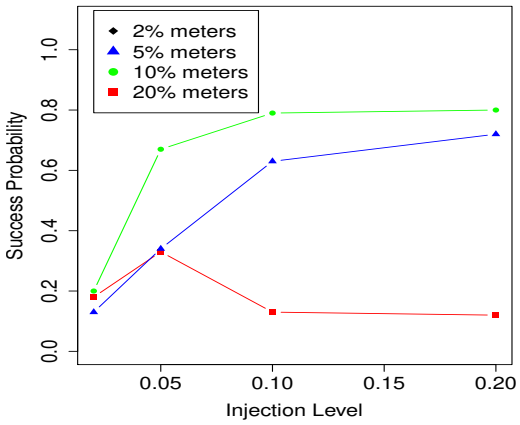
(c) 14-bus relative error.



(d) 14-bus success rate.



(e) 30-bus relative error.

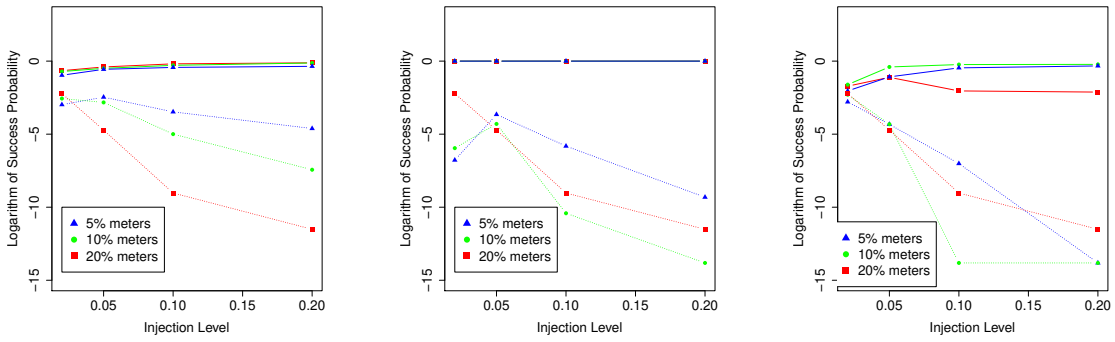


(f) 30-bus success rate.

Figure 2.2: Relative error and success rate of any k-meter attack on 3 test systems with $N = 400$ and $G_{MAX} = 400$.

resources. The attacker achieves a high success rate (80% of simulation instances) by compromising 10% of meters with injection level 10%. Especially for the 14-bus system, the attack achieves 100% success for any combination of R and the injection level.

Interestingly, for the 30-bus system, the impact of 10% compromised meters exceeds that of 20% compromised meters. In addition, the performance of the 20% of compromised meters drops drastically as the injection level increases. A possible explanation for this is that, with the expansion of search dimension and space, DE requires more generations to find a satisfactory solution.

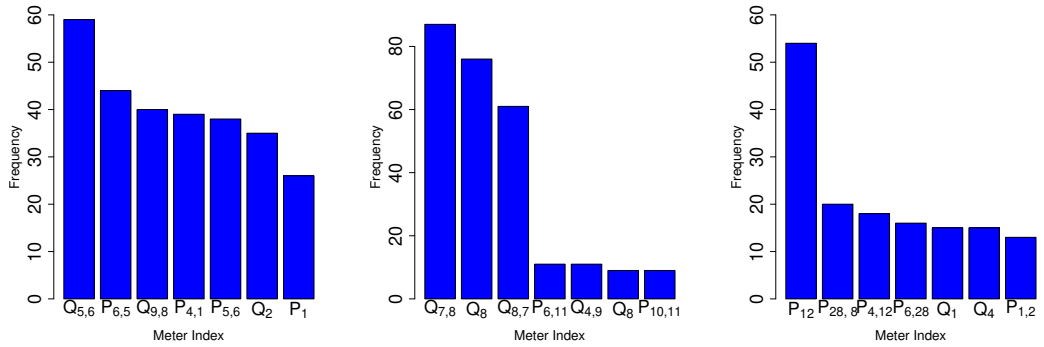


(a) 9-bus. (b) 14-bus. (c) 30-bus.

Figure 2.3: Success rate of the DE attack and the random attack on a log scale. Solid lines refer to the DE attack, and dashed lines refer to the random attack.

We compare our proposed attack with a random attack, where the injection vectors are generated from a uniform distribution. The success probability is reported on the same set of instances with $1,000$ attempts on each instance. The success rate is compared to that of our DE attack on a logarithmic scale (Figure 2.3). There is no significant difference between the impact of the DE attack and that of the random attack when the injection level is low, in which the attack impact is limited. However, if the attacker wants to achieve greater impact, our DE attack outperforms the random attack by order of magnitude.

Figure 2.4 shows the frequency of the meter indices that present in the attack vectors. Because most of the meter frequencies are small, only the seven meters with the largest frequencies are presented. The injection into high-frequency meters can introduce large error to the state variable. Our DE attacks also help to identify vulnerable meters, in which people can



(a) 9-bus. (b) 14-bus. (c) 30-bus.

Figure 2.4: Frequency of meters selected in the attack vectors.

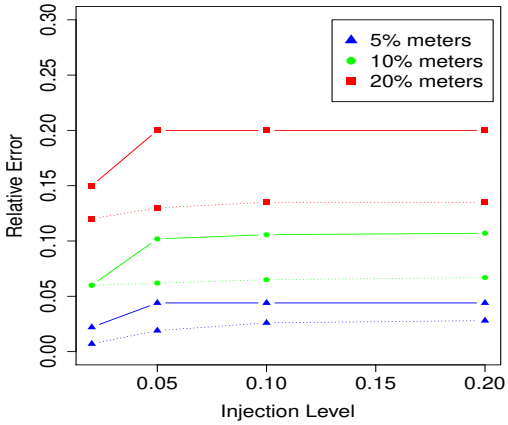
strengthen the physical protection, e.g., replace them with higher precision meters or lock them in boxes.

2.7.2 Specific k Meter Attack

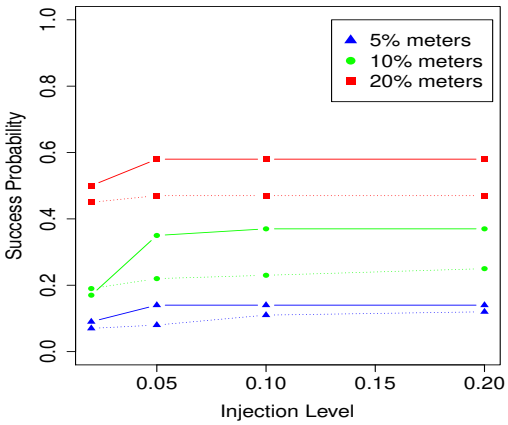
To explore the effect of the population size and iteration number, we evaluate the average *number of function evaluations* (NFEs) before delivering a successful attack or when no significant change in the solution is observed. In the DE case, NFE is equal to the population size multiplied by the number of generations. NFEs and the corresponding running time are shown in Table 2.5. For all combinations of systems and attack settings, the attacker can find a successful attack vector in 3 seconds or conclude that the attack is infeasible.

In this scenario, the attacker can compromise specific k meters due to restrictions in physical location. DE and SLSQP are implemented and compared in this attack scenario. To search for the injection amount in specific k meters, DE specifies the indices of the k meters in population initialization and disables the mutation operation of the meter index, while SLSQP only allows modifications to the k meters in the attack vector. We randomly select 3 sets of meters such that R is 5%, 10% and 20%, respectively. We perform the same set of experiments using both the DE and SLSQP algorithms and compare their performance using the same metrics.

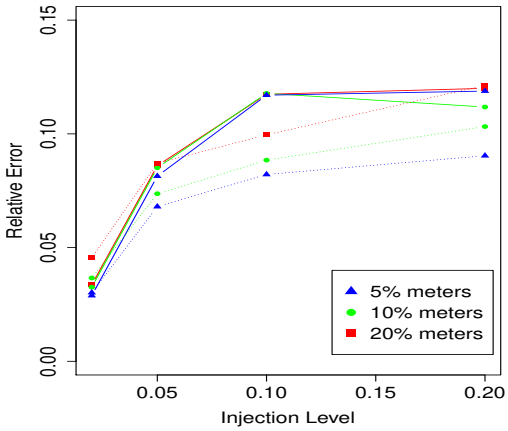
In general, the DE algorithm outperforms the SLSQP algorithm in effectiveness (Figure 2.5). This is not surprising, as the DE brings more diversity in every generation, whereas SLSQP only explores the neighbors in each iteration.



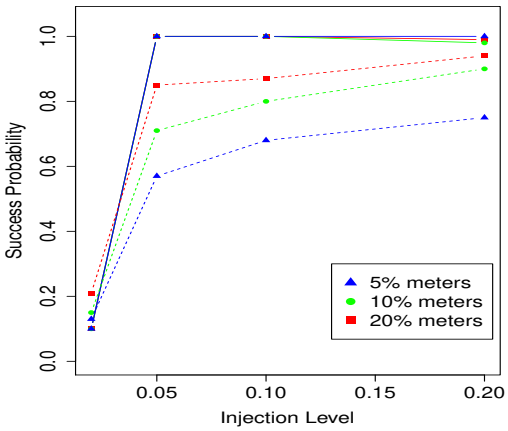
(a) 9-bus relative error.



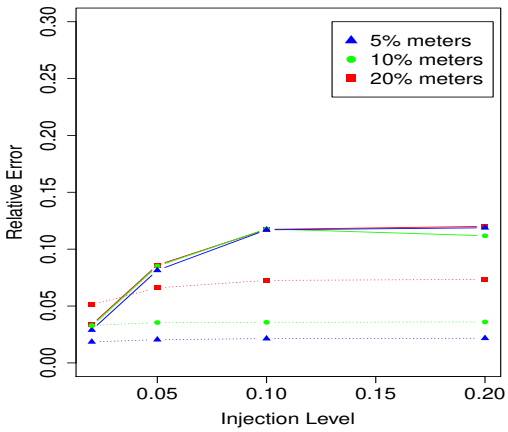
(b) 9-bus success rate.



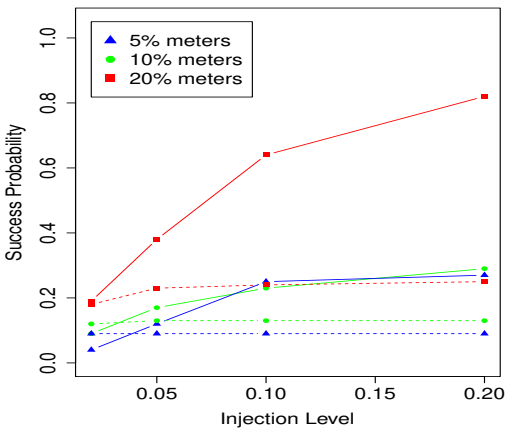
(c) 14-bus relative error.



(d) 14-bus success rate.



(e) 30-bus relative error.



(f) 30-bus success rate.

Figure 2.5: Relative error and success rate of the specific k-meter attack on 3 test systems.

Table 2.6 shows the convergence time of the DE attack with $10,000$ NFEs and the SLSQP attack with 100 iterations. Both attacks converge quickly within 3 seconds, which is feasible for an attacker to complete before the next state estimation takes place. A simple comparison of running time between them can be misleading, since the specific k meters involved in our test are chosen blindly. The convergence time is highly dependent on the meters chosen to perform the attack. The selection of vulnerable meters would greatly shorten the attack time. In addition, the execution time can be further shortened by applying an early-stop criteria or parallel processing to the DE attack, or adjusting the max iterations for the SLSQP. Therefore, without taking into account the running time, our experiments exhibit a clear pattern that the DE attack is more effective than the SLSQP attack.

Table 2.5: Average NFEs and execution time (in seconds) of any k-meter attack on 3 test systems.

Test System	**NFEs**	**Time (s)**
9-bus	500-1500	0.25-0.45
14-bus	500-3500	0.5-1.73
30-bus	800-5600	1.5-2.7

Table 2.6: Convergence time (in seconds) comparison of the specific k-meter attack on 3 test systems.

Test System	**DE (s)**	**SLSQP (s)**
9-bus	0.12-0.4	0.036-0.6
14-bus	0.06-0.6	0.14-1.0
30-bus	0.3-3.0	0.26-2.2

2.8 Potential Defenses

In this section, we are interested in how the proposed attacks behave when a defense mechanism is specifically customized / optimized for these attacks. Note that such a specialized defense mechanism is in sharp contrast to the general defense mechanisms considered in previous works, which do not assume/exploit any knowledge or feature of the proposed attacks. Putting the proposed attacks in the context of a strong and specialized defense mechanism allows us

to gain insights on the limit of both the attacker and the defender in a more realistic "sharpest-sword vs. strongest-shield" setting, as in practice "maximum effort" is commonly executed not only by attackers but also by defenders, especially when it comes to a mission-critical infrastructure such as the power grid. In the following, we first review existing state-of-the-art defense proposals against adversarial examples in image classification and explain why some of them are not applicable to our problems. Then, we propose an adversarial training-based defense mechanism to counter our proposed attacks. Several techniques are also developed to optimize the proposed defense. The performance of the proposed mechanism is evaluated by simulations in Section 2.7.

Despite the significant number of works on detection against the FDI attack, most of the existing detection mechanisms are mainly built on the DC state estimations or traditional WLS state estimators. These detection methods achieve high detection accuracy with a low false alarm rate, but they are not applicable to the ANN-based state estimator. The defense strategy against the FDI attack on the AC ANN-based state estimation has not been intensively studied.

In the image classification area, proactive countermeasures against adversarial examples aim to make the ANN model more robust before the attacker gets the chance to generate adversarial examples. Mainstream proactive countermeasures fall into three categories [126]: the defensive distillation, adversarial training, and classifier robustifying.

However, our problem has a different goal compared to image classification. Methods based on the probability of the target class, such as defensive distillation and classifier robustifying, are not applicable. To propose the defense, we need to address two challenges: (1) in contrast to an image classification problem, our goal is to minimize the error in the state space while keeping the residual in the measurement space below a pre-defined threshold; (2) measurements contaminated by a small injection level are well hidden as they are nearly from the same distribution as clean measurements. The defense should not be sensitive to adversarial injections, yet measurements with regular noise should not trigger bad data detection alarms.

As stated in [51], there are two main methods to strengthen a regression model: noise-resilient regression and adversarial training. The idea behind the noise-resilient regression is to enhance the model's tolerance to noise and identify and remove the outliers, while not

triggering bad data alarm or losing accuracy. In the target model training process in Section 2.4, we adopt the idea of noise resilience by adding noises sampled from a certain distribution to the training data, so that the model learns the distribution and is able to eliminate the effect of such noises. In addition, we minimize both errors in the state space and measurement space to improve the accuracy of ANN-based state estimation and narrow the left-over space for attacks. Although these methods provide a robustness improvement against noise and outliers, the results in Section 2.7 show that a noise-resilient model is not resistant to our attacks. It is suggested that an adversary can still generate noise-like injections to mislead the state estimate. It turns out that introducing noise to the measurements and minimizing the training error in both spaces does not make the model more robust to adversarial injections.

Among many defenses against adversarial examples, adversarial training [109, 38] has been one of the most effective methods [75, 59]. The adversarial training attempts to minimize the impact of injection in the model training phase, rather than trying to identify and mitigate them in the operational phase of the trained model. This is achieved by a min-max formulation:

$$\theta = \arg\min_{\theta} E_{(x,y)\sim D}\left[\max_{\delta\in S} L(\theta, x+\delta, y)\right], \tag{2.21}$$

where D is the set of training data, L is the loss function, θ is the parameter of the network, and S is a norm-constrained ball centered at 0. In contrast to regular training, adversarial training uses min-max optimization, where *inner maximization* produces injection data based on the current model and injects them into the training data set, while *outer minimization* minimizes the state estimation deviation on the enlarged training data set, in which the injection data are included.

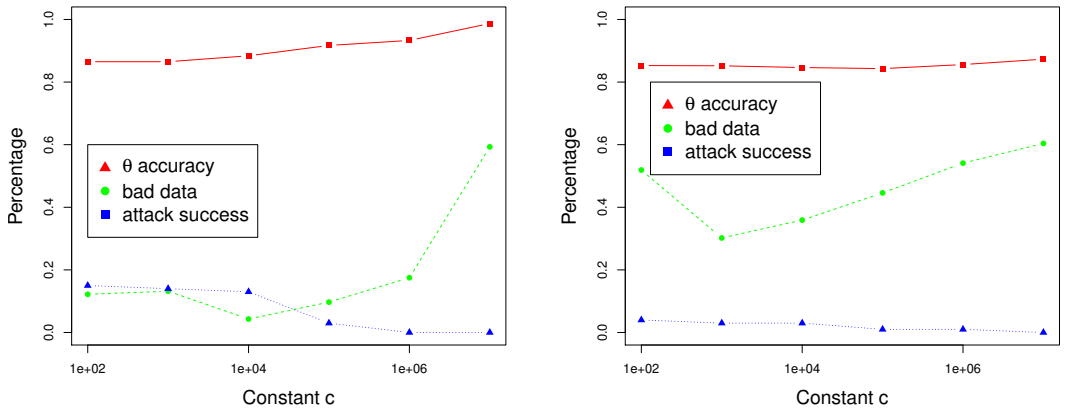
Inspired by [75] and considering the uniqueness of our problem, we propose a defense through an optimization perspective with the goal of improving robustness while maintaining the accuracy of the ANN-based state estimation model:

$$\theta = \arg\min_{\theta} c \cdot E_{(\mathbf{x},\mathbf{z})\sim D}\left[\max_{\delta} \|\tilde{\mathbf{x}} - \mathbf{x}\|\right] + loss(\mathbf{z}, \mathbf{x}), \tag{2.22}$$

where $c > 0$ should be well chosen to balance the optimization strength on each term. Compared to Eq. (2.21), a training loss term is added to the optimization to take into account the accuracy of the model.

In the process of choosing a suitable c, since the value of the first term is very small, a large c would make optimization emphasize minimizing the risk of the FDI attack, while a small c would cause a high false alarm rate. Empirically, we find that the best way to choose c is to balance the accuracy of the model, the bad data rate, and the robustness of the model. We verify this by running the adversarial training model for values of c spaced uniformly (on a log scale) from $c = 1 \times 10^2$ to $c = 1 \times 10^7$, on the 9-bus system customized for the 10%-meter specific DE and SLSQP attack, respectively. The model accuracy and bad data rate are evaluated on the test data set, while the effectiveness of adversarial training is evaluated by DE and SLSQP attacks. We plot the voltage angle accuracy, bad data rate, and attack success rate as a function of c in Figure 2.6. We find that both attacks show similar patterns. As c increases, attacks become rarely successful at the cost of the state estimation model being more conservative. The conservativeness is mainly reflected by the model recognizing a growing number of measurements with regular noises as bad data. In practical state estimation applications, bad measurements are usually discarded and will not be used to estimate the current system status. Therefore, a high false alarm rate would increase the risk of unobservability of the system. Although the adversarial trained state estimation model could identify more data as bad data, this is a minor model degradation, which can be manually resolved, for example, by increasing the sampling rate.

As claimed in [75], solving optimization alone is not a sufficient condition for the accuracy and robustness of the model. In addition, it requires both the optimization and the value of the objective function to be small. This is because, in general, a smaller objective value implies a better model. However, in our problem, this is not always true. Due to the presence of noise, a lower objective value does not always indicate a better model. Furthermore, obsessively pursuing a small objective value may lead to overfitting. Therefore, we stop the training process when we observe that the loss is consistently smaller than the threshold.



(a) SLSQP. (b) DE.

Figure 2.6: Sensitivity to the constant c.

We then use *Adam optimizer* to adversarially train state estimation models on the 9-bus, 14-bus and 30-bus systems with the same attack settings and meter indices as in Section 2.7.2. According to our results, the three systems present similar patterns. To evaluate the effectiveness of adversarial training on all test systems, we present the experiment results of the adversarial training for the 10%-meter specific attack with the injection level of 10% in Table 2.7, in terms of voltage angle accuracy, bad data rate and attack success rate. While adversarial training significantly reduces the attack success rate, it achieves this benefit at the cost of an elevated bad data rate and a slight degradation (several percent) in model accuracy, for defenses against both the DE and SLSQP attacks.

The reason for the slightly degraded accuracy is that the adversarial training is done on an enlarged training data set, in which the adversarial data are generated and added to the data set as the training process goes on. At the individual level, an adversarial example may hide well in the actual data distribution. However, if the whole population is examined, the adversarial data distribution may differ slightly from the actual data distribution. Therefore, the model learned from the adversarial data may shift accordingly, causing a slightly lower accuracy.

It is also noted that adversarial training with examples generated by DE has a higher bad data rate than training with examples generated by SLSQP. One possible explanation is the high skewness in the residual distribution. In the process of generating adversarial examples, while the SLSQP finds adversarial examples in the neighbors, DE, being a stochastic method, always

probes more possibilities to make use of the resource. Taking a closer look at the residuals of the adversarial data, we can notice that the residual distribution is highly left skewed and is highly condensed at the value of the bad data detection threshold. Due to the skewness, it takes more adversarial training iterations to converge, yet to a value just below the threshold. Such an unsteady convergence is susceptible to distribution difference, therefore, data from the true distribution are very likely to violate the bad data threshold, resulting in an elevated bad data rate. Note that such a drawback is not critical to the state estimation, as it can be easily overcome by proportionally raising the sampling rate to compensate for those good data lost due to the false alarm.

In summary, our proposed adversarial training works well in significantly reducing the attack success rate, but only at the cost of a higher bad data rate and a slight degradation of the model accuracy.

Table 2.7: Performance of adversarial training against specific attack of 10% meters with injection level of 10%.

Without Adversarial Training

	θ **Accuracy(%)**	**Bad Data(%)**	**SLSQP(%)**	**DE(%)**
9-bus	96	0	22	35
14-bus	99	3	71	100
30-bus	98	5	13	17

SLSQP	**With Adversarial Training**		
	θ **Accuracy(%)**	**Bad Data(%)**	**Attack Success**($\%$)
9-bus	91.7	9.7	3
14-bus	93.3	13.5	5
30-bus	92.4	14.9	2

DE	**With Adversarial Training**		
	θ **Accuracy(%)**	**Bad Data(%)**	**Attack Success**($\%$)
9-bus	85.2	30.2	3
14-bus	86.8	34.5	7
30-bus	80.3	40.2	1

2.9 Conclusions

In this work, we performed the first study of the adversarial FDI attack against the ANN-based AC state estimation. We first created target models that are sufficiently strong. Then we formulated the adversarial FDI attack into an optimization problem, followed by extensive evaluations under two attack scenarios on IEEE 9-bus, 14-bus and 30-bus test systems, based on the adaption of DE and SLSQP algorithms aiming to find attack vectors. In any k-meter attack, our results showed that the DE attack achieves a high success rate (more than 80% in all simulated cases), despite having a small number of compromised meters and low false injection strength. The DE outperforms SLSQP in the specific k-meter attack. Our findings also showed the potential of adversarial training in defending against these attacks, and such an approach can be further explored to improve the ANN-based AC state estimation model robustness.

Chapter 3

Assisting Backdoor Federated Learning with Whole Population Knowledge Alignment

3.1 Introduction

Federated learning (FL) [55, 77] is a distributed learning system, which allows multiple clients to collaboratively train a high accuracy model by taking advantage of a wide range of data from physically separated clients without sharing their locally collected data. Currently, FL applications thrive in next-word and emoji prediction on smartphones [22, 123, 97, 41], environmental monitoring [40], and aiding in medical diagnosis among hospitals [122, 15].

Due to the distributed nature and inherent data heterogeneity (i.e., data being non-i.i.d.) across FL clients, the local model updates uploaded by clients may be different from others. As a result, it is assumed that the central server cannot validate the legitimacy of received model updates, which provides a venue for new attacks. Backdoor is one of the data poisoning attacks [9], in which an adversary corrupts the global model such that the new global model reaches a high accuracy on the FL main task, as well as on a backdoor subtask activated by some trigger, and such a high backdoor subtask accuracy remains for multiple training rounds. Backdoor attacks have been shown to be unavoidable and computationally difficult to detect [113].

Although backdoor FL attacks are powerful, they have stringent requirements on the timing of attack, which are often difficult to meet in practice. To make our argument more concrete, in this work, we will focus on *single-shot* backdoor attacks [9], due to their benefits of stealthiness, simplicity in implementation, and the fact that the more general multi-shot backdoor attacks can be built upon them incrementally. Ideally, a single-shot backdoor attack, in which the adversary injects the designated backdoor trigger only once (so as to keep the attack stealthy), can achieve its goal with high accuracy by injecting the backdoor subtask when the

FL model is close to convergence. However, in practice, the attacker cannot always have the luxury of controlling injection time because clients in each FL training round are randomly selected. In fact, a backdoor subtask injected in the early stage of the training (before the FL model converges) can only generate very weak backdoor effects due to the following two reasons. (1) The strength of the injected backdoor model update will be severely diluted by the local model updates from other clients in the same round after the aggregation at the server, because the magnitude of the other clients' local model updates is significant when the global model is far from convergence. (2) The backdoor effect of the injected subtask vanishes quickly in subsequent training rounds as the injected backdoor will be overwritten by newcoming normal model updates in those rounds. As a result, the earlier the backdoor is injected, the faster the backdoor effect will diminish. In addition, the early injected backdoor is less stealthy as the main-task accuracy might deteriorate due to the dilution effect and the scaling operation to ensure the backdoor survives the aggregation at the server.

Realizing the stringent attack timing restriction of existing single-shot backdoor attacks, in this work, we are interested in studying a new single-shot backdoor attack technique that allows the backdoor subtask to be injected in the early stage of FL training while still achieving a strong and sustaining backdoor effect, making the effect of the attack less dependent on the timing of the attack, and hence making the attack more practical and applicable to general applications. Our new attack technique is inspired by the latest research findings on FL privacy, which demonstrate that although private client data is not directly revealed in FL, the shared FL global model can unintentionally leak sensitive information about the data on which it was trained [28, 34, 46, 79, 86]. This finding has motivated us to consider the following research problem: **does FL information leakage render a stronger backdoor attack in the early stage of FL training?** Our main insight is that the slow and unstable convergence of the global model in FL is mainly caused by the weight divergence [122] of the local model updates of different clients. This weight divergence is mainly decided by the difference in the label distribution (henceforth referred to as the "distribution") and the difference in gradients between a single client's local data and the whole population's data (i.e., the aggregation of all clients' data). Therefore, reducing these differences will shrink the weight divergence and henceforth

expedite FL convergence. This will increase the strength and sustainability of an early-stage backdoor subtask injection.

In this work, we propose a novel information leakage assisted two-phase FL backdoor attack, which enhances the effectiveness of FL early injected single-shot backdoor attack. We assume that attacker-controlled clients can interact with FL training multiple times, but they only have one chance to inject the backdoor. In our design, we do not directly strengthen the backdoor attack. Instead, we design a preliminary phase for the subsequent backdoor injection, where the attacker-controlled clients play the role of accomplices and reach out the FL global model by uploading model updates that are beneficial to FL convergence, to pave the way for the subsequent backdoor injection. Formally, the proposed backdoor attack consists of two phases: a preliminary phase, in which the attacker-controlled clients help to accelerate the convergence of the FL model, and an attack phase, in which the backdoor attack is launched. In the preliminary phase, attacker-controlled clients first perform a passive inference attack to get an estimate of the whole population distribution. Then, instead of training on the original local data, they train on locally crafted datasets whose distributions are aligned with the inferred whole population distribution, so that the weight divergence is reduced, and the FL model converges more quickly. Although the operations in the preliminary phase seem legitimate, they help to improve the effectiveness and persistence of the backdoor by reducing the dilution effect from other clients (as the magnitude of their local model updates decreases more quickly).

When the expected FL model accuracy is reached or the client that has the capability to perform a backdoor attack is selected, the backdoor attack is launched by training on a locally poisoned dataset and the backdoored local model updates are scaled up before submitting to the FL server. Benefiting from the preliminary phase, the single-shot backdoor injected into the resulting FL model will be less likely to be diluted by model updates from other clients. Therefore, the designed preliminary phase successfully overcomes the deficiencies of early injected single-shot backdoor and significantly improves the strength and persistence of the backdoor effect. Note that the proposed preliminary phase benefits the backdoor effectiveness by improving the FL convergence, and hence reduces the dilution effect from other clients.

And this preliminary phase is independent of the attack phase, therefore, can be combined with any kind of backdoor attacks to enhance their backdoor performance.

To the best of our knowledge, we are the first in the literature to enhance the effectiveness of FL backdoor attacks by utilizing the information leaked from the FL model. Our **contributions** in this work are fourfold:

- We prove an upper bound for the intra-aggregation weight divergence between the FL model and the centralized learning (CL) model and demonstrate that the weight divergence is small. Thus, FL global model updates can be used to approximate CL model updates.

- We propose a novel optimization-based whole population distribution inference attack utilizing the above approximation and the linearity of the cross-entropy. Unlike the existing property inference attack, in which it can only generate binary property inference results, our proposed inference attack produces precise quantitative property information about the dataset.

- We propose a preliminary phase for the early injected single-shot backdoor attack, which improves the attack effectiveness by reducing the dilution effect from local updates of normal clients. Specifically, attacker-controlled clients use the inferred distribution to craft auxiliary datasets using augmentation and downsampling techniques so that the distribution of the auxiliary dataset is aligned with both the gradient and the inferred global distribution. Training on the auxiliary dataset can facilitate the convergence of the FL model and reduce the magnitude of local model updates from normal clients, and further boost the performance of the backdoor attack.

- Extensive experiments are conducted on the MNIST dataset under various data heterogeneity settings to evaluate the accuracy of the proposed whole population distribution inference attack, the improvement of the convergence of the FL global model brought about by the proposed preliminary phase, and the effectiveness of the proposed backdoor attack. We also evaluate the proposed attack against two state-of-the-art defense mechanisms. The experimental results show that the proposed inference attack achieves high

accuracy against FL in scenarios with and without defense mechanisms. The FL model assisted by the preliminary phase has a faster convergence rate, especially in the early training stage. The proposed backdoor outperforms existing backdoor attacks both in success rate and longevity, even when defense mechanisms are in place.

The remainder of this chapter is structured as follows. We start by providing the background and related work in Section 3.2. We present the threat model and the attack design philosophy in Section 3.3. Subsequently, the overview and detailed attack steps are presented in Section 3.4. Finally, the experimental setup and results are presented in Sections 3.5 and 3.6, respectively. We evaluate the robustness of the proposed backdoor attack against two defense mechanisms in Section 3.7, and we conclude our work in Section 3.8.

Throughout this work, we use the following notation:

- $\|\cdot\|$ denotes the ℓ_2 norm.
- D_k and D denote the training data on the k-th client and the entire training data population, respectively. And we have $D = \cup_{k=1}^{N} D_k$.
- n_k and n denote the number of training samples in D_k and D, respectively. And we have $n = \sum_{k=1}^{K} n_k$.
- w_k^T and w^T denote the k-th local model weight and the global model weight in the T-th aggregation, respectively.
- $F_k(w_k; D_k)$ and $F(w; D)$ denote the loss function on the k-th client and the loss function of a CL model, respectively.
- $\nabla L(w_k; D_k)$ and $\nabla L(w; D)$ denote the loss gradients of the client k and the loss gradients of the CL model, respectively.
- $p(y = c)$ is the proportion of the label c in the training data, and we have $\sum_{c=1}^{C} p(y = c) = 1$.

3.2 Background and Related Work

3.2.1 Federated Learning

The whole population $D = \cup_{k=1}^{N} D_k$ is allocated to N clients and each client maintains D_k. Each client maintains a local model trained from the local training dataset. And a central server maintains a global model by aggregating the local model updates from the participating client in each training round. The objective of FL training is to minimize the loss:

$$F(w) = \frac{1}{|D|} \sum_{(x,y)\in D} L(w; (x, y)). \tag{3.1}$$

To achieve this goal, each client k optimizes their local model weights w_k to minimize the loss function $F_k(w) = \frac{1}{|D_k|} \sum_{(x,y)\in D_k} L(w; (x, y))$. Here, we describe the FedAvg aggregation method [77], which is perhaps the most widely used averaging scheme. FedAvg iteratively performs the following three steps:

(1) Global model synchronization. In the T-th aggregation, the central server randomly selects K $(K \leq N)$ from N clients and broadcasts the latest global model w^T to the selected clients: $w_k^{T,0} \leftarrow w^T$.

(2) Local model training. Each client k updates its own local model w_k^T by running an SGD on the local dataset D_k for t steps. The τ-th step on client k follows:

$$w_k^{T,\tau+1} \leftarrow w_k^{T,\tau} - \eta \nabla F_k(w^{T,\tau}), \tag{3.2}$$

where η is the local learning rate.

(3) Global model update. After performing local training for t steps, the client transmits the model update $\Delta w_k^T = w_k^{T,t} - w_k^{T,0}$ back to the central server. The central server then updates the global model by performing a weighted average on the local model updates sent from K clients:

$$w^{T+1} \leftarrow w^T + \sum_{k=1}^{K} \frac{n_k}{n} \Delta w_k^T, \tag{3.3}$$

where $n_k = |D_k|$ is the number of training data on the client k and $n = \sum_{k=1}^{K} n_k$ is the total number of training data used on the selected clients.

3.2.2 Information Leakage in FL

We mainly discuss the literature related to property inference attack. The property inference attack was first proposed by Ateniese et al. [7] against Hidden Markov Models and Support Vector Machines. The authors in [36] designed a property inference attack on fully connected networks, in which the adversary trains a meta-classifier to classify the target classifier depending on whether it possesses the property of interest or not. A malicious user can infer attributes that characterize the entire data class or a subset of data [79].

We also note that our whole population distribution inference attack is similar to that of [115], where the authors analyzed the relationship between the number of data samples of a specific label and the magnitude of the corresponding gradients. Our work differs from their work from the following two perspectives: (1) their work draws a comparison between a pair of labels and generates a binary output of which label possesses a larger number of data samples, while our work is able to provide a precise quantitative distribution of all labels; (2) to get a satisfying inference result of the whole distribution, their work has a high computation complexity and needs to be performed multiple rounds, while in our work the distribution can be inferred in one training round and requires less computation.

3.2.3 Backdoor Attacks against FL

The backdoor attack is one of the data poisoning attacks whose goal is to misclassify inputs with backdoor triggers as the target class, while not affecting the model accuracy on clean data. The backdoor attack was first introduced in [9]. They also proposed train-and-scale and constrain-and-scale techniques to maximize the attack impact while evading anomaly detection. The researchers in [113] introduced an edge-case backdoor that targets data on the tail of the distribution. They also claimed that the backdoors against FL are unavoidable and computationally hard to detect. To make the backdoor stealthier, the scholars in [118] decomposed a

centralized backdoor into parts, and each trigger is injected by a client. The distributed backdoor is more effective and persistent than the centralized backdoor. However, the distributed backdoor is fully activated upon completion of injection of all distributed triggers. Additionally, to survive the newcoming normal updates, the injection of local triggers must be finished in a relatively short attack window. Given that the attacker cannot manipulate the timing of selecting a compromised client to participate in the training, the above conditions are hardly satisfied in practice.

3.2.4 Defenses against FL Backdoor Attacks

Defense against backdoor attacks falls mainly into two categories, robust aggregation and differential privacy.

Robust aggregation. One approach from existing work focuses on building a robust aggregation algorithm that estimates the most possible aggregation rather than directly taking a weighted average. These robust aggregation, such as Foolsgold [35], Krum [11], Bulyan [31], RFA [92] and trimmed mean [124] are designed based on the statistical characteristics of model updates, and aim to identify and deemphasize possibly malicious model updates in the aggregation. Most robust aggregations are built on an assumption of the i.i.d. data distribution across the participating clients. However, this assumption is hardly met in practice. For the FL with non-i.i.d. data among clients, robust aggregation algorithms could mis-identify the non-i.i.d. but normal model updates as malicious or vice versa, and then their weight could be reduced or raised in the aggregation, which degrades the FL model accuracy. These approaches are capable of minimizing the impact of malicious model updates to a certain level, but cannot completely eliminate them [64].

Differential Privacy (DP). DP was originally designed to protect individual privacy. The authors of [108] discovered that by adding noise, the model update could also reduce the effect of malicious model updates. DP has been shown to be effective in mitigating backdoor attacks, but at the cost of model accuracy. The authors of [85] evaluated the effectiveness of both local DP and central DP in defending against backdoor attacks.

3.3 Threat Model and Attack Design Philosophy

3.3.1 Threat Model

We consider the single-shot attack scenario, in which the attacker-controlled client has only one chance to inject the backdoor. And we aim to improve the effectiveness and lifespan of the backdoor injected in the early training stage. First, the attack should be kept stealthy, i.e., the impact on the main task accuracy should be as small as possible. Second, the backdoor injected in the early training stage should remain for a long period.

We assume that the attacker can compromise one or more clients and can interact with the FL model multiple times. In addition to the attacker's capabilities mentioned in [9], such as local data poisoning, local training process control, we also assume that the attacker has the capability of local label distribution adjustment, in which the attacker could use data augmentation and sampling techniques to change the number of samples in each label. The ability to adjust the label distribution may vary for different attackers. To augment the data, attackers can obtain extra data samples from public datasets or use trivial techniques, such as random rotation, random zoom, random crop, etc. For attackers with strong capabilities, they can synthesize data samples from the current local dataset, and reconstruct data samples from the local dataset and gradient leakage [46]. This assumption is practical, as the attacker can easily integrate the above operations into data prepossessing. It is also assumed that attackers can set their own learning rate and local steps to maximize backdoor performance while minimizing impact on the main learning task.

3.3.2 Attack Design Philosophy

Let w_a denote the malicious client's local model. The single-shot backdoor attack achieves its malicious goal by trying to substitute the new global model w^T with a backdoored local model w_a^T in Eq. 3.3. FL aggregation with a backdoored model update is as follows:

$$w^{T+1} \leftarrow w^T + \sum_{k \neq a} \frac{n_k}{n} \Delta w_k^T + \frac{n_a}{n} \Delta w_a. \tag{3.4}$$

The malicious model w_a can only fully substitute the global model by scaling to $\gamma = \frac{n}{n_a}$ and when the global model converges, i.e., $\sum_{k \neq a} \frac{n_k}{n} \Delta w_k^T \approx 0$. When the FL global model converges, the newcoming client model updates are too small to overwrite the backdoor effect. As a result, the injected backdoor can last a long period.

early in

We consider a C-class classification FL problem with cross-entropy loss. The loss function of a client k computed on its local dataset D_k is defined as:

$$F(w; D_k) = \sum_{c=1}^{C} p_k(y = c) \mathbb{E}_{x \in D_k | y=c} [\log f_c(x; w_k)], \tag{3.5}$$

where $p_k(y = c)$ denotes the proportion of class c in D_k, and f_c is the probability that a training sample x belongs to the c-th class.

The CL on the whole population serves as the upper bound of the FL. Due to the non-i.i.d. data distribution among participating clients, and multiple SGDs are performed on the same local dataset, the locally trained model in the FL scheme could introduce weight divergence, which deteriorates the FL global model. And this contributes to the performance gap between CL and FL. Thus, the weight divergence between the models in the CL and FL settings can be used to characterize how good an FL model is.

Consider three models here, the local model w_k of the k-th client, the FL global model w, and the CL model w_{cen} trained on D. Previous work [131, 120] has analyzed the weight divergence of the FL model w and the CL model w_{cen} throughout the training process and tried to catch what causes such a weight divergence. They proved that the weight divergence between w and w_{cen} throughout T global aggregations is bounded by two terms: (1) the sum of the distribution distance between each client's local data and the whole population; (2) the weight divergence inherited from $(T-1)$-th aggregation. And such a divergence is accumulated over time and finally leads to a model accuracy degradation.

Inspired by their work, we are more interested in the intra-aggregation weight divergence, i.e., the weight divergence between two aggregations between w_{cen} and w, and w_{cen} and w_k.

To remove the influence of previous aggregations, we let the CL model and client's local synchronize with the T-th FL global model, $w_{cen}^{T,0} \leftarrow w^T$ and $w_k^{T,0} \leftarrow w^T$. And the CL model and client's local perform t steps training on the whole population data, and their weights after τ steps are:

$$\begin{aligned} w_{cen}^{T,\tau} &= w_{cen}^{T,\tau-1} - \eta \nabla F(w_{cen}^{T,\tau-1}; D) \\ &= w_{cen}^{T,\tau-1} - \eta \sum_{c=1}^{C} p(y=c) \nabla \mathbb{E}_{x \in D|y=c}[\log f_c(x; w_{cen}^{T,\tau-1})] \end{aligned} \tag{3.6}$$

$$\begin{aligned} w_k^{T,\tau} &= w_k^{T,\tau-1} - \eta \nabla F(w_k^{T,\tau-1}; D_k) \\ &= w_k^{T,\tau-1} - \eta \sum_{c=1}^{C} p(y=c) \nabla \mathbb{E}_{x \in D_k|y=c}[\log f_c(x; w_k^{T,\tau-1})]. \end{aligned} \tag{3.7}$$

The weight divergence relationship among the three models can be visualized in Figure 3.1. We have the following proposition.

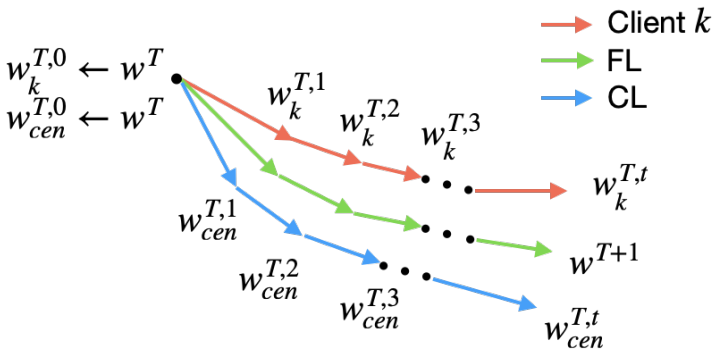


Figure 3.1: Illustration of weight divergence relationship among an FL client's local model, FL global model, and CL model.

Proposition 1. *At the T-th FL global aggregation, let the local model w_k and the CL model on the entire population w_{cen} synchronize with the FL global model w^T, i.e., $w_k^{T,0} \leftarrow w^T$, and $w_{cen}^{T,0} \leftarrow w^T$. And we have $p(y=c) = \sum_{k=1}^{K} p_k(y=c)$, where $p(y=c)$ and $p_k(y=c)$ are denoted as the proportion of the label c on D and D_k. Let each model train for t steps, in which the global aggregation conducts. The model weight divergence between w and w_{cen}, and w_k and w_{cen} after t training steps are bounded by the following two equations, respectively:*

$$
\begin{aligned}
&\|w^{T,t} - w_{cen}^{T,t}\| \\
\leq &\eta \sum_{\tau=1}^{t} \Big[\| \sum_{c=1}^{C} \sum_{k=1}^{K} \frac{n_k}{n} p_k(y=c) \big[\nabla \mathbb{E}_{x \in D_k | y=c} [\log(f_c(w_k^{T,\tau-1})] - \nabla \mathbb{E}_{x \in D | y=c} [\log(f_c(w_{cen}^{T,\tau-1})] \big] \| \Big]
\end{aligned} \tag{3.8}
$$

$$
\begin{aligned}
&\|w_k^{T,t} - w_{cen}^{T,t}\| \\
\leq &\eta \sum_{\tau=1}^{t} \Big[\| \sum_{c=1}^{C} \big[(p(y=c) - p_k(y=c) \big] \nabla \mathbb{E}_{x \in D | y=c} [\log(f_c(w_k^{T,\tau-1})] \| \\
&+ \| \sum_{c=1}^{C} p_k(y=c) \big[\nabla \mathbb{E}_{x \in D_k | y=c} [\log f_c(x; w_k^{T,\tau-1})] - \nabla \mathbb{E}_{x \in D | y=c} [\log(f_c(x; w_{cen}^{T,\tau-1})] \big] \| \Big]
\end{aligned} \tag{3.9}
$$

The proof can be found in Appendices 5, and we have the following remarks.

Remark 1. *The intra-aggregation weight divergence $\|w^T - w_{cen}^T\|$ is determined by the distance between the gradient of the local model taken on $D_k, k \in [1, K]$ and the gradient of the CL model taken on D. This gradient distance can be reduced by increasing the local data sample size. The weight divergence is also an increasing function of the internal training steps t. Therefore, increasing the number of local data samples or decreasing the internal training steps could mitigate weight divergence.*

Remark 2. *The intra-aggregation weight divergence $\|w_k^T - w_{cen}^T\|$ is mainly due to two parts, which are the distribution distance between D_k and D, that is, $\sum_{c=1}^{C} \big(p_k(y=c) - p(y=c)\big)$, and the gradient distance between the gradient calculated on D_k and the gradient calculated on D over classes, that is, $\big[\nabla \mathbb{E}_{x \in D_k | y=c} [\log f_c(x; w_k^{T,t-1})] - \nabla \mathbb{E}_{x \in D | y=c} [\log(f_c(x; w_{cen}^{T,t-1})]\big]$.*

According to Remark 2, the weight divergence $\|w^{T,t} - w_{cen}^{T,t}\|$ could be mitigated by reducing the following two terms: (1) the difference between the data distribution of D_k and that of D, implying the first term in the Eq. 3.9 is reduced; (2) the difference between the gradient calculated on D_k and that calculated on D, which implies the second term in the Eq. 3.9 is reduced.

As a result, a client in an FL setting could benefit from mimicking the distribution and gradients of the whole population to achieve better convergence behavior (faster convergence

or higher model accuracy). This finding is a double-edged sword. On the one hand, a benign client can use it to alleviate weight divergence to facilitate FL convergence, as the data sharing strategy proposed in [131]. On the other hand, the finding could also be taken advantage of by an adversary. As will be shown in the next section, we propose a two-phase backdoor attack, in which the above finding is utilized by an adversary to improve the FL global convergence performance, and further enhance both the strength and persistence for the subsequent single-shot backdoor injection.

3.4 Our Approach

In this section, leveraging the aforementioned insights, we present an overview of our proposed two-phase backdoor attack. Then we describe the detailed workflow of the proposed backdoor attack.

3.4.1 Overview

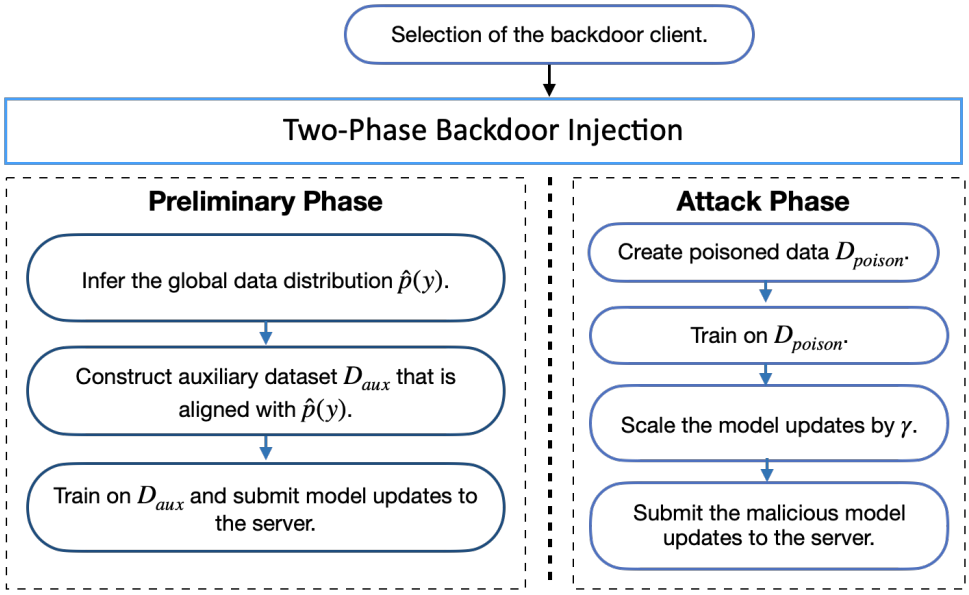


Figure 3.2: The flow chart of the proposed two-phase backdoor attack.

Our proposed two-phase backdoor attack, illustrated in Figure 3.2, consists of a preliminary phase and an attack phase. The backdoor attack can be any kind of existing backdoor attack. Our approach is different from existing backdoor attacks in the proposed preliminary

phase before the attack. The goal of the preliminary phase is to expedite the FL model convergence such that the subsequent backdoor can be more effective and consistent. Specifically, the attacker-controlled client first launches a passive whole population distribution inference attack by analyzing their local model updates and the FL global model update. To reduce weight divergence and improve the convergence behavior of the FL model, the attacker-compromised client then crafts the local training data by augmentation and downsampling such that the distribution $p_k(y)$ aligns with the inferred whole population distribution $\hat{p}(y)$. This step reduces the first term in Eq. 3.9, i.e., the distribution difference $\sum_{c=1}^{C}(p_k(y = c) - p(y = c))$. A dynamic sample size determination method is also utilized in the dataset crafting in order to reduce the second term in Eq. 3.9, i.e., the gradient distance $\big[\nabla\mathbb{E}_{x\in D_k|y=c}[\log f_c(x; w_k^{T,t-1})] - \nabla\mathbb{E}_{x\in D|y=c}[\log(f_c(x; w_{cen}^{T,t-1})]\big]$. Instead of training on the original local dataset, attacker-compromised clients train on the crafted datasets and submit the model updates to the central server. These steps seem legitimate, but they benefit the subsequent injected backdoor by reducing the dilution effect from other client model updates. When the backdoor client is selected or the expected accuracy is reached, the adversary injects the backdoor by training on a poisoned local dataset and scales the malicious model updates by γ to ensure that the injected backdoor survives aggregation before being submitted to the central server.

Our proposed two-phase backdoor attack improves the performance of the early injected backdoor because of the following features:

- We propose a passive whole population distribution inference attack that requires no access to other clients' local data samples nor their model updates.
- By crafting the local dataset, utilizing the inferred whole population distribution and sampling techniques, we are able to reduce the FL model weight divergence, which facilitates the FL model convergence.
- By improving the convergence of the FL model, the backdoor global model is less diluted by model updates from other clients, leading to a stronger and longer-lasting backdoor effect.

3.4.2 Attack Workflow

Preliminary phase: whole population distribution inference

Step 1. Approximation of the CL model updates. The attacker's goal is to estimate the whole population distribution $p(y)$ in the following expression of the CL loss function gradient:

$$\nabla F(w_{cen}; D) = \sum_{c=1}^{C} p(y=c) \nabla E_{x \in D|y=c}[\log f_c(x; w_{cen})]. \tag{3.10}$$

Therefore, $p(y)$ can be calculated if the values of $\nabla E_{x \in D|y=c}[\log f_c(x; w_{cen})]$ and $\nabla F(w_{cen}; D)$ are known. Based on the findings in Remark 1, we approximate the CL model update by the FL model update:

$$\sum_{k=1}^{K} \frac{n_k}{n} \Delta w_k \approx \Delta w_{cen} = \eta \sum_{\tau=1}^{t} \nabla F(w_{cen}^{\tau-1}; D). \tag{3.11}$$

The reasonability of the approximation is demonstrated by: **(1) the bounded and small intra-aggregation weight divergence between the CL model and the FL model.** In Proposition 1, we show that the intra-aggregation weight divergence between a CL model and a FL model is bounded by the difference in the gradient of the local data and the whole population. This gradient difference is usually caused by the difference in the number of samples between the local data and the whole population. The adversary could refer to a public dataset or use augmentation techniques to get a good estimate of the gradient of the whole population. In addition, although the number of internal training epochs increases the bound, the number of local training epochs in practice is relatively small, usually between 2 to 5, and therefore their impact should be minor. As a result, the FL model would not deviate much from the CL model in one aggregation; **(2) the accurate global distribution inferred from the approximation.** Extensive experiments are conducted in Section 3.6.1 to verify that the approximation produces accurate whole distribution inference results. The settings of these experiments are comprehensive, as they cover both the balanced/imbalanced global distribution and the different non-i.i.d.-ness among local data. The results under all settings show that the difference

of the true distribution and the global distribution that is inferred from the approximation is condensed and small.

Step 2. Decomposition of the model updates. Combining Eq. 3.10 and Eq. 3.11, we have the following gradient expression:

$$\sum_{k=1}^{K} \frac{n_k}{n} \Delta w_k \approx \Delta w_{cen} = \eta \sum_{\tau=1}^{t} \sum_{c=1}^{C} p(y=c) \nabla E_{x \in D|y=c}[\log f_c(x; w_{cen}^{\tau-1})]. \tag{3.12}$$

The model update of the compromised client a can be expressed as

$$\Delta w_a = \eta \sum_{\tau=1}^{t} \nabla F_k(w_a^{\tau-1}; D_a) = \eta \sum_{\tau=1}^{t} \sum_{c=1}^{C} p_a(y=c) \nabla E_{x \in D_a|y=c}[\log f_c(x; w_a^{\tau-1})]. \tag{3.13}$$

Normally, the gradient is directly calculated by the partial derivative of the loss, e.g., $\nabla F_k(w_a; D_a) = \frac{\partial F_k(w_a, D_a)}{\partial w_a}$. Taking advantage of the linearity of cross-entropy loss, the gradient $\nabla F_k(w_a, D_a)$ can also be viewed as a weighted average over $\nabla E_{x \in D_a|y=c}[log f_c(x; w_a)]$. If the adversary gets a good estimate of $\nabla E_{x \in D|y=c}[\log f_c(x; w_{cen})]$, the global distribution $p(y)$ can be estimated by minimizing the difference between Eq. 3.12 and Eq. 3.13.

Step 3. Estimation of the gradients. The difference between the gradients calculated on D and D_a is mainly caused by the difference in the size of the data sample. Typically, a larger size of data samples would provide a less biased estimate. The adversary could obtain a more accurate estimate of $\nabla E_{x \in D|y=c}[\log f_c(x; w_{cen})]$ by enlarging D_a using public dataset or data augmentation techniques. However, purely pursuing a large data sample size is not always practical and effective, as some data augmentation methods are computationally expensive and time-consuming, while others could generate similar samples, which could harm the estimation accuracy. Therefore, we adopt a dynamic data size determination algorithm proposed in [16] to determine when to stop the augmentation. The method evaluates the amount of augmentation by measuring the directional distance between the gradient of the augmentation and the gradient estimate. A scaler $\theta \in [0, 1]$, which indicates the cosine similarity between the gradient of augmentation and the gradient estimate, is used to determine when to stop the augmentation. A greater θ indicates a more accurate estimate, meanwhile a greater amount of augmentation.

Step 4. Optimization-based global distribution estimation. In the previous step, the attacker gets a good estimate of $\nabla E_{x \in D|y=c}[log f_c(x;a)]$ by augmenting D_a, the inference of whole population distribution $p(y)$ can then be formulated as an optimization problem, which seeks a $\hat{p}(y)$ that minimizes the difference of two losses in Eq. 3.12 and Eq. 3.13:

$$\hat{p}(y) = \arg\min_{p(y)} \left\| \sum_{k=1}^{K} \frac{n_k}{n} \Delta w_k^T - \eta \sum_{\tau=1}^{t} \sum_{c=1}^{C} p(y=c) \nabla E_{x \in D_a|y=c}[log f_c(x; w_a^{T,\tau-1})] \right\|$$
$$s.t. \sum_{c=1}^{C} p(y=c) = 1, \tag{3.14}$$

where $\sum_{k=1}^{K} \frac{n_k}{n} \Delta w_k^T$ is the FL global model update at the T-th aggregation and can be obtained by taking the difference between the synchronizations of the FL global model $(T-1)$ -th and T -th.

Since the distribution $p(y)$ is not differentiable, an evolution algorithm is used to solve the above optimization. The evolution algorithm begins with a randomly initialized population of $p(y)$, namely, the "fathers". Next, the individuals in the fathers go through mutation and crossover operations with a certain probability to generate more diverse individuals, namely the "children". Then, "fathers" and "children" are evaluated by an objective value, in which the individuals with better objective values will enter the next generation. Algorithm 2 and Algorithm 3 detail the steps to solve optimization.

Algorithm 2 Whole population distribution inference by the evolution algorithm

Input: Number of classes C, population size S.
Output: An estimate of the whole population distribution $\hat{p}(y)$.

1: $g = 0$.
2: Initialize the distribution population $\mathbf{p}_0$, which consists of S individuals. Each individual $p_{0,s}$ satisfies $\sum_{c=1}^{C} p_{0,s}(y=c) = 1$.
3: Compute the FL global model update Δw^T.
4: Evaluate individuals in population $\mathbf{p_0}$ by Algorithm 3.
5: **while** the termination criterion is not satisfied **do**
6: $\quad g = g + 1$.
7: $\quad$ Create population $\mathbf{q}_g$ by crossover and mutation of individuals from $\mathbf{p}_{g-1}$.
8: $\quad$ Evaluate each individual in $\mathbf{p}_{g-1}$ in the children by Algorithm 3.
9: $\quad$ Select S best individuals to population $\mathbf{p}_g$ from the populations $\mathbf{p}_{g-1}$ and $\mathbf{q}_g$.
10: **end while**
11: Return the best individual in population $\mathbf{p}_g$.

Algorithm 3 Evaluation of objective values.

Input: Number of classes C, internal training steps t, learning rate η, the global model update Δw^T, the label composition $p(y)$.

Output: The objective value defined in Eq. 3.14.

The attacker synchronizes with the latest global model $w_a^{T,0} \leftarrow w^T$.
for $\tau = 1 : t$ **do**
  for $c = 1 : C$ **do**
    The attacker calculates the gradient component on class c: $\nabla E_{x \in D_a | y=c}[\log f_c(x; w_a^{T,\tau-1})]$.
  end for
  The model weight is updated by:
  $w_a^{T,\tau} = w_a^{T,\tau-1} - \eta \sum_{c=1}^{C} p(y=c) \nabla E_{x \in D_a | y=c}[\log f_c(x; w_a^{T,\tau-1})]$.
end for
Return the objective value $\|\Delta w^T - \Delta w_a^T\|$, where $\Delta w_a^T = w_a^{T,t} - w_a^{T,0}$.

Preliminary phase: auxiliary dataset construction

After the adversary gets the inference of the whole population distribution, instead of training on the original local dataset, the compromised client trains on an auxiliary dataset, which is crafted to align with the inferred global distribution.

The basic idea of auxiliary dataset construction is to augment the data in classes with inadequate samples and downsample the data in classes with excessive samples based on the inferred whole population distribution. Algorithm 4 describes the steps of auxiliary dataset construction. In particular, the attacker first determines the total size of the auxiliary dataset. The attacker then calculates the amount of data needed for each class by the size of the datset and the inferred global distribution. As for the augmentation operation, the adversary with a limited computation budget can use trivial techniques, such as random shift, random rotation, random shear, and random zoom, while a strong adversary could utilize more advanced methods, such as data synthesis and data reconstruction. For the downsample operation, it randomly samples from current data until the desired number of samples is reached. The auxiliary dataset crafted in this way mitigates both terms in Eq. 3.9.

Attack phase: backdoor injection

The attacker-compromised clients perform training on the crafted auxiliary dataset when selected in FL training until the malicious client capable of launching the backdoor attack is

Algorithm 4 Auxiliary dataset construction.

Input: Auxiliary dataset size M, the inferred data distribution $\hat{p}^(y)$, number of classes C, the compromised dataset D_a

Output: Auxiliary dataset D_{aux}.

Calculate the data size of each class c by $M_c \leftarrow M \times \hat{p}(y = c)$ for $c = 1, ..., C$.
Calculate the data size of each class c of D_a, $|D_a|c|$, where $D_a|c := \{x|y : x \in D_a, y = c\}$.

for $c = 1 : C$ **do**
  if $|D_a|c| < M_c$ **then**
    Augment $|D_a|c|$ to M_c.
  else
    Down-sample from $D_a|c$, such that $|D_a|c| = M_c$.
  end if
  Auxiliary dataset $D_{aux} \leftarrow \cup_{c=1}^{C} D_a|c$.
end for
Shuffle dataset D_{aux}.
Return D_{aux}.

selected. The backdoor client first poisons its local data D_a by adding backdoor triggers to a subset of D_a, and changes their labels to a target one to form a poison data subset D_{poison}. The rest of the data is kept clean and is denoted as D_{clean}. The attacker then performs local training on $D_{poison} \cup D_{clean}$ aiming to maximize the accuracy on both the main task and the backdoor task.

$$w_a^* = \arg\min_{w_a} [F_a(w_a; D_{clean}) + F_a(w_a; D_{poison})].$$

After local training, the attacker scales the model updates by a parameter $\gamma = \frac{n}{n_a} \approx K$ to ensure that the backdoor model survives the aggregation and ideally replaces the global model. The attacker could also use constrain-and-scale or train-and-scale to improve its persistence and evade anomaly detection mechanisms.

3.4.3 Coordination of Multiple Attacker-Controlled Clients

The above presentation of the attack process is based on a single attacker-controlled client, but it can easily be extended to the scenario where the attacker controls multiple clients. The whole population distribution inference attack can be performed by any of the compromised clients. The inferred global distribution is then shared with other attacker-controlled clients, and each

of them constructs and trains on the auxiliary dataset locally. The use of multiple malicious clients can further improve the accuracy of the FL model.

3.5 Experimental Setup

3.5.1 Dataset

We evaluate our proposed method on the handwritten digit recognition data set, MNIST [60]. The dataset contains 60,000 training data samples and 10,000 testing data samples. Each data sample is a square 28×28 pixel image of hand-written single digit between 0 and 9.

3.5.2 Evaluation Metrics

1. **Accuracy of whole population distribution inference attack.** We measure its accuracy by the ℓ_2 distance of the inferred whole population distribution $\hat{p}$ and the true whole population distribution p_{global}, i.e., $\|\hat{p} - p_{global}\|$, referred to as "inferred-to-true". A smaller distance indicates a more accurate inference result. And we also evaluate the ℓ_2 distance of the original distribution on k-th client p_k and p_{global}, i.e., $\|p_k - p_{global}\|$, referred to as "original-to-true". The difference between such two distances is positively related to the amount of weight divergence can be reduced by whole population distribution alignment.

2. **Main task FL model accuracy gain by whole population distribution alignment.** We measure the FL global model accuracy as a function of training epochs for regular FL (clients train on the original datasets) and the FL assisted by whole population knowledge (clients train in crafted local datasets that align with the gradients and distribution of the whole population).

3. **Main task FL model accuracy in presence of backdoor attack.** We also present the accuracy of the main task when the backdoor attack is in place. As mentioned previously, the main task might deteriorate due to the scaling operation and the dilution from the normal model updates, especially when they are large in early training stage. The server could discard the model updates if an unexpected leap or drop in the main task accuracy is observed.

4. **Backdoor attack success rate and longevity.** Given a classifier $f(\cdot)$, the backdoor attack accuracy is defined as the portion of samples in the backdoor samples that are predicted as the target label y_t by the classifier:

$$Acc_{backdoor} = \frac{\{|x \in D_{poison} : f(x) = y_t\}|}{|D_{poison}|}.$$

The test data are constructed by adding the backdoor triggers to the original test data samples. And to avoid the influence of the original data of the target label, we remove the data of the target label in the test data. We plot the backdoor success rate of 20 global epochs since the injection to assess their longevity.

3.5.3 FL System Setting

We implement the FL and the proposed two-phase backdoor attack using the PyTorch framework. We conduct our experiments on Google Colab Pro (CPU: Intel(R) Xeon(R) CPU @ 2.20GHz; RAM: 13 GB; GPU: Tesla P100-PCIE-16GB with CUDA 11.2).

The dataset is allocated to 100 clients. In each global model aggregation, 10 clients are randomly selected to participate in FL training. Each client maintains a local model consisting of two convolutional layers and two fully connected layers. We build four distributed MNIST datasets (Table 3.1) to cover both the balanced/imbalanced whole population and different non-i.i.d.-ness among clients' local data. The global imbalance is simulated by randomly sampling $50\% - 100\%$ for each class from the original dataset. And we use the Dirichlet distribution [82] with a hyper-parameter α to generate different data distributions among clients, where a smaller α indicates a greater non-i.i.d.-ness.

Table 3.1: MNIST dataset settings.

Settings	Whole population	Local distribution
1	balanced	non-i.i.d., $\alpha = 1$
2	balanced	non-i.i.d., $\alpha = 0.1$
3	imbalanced	non-i.i.d., $\alpha = 1$
4	imbalanced	non-i.i.d., $\alpha = 0.1$

Preliminary phase

The clients are randomly selected to participate in a training round, with a certain fraction of clients training on D_{aux}, which aligns with the whole population by Algorithm 4. The FL model is trained with full-batch gradient descent with internal epoch $t = 1$ and learning rate $\eta = 0.1$.

As specified in Section 3.6, the adversary has the capability of augmenting the local dataset by augmentation techniques or accessing public datasets. In our experiment, the adversary is equipped with trivial augmentation methods. We also assume that the attacker holds 1% of the MNIST dataset, from which the attacker can draw data samples and complement the auxiliary dataset. In the dynamic data size determination algorithm that determines when to stop augmentation, we set $\theta = 0.8$, which means that the augmentation operation stops when the cosine similarity between the gradient of augmentation and the gradient estimate reaches 0.8. To avoid the influence of the size of D_{aux}, we set the size of D_{aux} to be the same as that of the original dataset. The fractions of clients controlled by the attacker are chosen to be 5%, 10% and 20% of the total number of clients, denoted as "ours_5", "ours_10" and "ours_20", respectively. And they are collectively referred to as "ours".

Attack phase

We use pixel-pattern backdoors, as the same as those in [118, 9]. We set the 4×4 pixels in the upper left corner of the image to white (pixel value 0) and swap the label with the target label "0". The ratio between the size of the backdoor trigger and the size of the data sample is 2%.

The performance of the proposed backdoor (both the main task accuracy and the backdoor success rate) is evaluated on an FL with mini-batch gradient descent with a batch size of 128. The backdoor client poisons 40 of 128 data samples in each mini-batch and locally trains for poison epochs of 10 with a poison learning rate of 0.05. The global learning rate is the same as the local learning rate $\eta = 0.1$. The scaling factor is $\gamma = K = 10$.

3.6 Experimental Results

3.6.1 Accuracy of the Whole Population Distribution Inference

The global distribution inference attack is launched at every epoch of the first 30 epochs. We present the box plot of $\|p_k - p_{global}\|$ (referred to as “original-to-true") and $\|\hat{p} - p_{global}\|$ (referred to as “inferred-to-true") in Fig. 3.3. In all four settings, compared to “original-to-true", “inferred-to-true" is significantly smaller and more condensed, indicating that the proposed whole population distribution inference attack achieves high accuracy. Furthermore, our proposed inference attack is equally accurate in both balanced and imbalanced whole population distribution settings (setting 1 vs. setting 2 and setting 3 vs. setting 4).

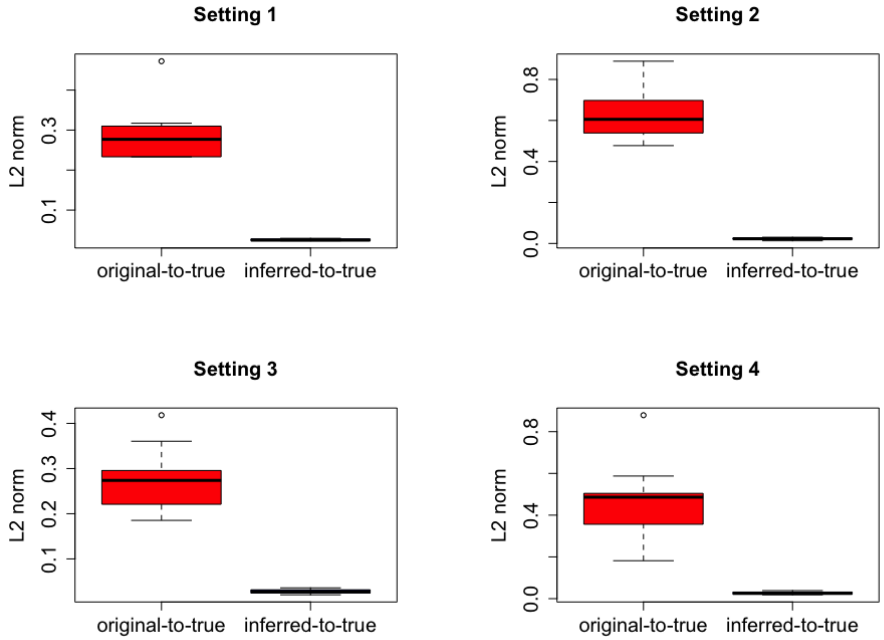


Figure 3.3: Box plot of $\|p_k - p_{global}\|$ (“original-to-true”) and $\|\hat{p} - p_{global}\|$ (“inferred-to-true”).

We also plot the “inferred-to-true” as a function of training epochs (shown in Fig. 3.4). The FL model begins to converge at epoch 20, so our inference attack window covers different convergence stages of the training process. The results show that the inference results are stationary along the training process, which means that inferring at any training stage does not affect the inference accuracy. The fluctuations presented in Fig. 3.4 are due to the randomness

of the local distributions in the selected clients in each FL training round. Especially, the fluctuation becomes more noticeable when clients' local distributions are more non-i.i.d. (Settings 2 and 4). To further reduce such fluctuations and improve the accuracy of the inference, the adversary could further refine the inference result by performing statistical analysis on multiple inference results, such as averaging or clustering.

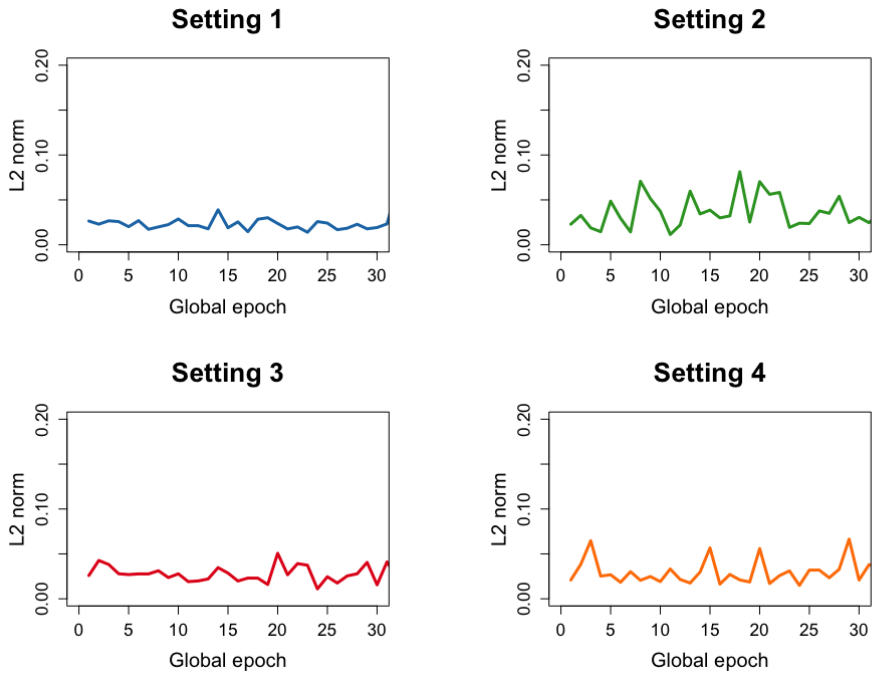


Figure 3.4: $\|\hat{p} - p_{global}\|$ ("inferred-to-true") vs. the global training epoch.

3.6.2 Main Task Accuracy under the Non-Attack Scenario

We evaluate the effectiveness of the proposed preliminary phase in improving FL convergence by the accuracy of the FL main task, shown in Fig. 3.5. In all 4 settings, compared to FedAvg, the FL with global distribution alignment converges faster, although they eventually reach the same accuracy. This performance gain is more perceptible before the FL begins to converge and when a greater fraction of clients perform the proposed alignment. In addition, while the global distribution alignment has more influence on the very early stage (epoch 0 to epoch 10) for setting 1 and setting 3 ($\alpha = 1$), a higher non-i.i.d.-ness ($\alpha = 0.1$ in setting 2 and setting 4) has more impact on the middle training stage (epoch 5 to epoch 15). The experimental

results are consistent with the findings of Proposition 1: reducing both the gradient and distribution between the client's local data and the whole population could reduce the model weight divergence, leading to a better convergence performance.

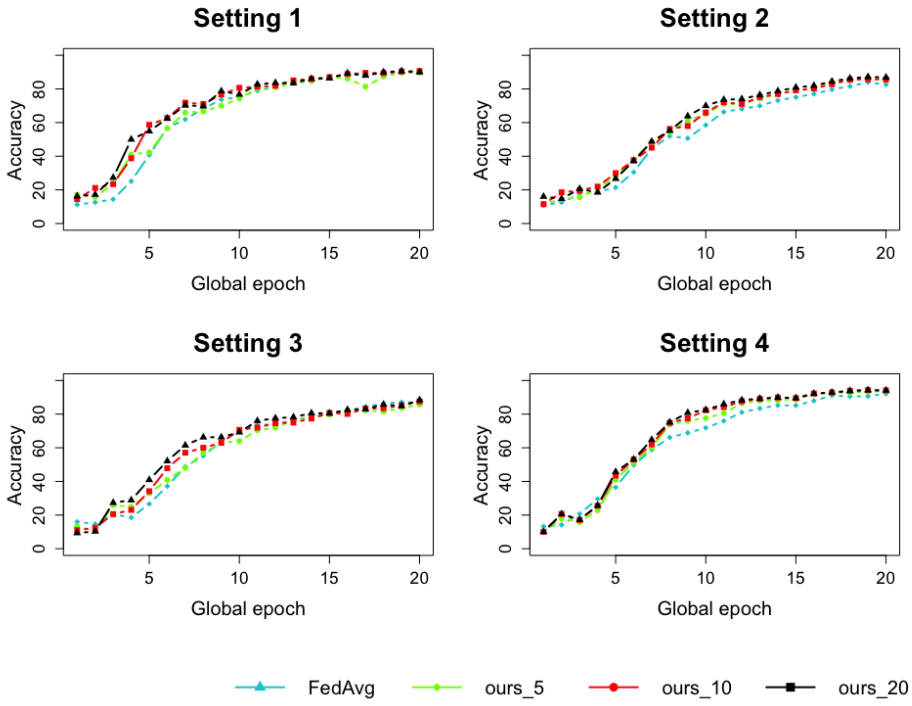


Figure 3.5: The accuracy of the main task of 5%, 10%, and 20% of the local data of the clients who perform alignment in 4 settings, averaged over 10 experiments.

3.6.3 Backdoor Attack Performance

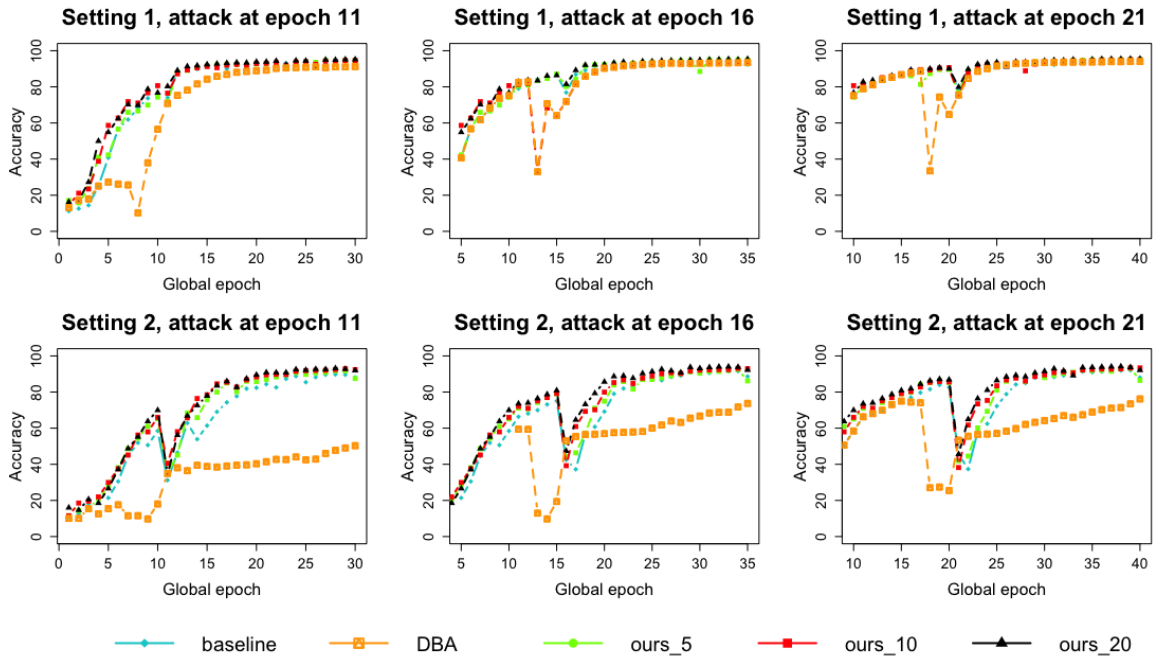
We present the impact of backdoor injection on the main task accuracy as well as the backdoor success rate. We evaluate the proposed two-phase backdoor attack and compare it with two existing backdoor attacks: (1) the centralized backdoor attack [9] (referred to as "baseline"), in which the local dataset is poisoned by a centralized backdoor trigger; (2) the distributed backdoor attack [118] (referred to as "DBA"), in which the backdoor trigger is divided into parts and each part is injected separately.

Main task accuracy

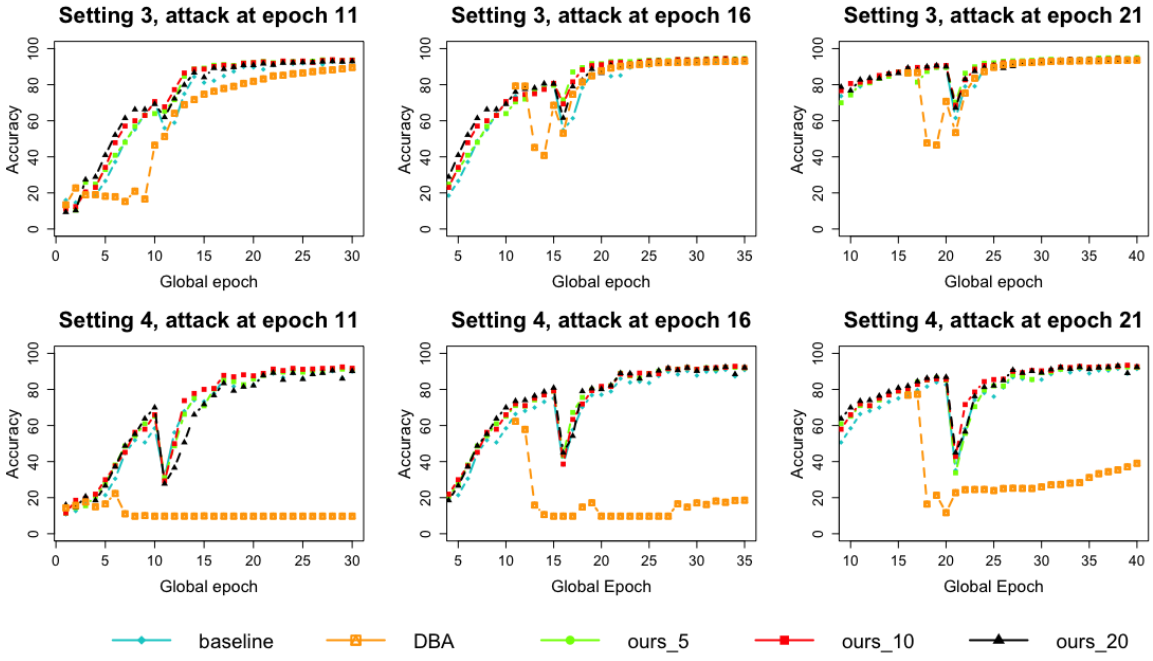
Unlike the backdoors injected at the convergence of the FL model, where the injection of the backdoor barely disturbs the accuracy of the main task, the early injected backdoor usually

noticeably deteriorates the accuracy of the main task due to not small enough model updates from normal clients. When the backdoor is injected in the early training stage, the accuracy of the main task usually experiences a sudden drop and then gradually goes back to normal status afterward. As introduced in [102], the central server could monitor the FL model main task accuracy and reject model updates that make the main task accuracy abnormally low. This approach could fail to be deployed on the FL system, since the central server does not always have access to the model updates and test data, thus cannot measure their accuracy, or a false alarm could be triggered due to the extremely low local accuracy caused by the participation of clients with highly imbalanced local data. However, the main task can still be used to evaluate the stealthiness of the backdoor attack.

As shown in Fig. 3.6, the accuracy of the main task is affected by the backdoor injection in varying degrees. The dropped main task is a collective consequence of the scaled backdoored model updates and not small enough model updates from the rest of participating clients. And such a main task accuracy drop becomes more critical for a greater non-i.i.d.-ness among clients (setting 2 and setting 4). Compared to the "baseline", "ours" introduces less drop in main task accuracy in most cases. And in some cases, the main task accuracy impacted by our proposed backdoor attack presents a faster recovery rate. Furthermore, compared to the "baseline" and "ours", the "DBA" suffers the greatest drop in the main task accuracy and it takes much longer for the underlying FL to return to the normal main task accuracy. This phenomenon is even worsened in the setting of high non-i.i.d.-ness (setting 2 and setting 4). A possible explanation is that "DBA" requires multiple clients to sequentially perform injection of part of the backdoor trigger to complete injection of a complete backdoor, which poses a longer and worse impact on the main task accuracy. Especially in the highly non-i.i.d. and globally unbalanced scenario, given that the model updates are already far from others, the consecutive injection and scale operations could make the deviation even worse and prevent the FL model from convergence (evidenced in setting 4). Thus, we conclude that the proposed backdoor attack is more stealthy than the "baseline" and "DBA".



(a) Setting 1 and setting 2.



(b) Setting 3 and setting 4.

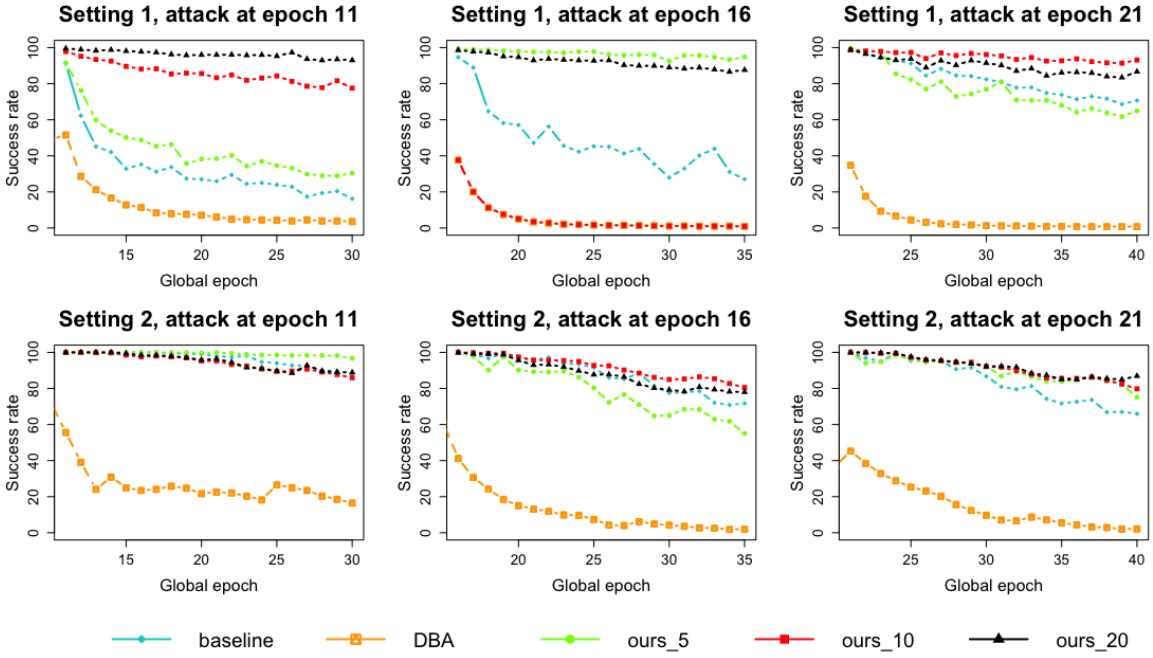
Figure 3.6: The main task accuracy of the FL global model when the backdoors are injected at FL epochs 10, 15, and 20, respectively.

Backdoor attack accuracy

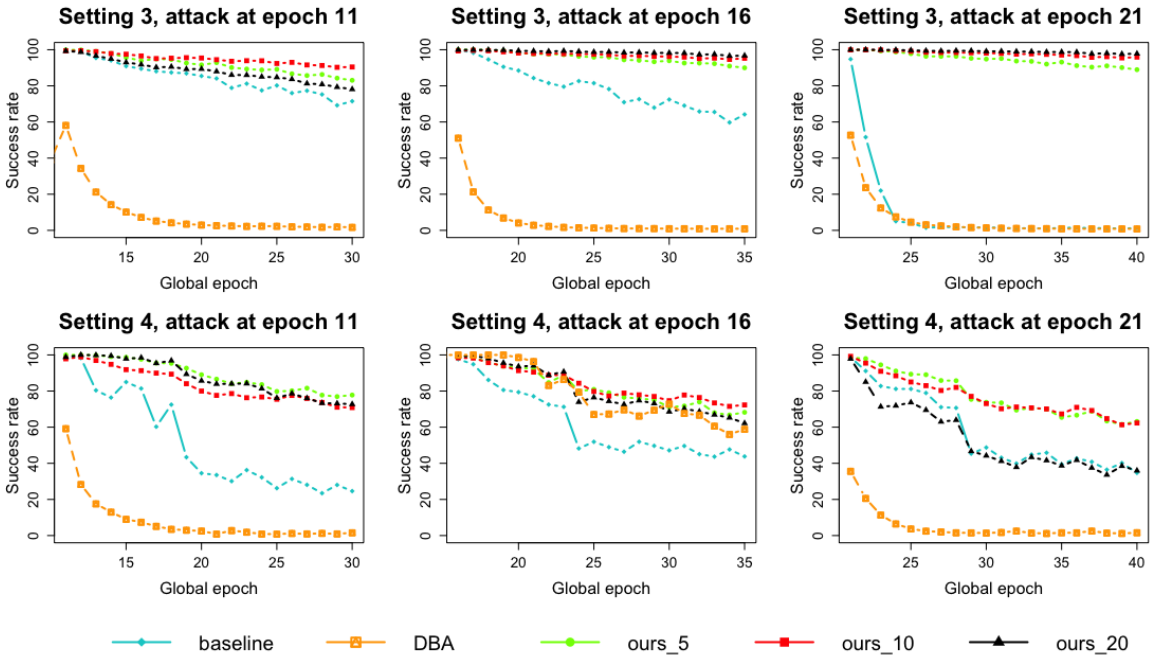
To explore the effectiveness of a backdoor before the FL model converges, we inject the centralized backdoors (“baseline” and “ours”) at FL global epochs 11, 16, and 21, respectively. To fairly compare with “DBA”, the distributed backdoors are sequentially injected and finished in the same round as the centralized backdoors. For example, if the centralized backdoor is injected in round 11, 4 distributed backdoor triggers are injected separately in rounds 8, 9, 10, and 11.

Fig. 3.7 presents the backdoor success rate for 20 FL global epochs since the completion of the backdoor injection. For each setting, injections at different epochs are performed by the same client. The injected backdoor reaches maximum effectiveness immediately after injection. In the subsequent epochs, as the FL model aggregates new normal updates, the effect of the backdoor is weakened, which is reflected by the gradually decreasing success rate. In most cases, after 20 rounds since backdoor injection, the success rates of almost all settings and injection epochs are greater than that of the “baseline” and “DBA”. “DBA” does not reach a comparable backdoor effect as in “baseline” and “ours”. The reason for this gap could be that the partially injected backdoor effect in previous rounds is more likely to be hindered by normal local updates in the subsequently injected backdoor parts. And in most cases, our proposed backdoor retains a lower diminishing rate, compared to the “baseline”.

Due to the non-i.i.d.-ness among clients’ local data, some clients’ data may be in favor of the attack, while others are not. In addition, the backdoor effect does not always steadily decrease, and it bounces in some cases. Therefore, we evaluate both attack strength and longevity by the mean attack success rate of 10 FL epochs since injection (Table 3.2). In general, the backdoor injected in very early rounds (epoch 5 and epoch 10) achieves a lower mean attack success rate, compared to the ones injected in epoch 20. This degradation in the effectiveness of the attack is made even worse when the whole population is imbalanced (setting 1 vs. setting 3) and non-i.i.d.-ness among clients increases (setting 1 vs. setting 2). In most cases, our proposed backdoor attack outperforms the “baseline” and the “DBA”. And compared to “baseline”, the attack performance gain is positively related to the fraction of attacker-controlled clients performing the whole population distribution alignment.



(a) Settings 1 and setting 2



(b) Settings 3 and setting 4

Figure 3.7: The backdoor success rate in 20 training epochs since backdoor injection.

Table 3.2: Mean backdoor success rate(%) over 10 FL epochs since backdoor injection (averaged over 10 randomly selected clients).

Attack epoch	Baseline	DBA	Ours_5	Ours_10	Ours_20
			Setting 1		
11	95.80 ± 1.84	23.25 ± 4.19	94.06 ± 5.97	94.50 ± 1.49	95.93 ± 2.46
16	97.40 ± 2.39	17.23 ± 5.36	97.40 ± 1.34	95.59 ± 3.91	95.34 ± 1.61
21	95.33 ± 5.32	14.46 ± 1.32	95.32 ± 4.24	96.31 ± 3.35	95.68 ± 1.87
			Setting 2		
11	52.91 ± 4.66	77.78 ± 28.42	73.96 ± 22.98	76.58 ± 12.17	80.57 ± 27.21
16	61.25 ± 27.53	67.78 ± 10.82	75.01 ± 22.62	82.25 ± 14.36	78.44 ± 17.75
21	68.38 ± 22.28	11.78 ± 2.22	79.24 ± 19.67	79.72 ± 13.61	77.10 ± 18.13
			Setting 3		
11	15.46 ± 6.35	13.54 ± 3.24	44.82 ± 25.61	65.27 ± 28.11	66.14 ± 25.11
16	57.71 ± 16.50	9.7 ± 1.86	69.29 ± 18.25	66.11 ± 28.66	62.15 ± 20.13
21	64.44 ± 13.77	6.32 ± 7.81	73.41 ± 14.73	69.74 ± 13.62	72.38 ± 16.48
			Setting 4		
11	48.70 ± 41.45	52.34 ± 10.33	67.13 ± 18.19	75.14 ± 26.52	88.28 ± 10.11
16	68.98 ± 4.21	40.67 ± 9.87	72.33 ± 15.72	83.26 ± 23.27	72.22 ± 17.28
21	70.33 ± 1.94	11.7 ± 4.24	73.41 ± 14.73	88.09 ± 13.39	85.51 ± 8.28

3.6.4 Overhead Analysis

Preliminary phase

The computational cost of this phase consists of three parts: (1) calculating the gradients on the data of each label; (2) solving the optimization in Eq. 3.14; (3) constructing the auxiliary dataset.

For the first part, the attacker trains the FL global model on the data samples of each label separately to obtain gradients $\nabla \mathbb{E}_{x \in D_a | y=c}[\log f_c(x; w_a)]$, and because $n = \sum_{c=1}^{C} n_c$, where n_c is the number of samples in label c, the time complexity is the same as that of local training. Since the batch gradient has a time complexity of $\mathcal{O}(n^2 m)$, in which n is the number of data samples and m is the number of features, the time complexity of the first part is also $\mathcal{O}(n^2 m)$.

For the second part, we evaluate *number of function evaluations* (NFEs), which is commonly used to evaluate an evolution algorithm. NFE is usually measured when a good solution is delivered or when no significant change in the solution is observed. We plot "inferred-to-true" and the real time used against NFE, shown in Fig. 3.8, to demonstrate the effect of NFE on inference accuracy and inference time. Because there is no significant inference accuracy

gain after 400 NFEs, we set the NFE to 400 in our experiment. The real time taken to solve the optimization with 400 NFEs is 4 seconds.

The construction of the auxiliary data set that is aligned with the whole population consists of augmentation and sampling operations. Examples of trivial augmentation methods are flipping ($\mathcal{O}(np)$, where p is the number of pixels in each image), rotation, random crop, and scale (they have the same complexity of $\mathcal{O}(n)$). The sampling operation has a time complexity of $\mathcal{O}(n)$. Therefore, the total time complexity of the construction of the auxiliary dataset is at most $\mathcal{O}(np)$ and the real time spent is 0.03 seconds.

Backdoor phase

The backdoor client poisons a subset of the local data by injecting the backdoor pattern and swaps the label to the target ones, then performs local training on the poisoned local dataset. The total time complexity is $\mathcal{O}(n^2m)$. The real time spent on the backdoor attack with 10 internal training epochs is around 13 seconds.

The complexity analyses are summarized in Table 3.3. Gradient calculation and solving optimization are only needed to be performed a few times to get an accurate whole population distribution inference result. The real time for these two steps is less than 5 seconds, which is minor compared to the time taken for the backdoor attack. Once the whole population distribution is inferred, attacker-controlled clients only perform the auxiliary dataset construction, whose time complexity is negligible.

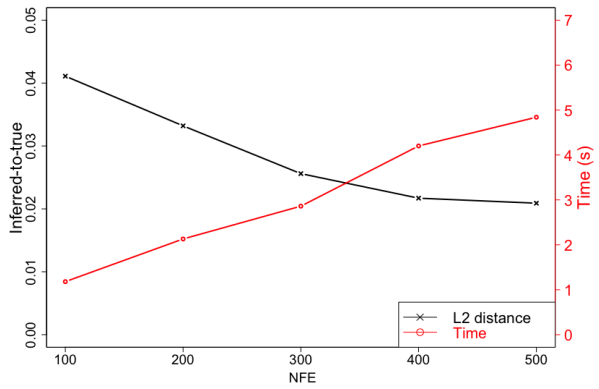


Figure 3.8: The inference accuracy ("inferred-to-true") and time taken vs. NFE.

Table 3.3: Time complexity and real time spent on the proposed inference attack.

Operation	Time complexity	Real time taken (s)
Gradient calculation	$\mathcal{O}(n^2m)$	0.11
Solving optimization	400 NFEs	4.20
Auxiliary dataset construction	$\mathcal{O}(nk)$	0.03
Backdoor attack	$\mathcal{O}(n^2m)$	13.37

3.7 The Robustness of the Proposed Attack

In this section, we are interested in how the proposed attacks will behave when defense mechanisms are in place. In the following, we will analyze the effectiveness of the proposed two-phase backdoor attack against two main defense strategies.

FoolsGold is a secure aggregation strategy, which calculates the cosine similarity of all historical gradient records and assigns smaller aggregation weights to clients that repeatedly contribute similar gradient updates [35].

DP is a noise-based method that limits the efficacy of backdoor attacks by two key steps [85]: (1) model parameters are clipped to bound the sensitivity of local model updates; (2) Gaussian noises are added to local model updates. We consider a local DP, in which each client adds noises before uploading the model updates to the server. We use the (ϵ, δ)-DP proposed in [1] with a popular choice of $\sigma = \sqrt{2\log\frac{1.25}{\delta}}/\epsilon$ with $\delta = 10^{-5}$ and $\epsilon = 50$. The clipping bound is set to the median of the norms of the unclipped local model updates during training. The noises are only applied to normal model updates, while the backdoor client sends the non-perturbed backdoored model updates.

3.7.1 Whole Population Distribution Inference Accuracy against Defense Strategies.

We first present the whole population distribution inference accuracy against FedAvg, FoolsGold, and DP, shown in Fig. 3.9. Since FoolsGold does not interfere with benign FL settings, the whole population distribution inference against FoolsGold is as accurate as that in FedAvg. Although DP provides a statistical guarantee for record-level information, DP fails to protect statistical information, such as the whole population distribution. With DP in place, although the inference is not as accurate as in the FedAvg case, the "inferred-to-true" is still notably

lower compared with "original-to-true". Thus, both FoolsGold and DP fail to defend against the proposed inference attack.

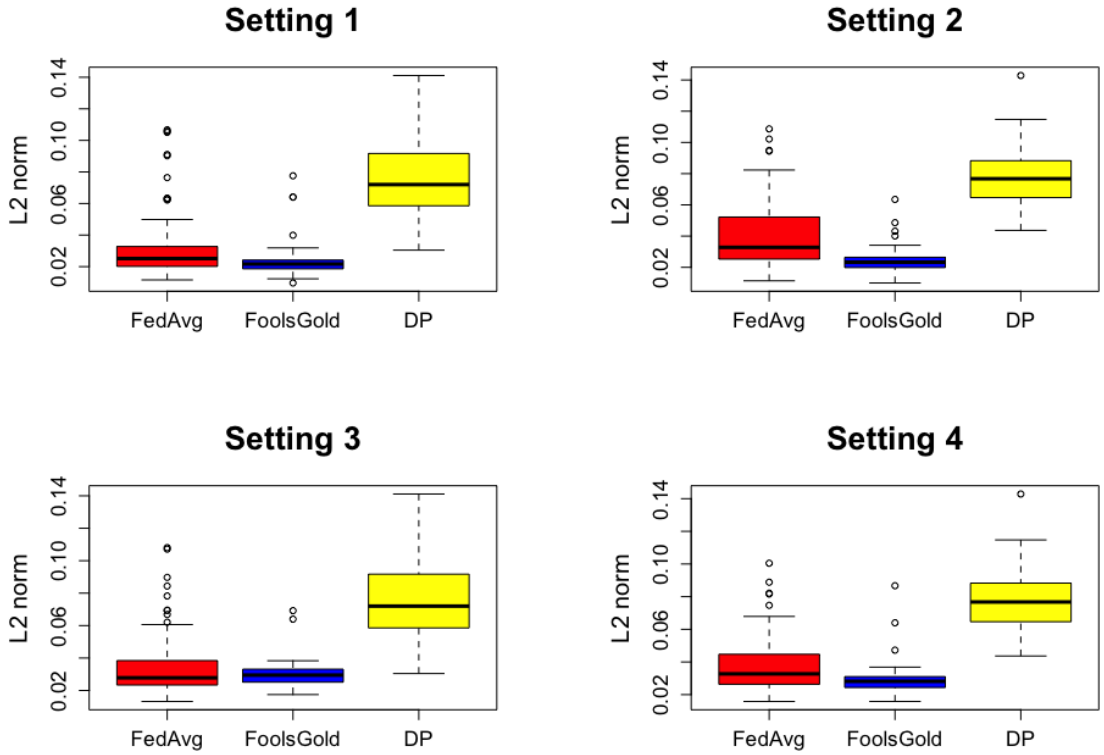


Figure 3.9: Box plot of "original-to-true" and "inferred-to-true" ($\|\hat{p} - p_{global}\|$) of FedAvg, FoolsGold and DP based on 30 instances.

3.7.2 Performance of the Backdoor Attack against Defense Strategies.

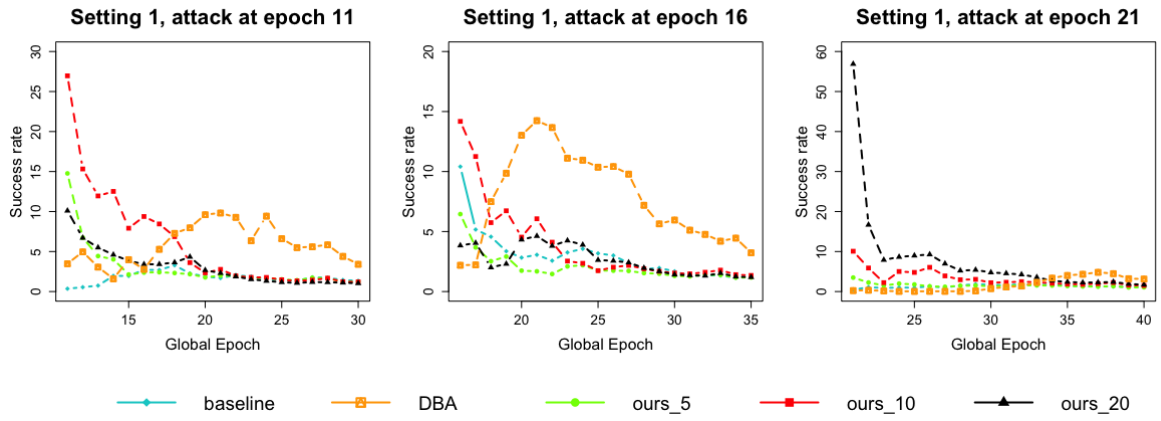
We implement the proposed backdoor attack against FoolsGold and DP in setting 1. We plot the backdoor success rate for 20 epochs after injection to observe its injection strength and longevity. Fig. 3.10(a) shows that in most cases "ours" reaches a significantly higher attack rate after injection through the backdoor and maintains such a high success rate in the epochs after injection. For example, when injected in the early training stage (in epoch 11), the baseline backdoor fails while "ours_5", "ours_10"and "ours_20" achieve attack success rates of 14%, 27% and 10%, respectively. As the convergence is expedited by the proposed preliminary phase, the model updates from normal clients become smaller and more similar, so that the FoolsGold will reduce their assigned weights, and as a result the backdoored model updates become more influential. Compared to "ours" and "baseline", the success rate curve of the

"DBA" has a different pattern, in which the success rate first increases and then decreases and is more persistent than both the "baseline" and "ours" in later rounds. However, in practice, the requirement for clients with distributed backdoor triggers to be selected in consecutive rounds cannot hardly be met.

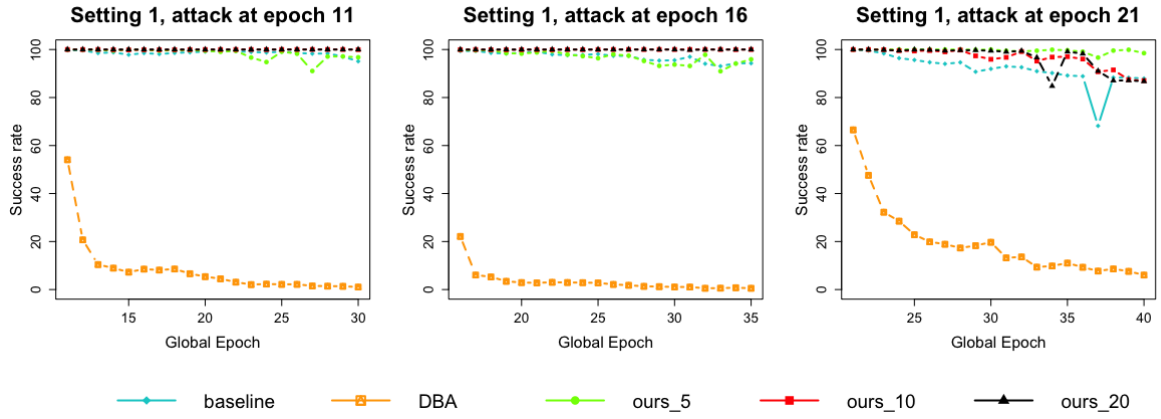
Fig. 3.10(b) shows the attack performance against DP. Both the "baseline" and "ours" achieve high attack success rates, even comparable to those of FedAvg, in which no defense mechanism is applied. This phenomenon indicates that instead of mitigating the backdoor effect, the noise added to the normal clients helps the backdoored model corrupt the FL global model. A possible explanation is that the added noise reduces the utility of normal model updates, which, in turn, strengthens the backdoored model updates in the FL aggregation. Furthermore, "ours" is markedly better than the "baseline" when the backdoor is injected in later rounds. For example,"baseline" and "ours" have similar attack performance in the early training stage, e.g., epoch 11. And "ours" performs distinctly better in the later training stage, that is, the backdoors injected in epoch 16 and epoch 21. Lastly, both the "baseline" and "ours" outperform DBA. This is because the effectiveness of the distributed backdoor is mitigated by both benign model updates and DP noise multiple times before DBA finishes the injection of the complete backdoor.

3.8 Conclusions

In this work, we proposed a novel information leakage-assisted single shot backdoor attack that improves the effectiveness of the backdoor injected in the early training stage. We first showed that clients training on datasets that are aligned with the whole population in both distribution and gradient can improve the FL model convergence. Based on this observation, we introduced a preliminary phase to the subsequent backdoor attack, in which the attacker-controlled clients first infer the whole population distribution from the shared FL model updates and then train on locally crafted datasets that are aligned with both the distribution and gradient of the whole population. Benefiting from the preliminary phase, the subsequent backdoor injection suffers less dilution effect from the model updates of other clients and achieves better effectiveness. We demonstrated the effectiveness of the proposed backdoor attacks in the early training



(a) FoolsGold



(b) DP

Figure 3.10: Backdoor success rate (%) of 20 training epochs since injection against Foolsgold (a) and DP (b) defense mechanisms.

stage through extensive experiments on a real-world dataset. The results have shown that the proposed backdoor can have a longer lifespan than existing backdoor attacks. We hope that our work brings attention to the vulnerabilities in the early training stage of FL. Our analysis and findings provide novel insights into the field of strengthening FL attacks by information leakage, which could help evaluate and improve the robustness of FL.

Chapter 4

High-Accuracy Low-Cost Privacy-Preserving Federated Learning in IoT Systems via Adaptive Perturbation

4.1 Introduction

The development of the Internet of Things (IoT) enables the connection of a wide range of devices to the Internet [69] to provide ubiquitous sensing and computation capabilities. The data collected by these devices can be used to train machine learning models. Although the data on one device may be insufficient to obtain a satisfactory model, the data on other devices can be benefited via network communication. Federated learning (FL) [55, 78] allows a machine learning algorithm to learn from data stored on a wide range of physically separated devices. Technically, FL is a distributed learning system, which allows multiple local clients to collaboratively train a high-accuracy global model by taking advantage of a wide range of data without sharing their local collected data. FL has found its applications in most emerging services and systems, e.g., in mobile applications such as next-word and emoji prediction on smartphones [22, 123, 97], environmental monitoring [40], smart healthcare [122, 15] and smart city [94].

Although clients do not directly reveal their private data, shared model updates can unintentionally leak sensitive information about the data on which they were trained [28]. As pointed out by previous studies, using FL scheme alone is insufficient in protecting the clients' local data privacy. For example, from the shared FL model, an adversary can infer whether a given data sample was presented in the training data or not [79, 86], or recover a representative data sample used in the training [34], or infer property information about the client's local training data [132].

Ideally, FL with a privacy-preserving mechanism on IoT devices, such as smartphones, smart watches, and cameras, should take into account the following **constraints**:(1) computational capacity is limited, so computationally expensive encryption algorithms are unaffordable; (2) devices have limited power supply and network connectivity; (3) clients are flexible to join or leave training, so dropouts are common.

Several studies have focused on how to preserve privacy in FL. But none of them can fully address the aforementioned constraints. In particular, the main approaches are secure multiparty computation (MPC) and differential privacy (DP). A branch in MPC is based on homomorphic encryption. Paillier cryptosystem is an additive homomorphic encryption algorithm [121, 90, 13], which naturally matches the aggregation operation in FL. But the main drawback is its high computational complexity. Another approach uses secret-sharing [13], which is relatively computationally efficient and can handle client dropout as well. However, the requirement of information exchange between each pair of clients makes this approach impractical in moderate to large-scale IoT systems. DP is a promising solution that injects random noise into the data or the model updates, providing a statistical privacy guarantee for individual records and privacy protection against inference attacks. However, privacy protection comes at the cost of model accuracy. Additionally, one challenge in training with DP is choosing an appropriate clipping bound. An inappropriate clipping bound can degrade model accuracy or even prevent a model from converging due to the bias introduced by the clipping operation [24].

In this work, we propose a novel low cost (for both communication and computation overhead) adaptive noise-perturbation privacy-preserving scheme, which does not sacrifice FL model accuracy for privacy, while enjoying a DP-comparable or in some cases better privacy protection. More specifically, our scheme protects local privacy by adding random noises to each local model updates (i.e., perturbing local model updates by adding random noises). These random noises are deliberately designed so that individually they can provide sufficient protection for the privacy of each local model. But when combined at the FL server, the aggregation of these noises will present a cancel-out effect by the central limit theorem (CLT), so that the aggregated noises at the server are more condensed and help to preserve the global model accuracy. In real FL applications, the number of clients is much

larger than 30, which is considered sufficient for CLT to hold. In addition, unlike the random noise in the DP scheme, our noise masking scheme takes both magnitude and direction into consideration when adding noise to local model updates to retain high global model accuracy and expedite global model convergence. Specifically, we introduce an adaptive noise scaling method that sets the magnitude of the random noise proportional to the magnitude of the local model updates, i.e., the magnitude of noise changes with that of local model updates at the same rate, which ensures sufficient privacy protection while preventing the introduction of excessive noise, especially when the FL model is close to convergence. To maintain the same convergence rate and accuracy as in regular FL, the noise scale is chosen on the basis of the number of participating clients, so that the magnitude of the aggregation of noise does not exceed the magnitude of local model updates. Moreover, we monitor the angular distance, calculated from cosine similarity, between the true local model updates and the noise-perturbed local model updates. Noise with a large angular distance will be filtered out, making it easier for the global model to converge. With deliberately chosen noise magnitude and angular distance, the FL with the proposed noise scheme achieves the same convergence performance as regular FL and DP-comparable or better privacy protection against state-of-the-art DP frameworks [37, 1].

To the best of our knowledge, we are the first to take both magnitude and direction into consideration aiming at protecting FL clients' privacy while preserving the FL model accuracy. Our **contribution** in this paper is threefold:

- For a strongly convex loss function, we prove that a noise-perturbed FL is guaranteed to converge to the same value as the regular FL (i.e., there is no accuracy loss) as long as the magnitude of the added noise is proportional to the magnitude of the local model update. Given the number of clients participating in the perturbed FL, we also derive the maximum tolerable variance of the added noise at individual clients that guarantees that the magnitude of the aggregated noise at the FL server does not exceed the magnitude of the aggregation of all local model updates (i.e., the direction of descent is still preserved), so that the perturbed FL maintains the same convergence rate $\mathcal{O}(1/T)$ as that of SGD on convex loss functions. These theoretical findings enable us to develop the proposed adaptive noise perturbation scheme that maximizes privacy protection for clients while

maintaining the same accuracy as that of regular FL. We also provide a statistical method to select the angular distance threshold based on the dimension of the model updates to accelerate the convergence of the perturbed FL.

- For the non-convex loss function scenario, we derive the worst-case convergence bound for FL under the proposed noise perturbation scheme. This bound shows that the noise-perturbed learning process converges at a rate of $\mathcal{O}(1/\sqrt{T})$, the same as that of an SGD on non-convex functions. With the proposed angular distance filtering scheme, our proof indicates that the actual convergence is faster than the derived worst-case convergence bound.

- Extensive experiments are conducted on MNIST and CIFAR-10 datasets to validate our theoretical convergence analyses and evaluate the time and computational efficiency, as well as the effectiveness of the proposed scheme in defending against state-of-the-art privacy inference attacks. The numerical results show that the proposed scheme outperforms DP in convergence rate and accuracy in both dropout and non-dropout scenarios, which are consistent with our theoretical convergence analyses. The proposed scheme does not incur extra computation and communication overhead compared with DP. Our proposed noise perturbation scheme provides comparable or, in many cases, stronger privacy protection than DP, under the same global model accuracy.

The rest of this paper is organized as follows. Section 4.2 briefly reviews the FL and related work. Section 4.3 presents our threat model and security goals. Section 4.4 describes our proposed additive noise scheme. Theoretical convergence analyses are provided in Section 4.5. The settings and results of the experiments are presented in Section 4.6 and Section 4.7, respectively. We conclude our work and recommend future research directions in Section 4.8. And detailed proofs of our key findings are given in the Appendix.

Throughout this paper, we use the following **notation**:

- $\|\cdot\|$ denotes the ℓ_2 norm.

- $<_\epsilon$ denotes slightly greater than. $a <_\epsilon b$ means $b = a + \epsilon$, where $\epsilon \in \mathbb{N}^+$.

- D denotes the global data and is distributed to N clients, where $D = \cup_{n=1}^{N} D_n$, and D_n denotes the data on the n-th client. A subset of K clients ($K < N$) is selected to participate in a round of FL training.
- $F_k(\cdot)$ and $F(\cdot)$ denote the loss function on the client k and the global loss function, respectively.
- $\nabla F_k(\cdot)$ and $\nabla F(\cdot)$ denote the gradients of the local loss function and the global loss function, respectively.
- $w_k^{T,\tau}$ denotes the local model weight of client k in τ-th local step in T-th global aggregation, and w^T denotes the global model weight in T-th global aggregation.
- $\tilde{w}_k^T$ and $\tilde{w}^T$ denote the noise-perturbed local model weight and the noise-perturbed global model weight at T-th aggregation, respectively.
- r_k^T denotes the additive noise in the client k in the T-th global aggregation.

4.2 Preliminary and Related Work

4.2.1 Federated Learning

The global data $D = \cup_{n=1}^{N} D_n$ are distributed to N clients and each client maintains its local data D_n. Each time, K ($K \leq N$) of N clients are selected to participate in the training. Specifically, each client maintains a local model trained from the local training dataset. A central server maintains a global model by aggregating the local model updates from the participating clients in each round. The objective of FL training is to minimize the loss:

$$F(w) = \sum_{k=1}^{K} F_k(w), \tag{4.1}$$

by optimizing over the model parameter w, where $F_k(w)$ is the loss function on the local data of the k-th client :

$$F_k(w) = \frac{1}{|D_k|} \sum_{(x,y) \in D_k} L(w; (x, y)), k \in [K], \tag{4.2}$$

where L is the empirical loss function. Here, we describe FedAvg, which is probably the most widely used FL algorithm. FedAvg iteratively performs the following three steps (illustrated in Figure 4.1):

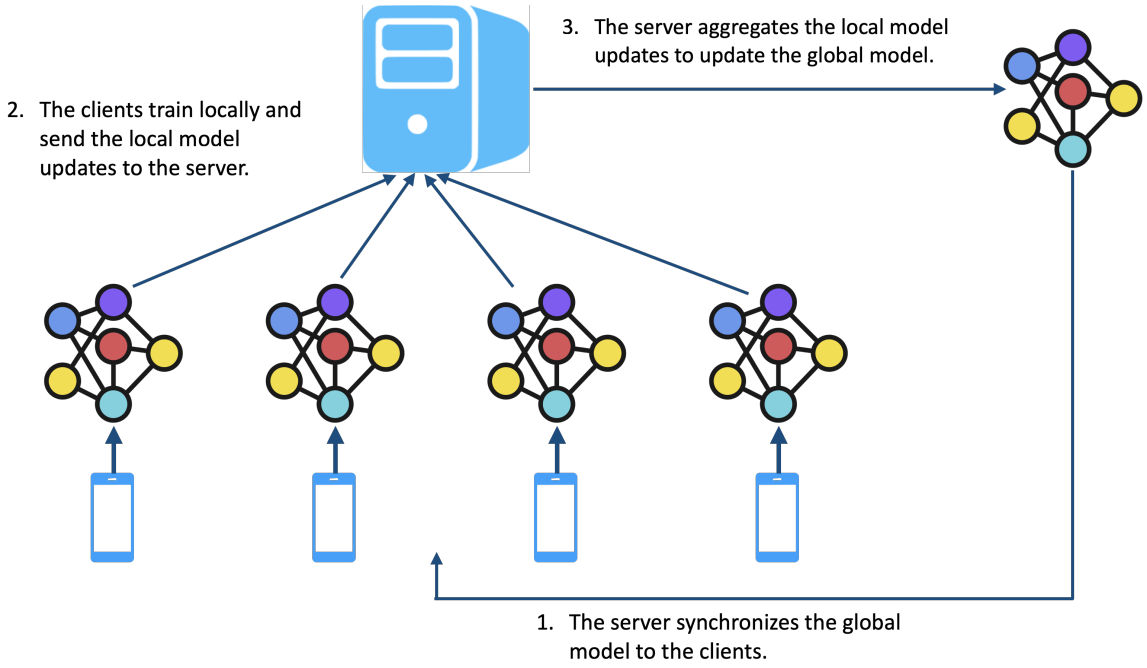


Figure 4.1: An illustration of the FL process.

Global Model Synchronization

In the T-th global aggregation, the central server randomly selects K from N clients and broadcasts the latest global model w^T to the selected clients: $w_k^{T,0} \leftarrow w^T$.

Local model training

Each client k updates its own local model w_k by running an SGD on the local dataset D_k for t steps. The τ-th step on the client k follows :

$$w_k^{T,\tau+1} \leftarrow w_k^{T,\tau} - \eta \nabla F_k(w_k^{T,\tau}, \xi_k^{T,\tau}), \tag{4.3}$$

where $\xi_k^{T,\tau}$ is a mini-batch of samples randomly chosen from the local dataset D_k, and η is the local learning rate.

Global Model Update

After performing local training for t steps, the client transmits the model updates $\Delta w_k^T = w_k^{T+t} - w^T$ back to the central server. The central server then updates the global model by performing a weighted average on the local model updates sent from K clients :

$$w^{T+1} \leftarrow w^T + \sum_{k=1}^{K} \frac{n_k}{n} \Delta w_k^T, \tag{4.4}$$

where $n_k = |D_k|$ is the number of training data on the client k and $n = \sum_{k=1}^{K} n_k$ is the total number of training data used on the selected clients.

4.2.2 Privacy-Preserving FL

Existing work in privacy-preserving FL can be classified into two categories: secure multi-party computation and differential privacy.

Secure multi-party computation (MPC) Existing work utilizes homomorphic encryption [90, 42, 18, 90] and secret sharing [101, 20, 13, 121] to preserve privacy in FL. With additive homomorphic encryption, for example, the Pallier cryptosystem, the server can perform gradient aggregation without decrypting them. Before training starts, the HE key pair is distributed to each client through a secure channel. In each training iteration, each client calculates the local model update, encrypts it with the public key, and uploads the ciphertext to the server. The server aggregates the encrypted gradients from all clients and sends the results back to the clients. Each client decrypts the received ciphertext using the private key to obtain global model updates due to additive homomorphism. But such algorithms are computationally expensive. FL systems with homomorphic encryption suffer from extremely high computational overhead and can hardly be applied on IoT devices. The researchers in [112] used secret sharing for secure aggregation in FL, allowing K parties to obtain the output of a function based on their K inputs while preventing any leakage of inputs other than the outputs. In [13], a noninteractive secure aggregation protocol based on secret sharing and key agreement was proposed, but a trusted authority was required. And scholars in [121] proposed a double masking

scheme that supports verification. The weakness of secret sharing lies in the communication cost. Each client needs to send a secret share to the majority of participating clients to guarantee the model robustness, or each pair of clients needs to communicate and agree on some random masks. Neither of them is applicable to IoT systems, in which devices have hardly direct communications.

Despite the high computational and communication overhead, such MPC approaches do not eliminate FL information leakage. In FL with homomorphic encryption, the server may collude with clients to decrypt local model updates from the ciphertext. As for secret sharing, the adversary still has a chance to infer the input information from the output of the function, since the function usually does not change.

Differential privacy (DP) Differential privacy [18, 29] is a noise perturbation mechanism that provides a statistical privacy guarantee for individual records. Existing work incorporates DP into FL from different perspectives. Shokri *et al.* [103] were the first scholars to integrate differential privacy in deep learning to protect training data privacy. NbAFL was proposed in [116] to protect uplink and downlink communication. In [120], 2DP-FL was proposed to handle non-i.i.d. distributions among clients and could adapt to different privacy needs. It has been empirically shown in [79] that DP is effective in defending against membership inference [112, 96], reconstruction [46] and model inversion [34] attacks. However, privacy protection comes at the cost of model accuracy.

In addition, in DP, bounding the influence of a single client is necessary for both privacy and the utility of the model. The choice of the bounding threshold, i.e., the clipping bound, has decisive effects on both privacy and model utility, due to the fact that the clipping bound could introduce bias to model updates [24]. Existing work quantifies the bias in ℓ_∞ [91] and ℓ_2 [130]. Nissim *et al.* [87] used a calibrated noise according to smooth sensitivity, but requires additional knowledge and communication of the original model updates. Adaptive clipping bounds that utilize the statistics of model updates to track and predict its change were proposed in [1, 5], but such clipping bounds do not immediately react to the change in model updates,

which could still result in excessive noise injection. Moreover, none of the existing work investigated the impact of the direction of the additive random noise on the convergence of the model.

4.2.3 Privacy Attacks against FL.

We mainly discuss two privacy attacks in FL: the membership inference attack and the property inference attack.

Membership inference attack Shokri *et al.* [104] demonstrated that an adversary can infer whether or not a given data sample was presented in the training data by the difference in model responses. Specifically, a binary classifier, called a shadow model, is trained for each output class using the same machine learning algorithm. Each shadow model identifies the membership of data samples of the corresponding class by outputting the probabilities over the membership and nonmembership classes. Studies in [8, 47, 93, 30] demonstrated the leakage of membership in various areas. Studies in [43, 112, 86, 96] analyzed membership inference from the perspectives of generative models, transferability, the relationship with overfitting, and defenses, respectively.

Property inference attack The property inference attack was first proposed by Ateniese *et al.* [7] against Hidden Markov Models and Support Vector Machine classifiers. Ganju *et al.* in [36] designed a property inference attack on fully connected networks. The adversary trains *meta-classifier* to classify *target classifier* depending on whether or not it has the property. In [72, 115], a training label composition inference attack was proposed. The adversary could infer the composition of the training label of a client's private data by finding a label composition such that the synthesized model updates are close to the true model update as much as possible.

4.3 Problem Setup

4.3.1 Threat Model

We consider a potential threat of privacy inference attack during the learning process. Specifically, an adversary could infer information about clients' private data through the model information exchange between clients and the server. Our proposed method is designed to withstand two potential adversaries: the central server and eavesdroppers.

- **Honest-but-curious server.** We assume that the central server is honest-but-curious, meaning that the server follows the FL protocol, but may try to infer some private information from the clients' model updates.

- **Eavesdroppers.** We also consider a potential attack that an eavesdropper monitors the communication link between clients and the server. We assume the attacker has no access to clients' training data, but they can eavesdrop model updates from the communication between clients and the server, and infer private information about clients.

4.3.2 Design Goals

We aim to design a noise perturbation scheme that achieves the following goals:

- **Utility.** The scheme should not sacrifice the accuracy of the global model. In particular, the FL with the noise perturbation should be able to learn a global model that is as accurate as that of the non-private FL.

- **Dropout-resilience.** The method should handle client dropout due to communication or power failure. When dropout happens, the server should still be able to get a reliable aggregation of local model updates from the remaining clients. A limited number of client dropouts should not affect the final global model accuracy.

- **Privacy.** The FL with the scheme should be able to mitigate the inference of private information from the communication of model updates between clients and the server.

- **Efficiency.** The FL with the scheme should not require additional training rounds to achieve comparable nonprivate FL accuracy. In addition, the method should not incur additional computation and communication overhead, since we consider the application scenario, in which clients are small devices that suffer from limited computation resources and network connectivity.

4.4 Our Approach

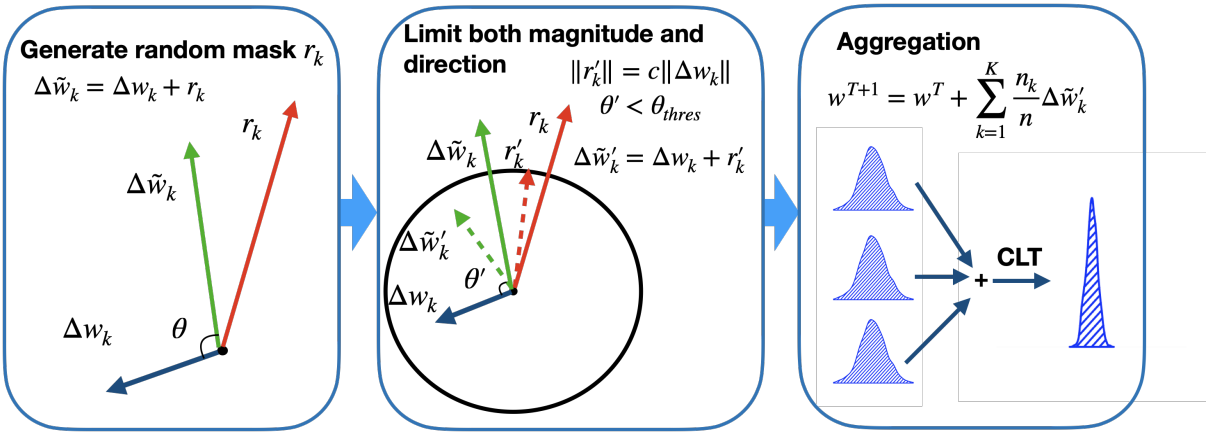


Figure 4.2: Geometric illustration of our proposed additive noise perturbation scheme.

4.4.1 Overview

In light of the drawbacks of DP discussed in Section 4.2.2, we introduce an adaptive noise scaling method and a direction-based filtering method in the additive noise perturbation scheme. In each iteration, our approach follows the three general steps of FL discussed in Section 4.2.1. Our approach is similar to FL with the DP scheme in [122, 120]. The difference lies in the second step. Instead of sending the original model updates, clients send the noise-perturbed local model updates to the central server, in which the noise is generated randomly and locally.

Our approach is different from the DP scheme in generating random noise. Specifically, a clipping bound is required in DP to limit the influence of a single client. The choice of the clipping bound could have a decisive impact on the utility and privacy of the model. A low clipping bound could destroy the direction of the gradients, weakening its strength in descent of the global model, whereas a high clipping bound might introduce too much noise to the

FL system, resulting in an accuracy degradation of the global model. Ideally, the clipping bound should be able to track the change of the norm of the model updates. But practically the behavior of the norms of model updates varies and is hard to predict. A popular method is to use the median of the norms of the unclipped local model updates over the course of training. However, the norm of model updates decreases along the training, whereas the clipping bound may not react as fast as the norm changes. This may introduce excessive noise to the global model, and this excessive noise could be the cause of the accuracy loss in the global model.

Furthermore, the direction of the gradients plays a significant role in both privacy and model utility aspects and is not considered in the DP scheme. On the one hand, there is plenty of privacy in the direction of the gradients. As indicated in [115], the presence of a given label class can be inferred by analyzing the signs of gradients. Therefore, the noise vector must be well chosen to hide the direction of gradients. On the other hand, large-scale noise could impair the accuracy or even destroy the convergence of the global model. Two noise vectors with the same magnitude could lead to opposite effects. To be specific, the one in the descent direction could be beneficial to the model convergence, while the other in the ascent direction could destroy the global model convergence.

Our approach is able to achieve a better convergence performance than DP due to the following three features:

- As will be shown in Section 4.5, setting the magnitude of additive noise to be proportional to the magnitude of local model updates ensures that the additive noise vanishes with local model updates when the FL model convergence occurs, preventing the FL model accuracy degradation.

- The scaling factor c chosen based on the number of participating clients ensures that the FL model enjoys the same convergence performance as a result of the cancelling out presented in the aggregation of noise on the server by the CLT. c can also be chosen to enable the ability to handle dropout clients.

- The proposed direction-based filtering scheme filters out noise vectors in bad directions, accelerating the convergence of the FL model.

Algorithm 5 Our CTL based FL privacy-preserving scheme

Input: K clients with local training datasets $D_k, k \in [K]$; client learning rate η; number of local iterations t; number of aggregations T; angular distance threshold θ_{thres}.
Output: Global model $\tilde{w}^T$.

Initialization global model weight to w^0.
for $T = 0 : T_{max}$ **do**
  The server synchronizes the latest global model to clients, $w_k^{T,0} \leftarrow \tilde{w}^T$.
  for $k = 1 : K$ **do**
    for $\tau = 0 : t - 1$ **do**
      The client updates the local weight by $w_k^{T,\tau+1} \leftarrow w_k^{T,\tau} - \eta \nabla F_k(w_k^{T,\tau})$
    end for
    while $\theta < \theta_{thres}$ **do**
      Generate new random noise r_k^T from $\mathcal{N}(0, I)$, and scale them by $\max(1, \frac{c\|\Delta w_k^T\|}{\|r_k^T\|})$, where $\Delta w_k^T = \sum_{\tau=0}^{t-1} \eta \nabla F_k(w_k^{T,\tau})$ and c is a scaler.
      Calculate the angular distance θ from the cosine similarity $\cos(\Delta w_k^{T,\tau}, \Delta w_k^{T,\tau} + r_k^T)$.
    end while
    Add the noise to the local model update, $\Delta \tilde{w}_k^{T,\tau} \leftarrow \Delta w_k^{T,\tau} + r_k^T$.
  end for
  The server aggregates the local model updates from clients, $\Delta \tilde{w}^{T+1} = \sum_{k=1}^{K} \frac{n_k}{n} \Delta \tilde{w}_k^T$, and update the global model $\tilde{w}^{T+1} \leftarrow w^T + \Delta \tilde{w}^{T+1}$.
end for

4.4.2 Our Additive Noise Scheme

Algorithm 5 details the steps in our proposed noise perturbation scheme, which consists of two key components: the adaptive noise scaling step and the direction-based noise filtering step.

Adaptive noise scaling

We introduce the steps to generate the proposed noise perturbation r_k, and how to determine the value of c in both dropout and non-dropout scenarios. After the client completes the local training, the noise r_k is randomly generated from $\mathcal{N}(0, I)$. Δw_k is denoted as local model updates. Then r_k is scaled by $\frac{c\|\Delta w_k\|}{\|r_k\|}$. The impact of c on model convergence will be theoretically analyzed and numerically evaluated in Sections 4.5 and 4.7, respectively.

Determine the value of c in a non-dropout scenario. As indicated in Theorem 2 and Theorem 3 (provided later in Section 4.5), setting the magnitude of additive noise in accordance with the magnitude of local model updates ensures the noise vanishes with the local updates when convergence occurs, avoiding accuracy degradation of the global model. Furthermore, as indicated in Theorem 1 (provided later in Section 4.5), the standard deviation

of the aggregated noise on the server is inversely proportional to the number of participating clients K, indicating that the effect of the scaling factor c will be counteracted by K when aggregated on the server. For convex optimization algorithms (e.g., gradient descent and proximal quasi-Newton), in which the loss function descends in every iteration, the magnitude of additive noise aggregation must not exceed the magnitude of model update aggregation, that is, $\| \sum_{k=1}^{K} r_k \| \leq \| \sum_{k=1}^{K} \Delta w_k \|$. Therefore, in a non-dropout scenario, K is a conservative upper bound for c, i.e., $c \leq K$. For optimization algorithms without monotonic requirement, e.g., SGD, the global model still converges as long as the descent of the global loss function is frequently achieved, indicating that c could be slightly greater than K ($c = K + \epsilon$, where $\epsilon \in \mathbb{N}^+$), which is denoted by $c <_\epsilon K$.

Determine the value of c in a dropout scenario. In a scenario with d client dropouts, the central server is expected to be able to get a reliable aggregation from the remaining $K - d$ clients. FL with our approach can tolerate at most d client dropouts by setting $c <_\epsilon (K - d)$, as previously indicated. Note that c controls the privacy protection strength on clients. Setting a large d results in a reduced c, which also reduces the strength of privacy protection for clients. When there are more than d client dropouts, the distribution of the noise aggregation becomes wider and there will be more noises falling in the tails of the distribution, which could cause the loss function to decrease less frequently (shown in Figure 4.3). In particular, when there is an extra dropout of clients, the standard deviation of noise aggregation increases slightly and becomes $\frac{K}{K-1}$ times the standard deviation of noise aggregation of K clients. Therefore, for a sufficiently large K, the impact of a small number of additional client dropouts is limited. There is still a great chance that the server can get a reliable aggregation from the remaining clients.

Direction-based noise filtering

Considering the noise scale alone is insufficient. To limit the negative impact on the accuracy of the FL model, we use cosine similarity to measure the angular distance between the true local model updates and the noise-perturbed local model updates. The client only adds a noise vector whose angular distance is less than the user-defined threshold θ_{thres}. A smaller θ_{thres}

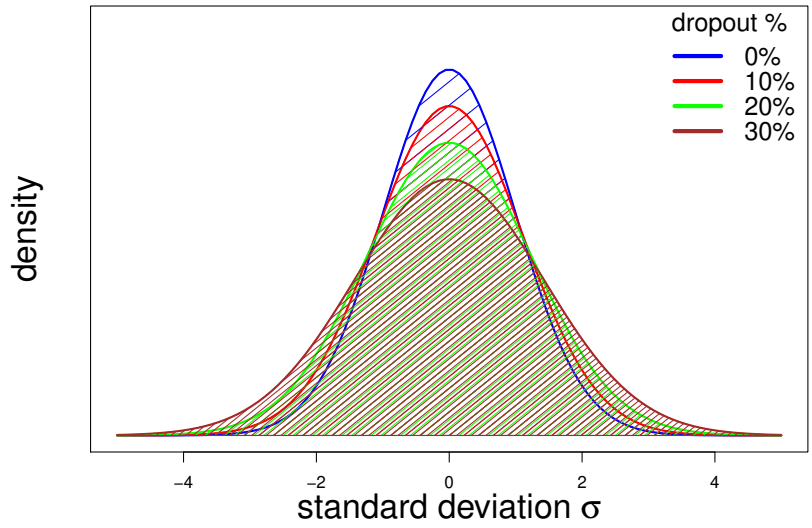


Figure 4.3: The distribution of aggregated noise with different dropout probabilities.

leads to a higher chance of global convergence, while a larger θ_{thres} provides better privacy protection.

Note that realistically the dimension of a neural network's parameter vector is usually extremely high. As illustrated in Figure 4.4, the angular distance between two arbitrary vectors is Gaussian distributed and becomes more concentrated as the dimension increases. Especially in an extremely high-dimensional space, such as the space of model updates, any two random vectors are orthogonal. Due to this observation, for a fixed θ_{thres}, it could be extremely computationally expensive or even impossible to find a satisfying noise vector in such a high-dimensional space. An intuitive way is to partition the model updates into smaller vectors and apply random noise individually. For convenience, we partition model updates by layers, and noises are generated and added to each layer separately. However, this could raise another problem that setting an absolute value of θ for all layers could be inappropriate. To align θ_{thres} in each layer, we use the three-sigma rule of thumb, setting $\theta_{thres} = \bar{\theta} + \rho\sigma_\theta$, where $\bar{\theta}$ and σ_θ are the mean and standard deviation of θ, respectively, and ρ is the multiple of σ_θ. $\bar{\theta}$ and σ_θ are only related to the dimension of vectors and can be pre-calculated, so this operation does not increase the computation cost. More importantly, this transforms the choice of an absolute

value of θ_{thres} into a relative value ρ, in which θ_{thres} is self-adjusted by the dimension of each layer.

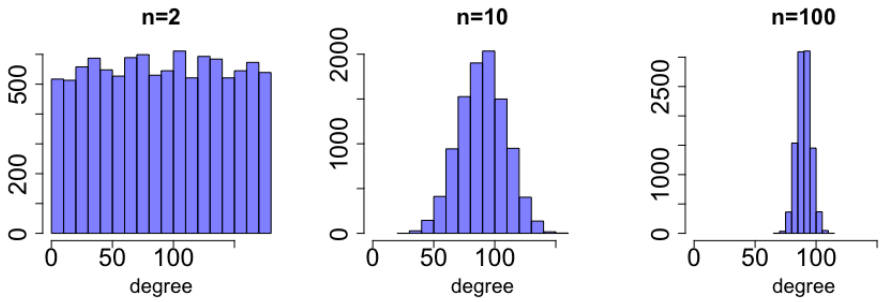


Figure 4.4: Histogram of the angular distance (in degree) between two arbitrary vectors in 2, 10 and 100 dimensional spaces, respectively (based on 10,000 samples).

The use of a larger c should combine with a small ρ to accelerate FL convergence. However, a smaller ρ increases the similarity between the noise-perturbed model updates and the original model updates, resulting in less privacy. Also, it could take more time to find a satisfying noise vector for a smaller ρ. Therefore, ρ should be chosen combining privacy requirements according to applications, as well as the choice of c. The numerical results of choosing different settings for ρ will be presented in Section 4.7.

4.5 Theoretical Analysis of Our Approach

In this section, we study the convergence performance of the proposed perturbation scheme for both convex and non-convex loss functions. The proofs show that FL with our proposed perturbation scheme can achieve the same global model convergence rate and accuracy as that of a regular FL in the convex case, and the same convergence rate as that of a regular FL in the non-convex case.

4.5.1 Assumptions

Denote the optimal value for $F(\cdot)$ by F^*, and the optimal value for $F_k(\cdot)$ by F_k^*. Define Γ as a measurement of non-i.i.d.-ness across clients: $\Gamma \triangleq \sum_{k=1}^{K} \frac{n_k}{n} F_k^* - F^*$, where $\Gamma \geq 0$ indicates how non-i.i.d. across the client's data. Note that given a large enough number of data samples on clients, we have $\Gamma \to 0$ for i.i.d. data distributions.

Four common assumptions are considered to facilitate the theoretical analyses of our proposed noise perturbation scheme.

Assumption 1. *The loss functions $F_k(\cdot)$ for $k \in [K]$ are all L-smooth; that is, $\forall v, w \in \mathbb{R}^d$,*

$$F_k(v) - F_k(w) \leq \langle v - w, \nabla F_k(w) \rangle + \frac{L}{2}\|v - w\|^2, \forall k \in [K]. \tag{4.5}$$

Assumption 2. *The loss functions $F_k(\cdot)$ for $k \in [K]$ are all μ-strongly convex; that is, $\forall v, w \in \mathbb{R}^d$,*

$$F_k(v) - F_k(w) \geq \langle v - w, \nabla F_k(w) \rangle + \frac{\mu}{2}\|v - w\|^2, \forall k \in [K]. \tag{4.6}$$

Assumption 3. *The expectation of the squared ℓ_2 norm of the stochastic gradients is bounded; that is,*

$$\mathbb{E}_\xi \left[\|\nabla F_k(w_k^{T,\tau}, \xi_k^{T,\tau})\|^2 \right] \leq G^2, \ \forall \tau \in [t], |\forall k \in [K]. \tag{4.7}$$

Assumption 4. *For the mini-batch $\xi_k^{T,\tau}$, we have the following.*

$$\mathbb{E}_\xi [\nabla F_k(w_k^{T,\tau}, \xi_k^{T,\tau})] = \nabla F_k(w_k^{T,\tau}), \tag{4.8}$$

where $\mathbb{E}_\xi$ denotes the expectation against the randomness of the stochastic gradient.

4.5.2 Convergence Analysis

We present the following theorems to show the theoretical convergence analyses of FedAvg with our proposed noise perturbation scheme. For simplicity of convergence analysis, we assume that there is no transmission error between the clients and the central server.

For ease of presentation, we denote the noise aggregated on the central server by $R = \sum_{k=1}^{K} \frac{n_k}{n} r_k$, and σ_k denotes the standard deviation of the local additive noise in each element of r_k. We have $\sigma_k \propto c$ and $\sigma_k \propto \Delta w_k$. For simplicity, we also assume that each client has the same amount of data, e.g., $\frac{n_k}{n} \approx \frac{1}{K}$.

Theorem 1. *For a sufficiently large K, each element in R follows the Gaussian distribution $\mathcal{N}(0, \sum_{k=1}^{K} \frac{(\sigma_k)^2}{K^2})$.*

Proof. See Appendix B.

Remark 3. *Theorem 1 reveals important properties about the number of participants K and the variance of noise aggregation. (1) The aggregation of additive noise can be characterized by a Gaussian distribution. (2) For sufficiently large K, that is, $K \geq 30$, the contribution of the noise on a single client to the variance of aggregation of noise is arbitrarily small. (3) A larger noise scale on the client will result in a greater variance in the aggregation of noise on the server.*

The Strongly Convex Case. We analyze the convergence property of our proposed noise perturbation scheme under strong convexity.

Theorem 2. *For a smooth and strongly convex objective function F_k, FedAvg satisfies*

$$\mathbb{E}\big[\ \|\tilde{w}^{T+1} - w^*\|^2\big] \leq A^T \mathbb{E}\big[\|\tilde{w}^0 - w^*\|^2\big] + \sum_{i=0}^{T-1} A^i B \tag{4.9a}$$

$$A = 2 - \mu\eta t + \mu\eta^2 t \tag{4.9b}$$

$$B = 2\eta t\Gamma + (1+2t)t\eta^2 G^2(1+\mu(1-\eta)) + \frac{t(t+1)(2t+1)\eta^2 G^2}{6} + \frac{9m^2}{K^2}\sum_{k=1}^{K}(\sigma_k^T)^2. \tag{4.9c}$$

Proof. See Appendix C.

Remark 4. *Since $\sigma_k^T \propto c$, we note that B is an increasing function of the noise scale c, while decreasing with the number of participants K. Furthermore, more non-i.i.d. local distributions between clients, resulting in higher Γ and G, will pose a negative impact on the convergence bound.*

Remark 5. *The FL converges iff $A < 1$, that is, $\eta \in \big[\frac{1-\sqrt{1-\frac{4}{\mu t}}}{2}, \frac{1+\sqrt{1-\frac{4}{\mu t}}}{2}\big]$. Let $\eta = \frac{1}{\sqrt{T}}$ for sufficiently large T and $\eta \in \big[\frac{1-\sqrt{1-\frac{4}{\mu t}}}{2}, \frac{1+\sqrt{1-\frac{4}{\mu t}}}{2}\big]$, the FL with our proposed scheme converges*

at a rate of $\mathcal{O}(1/T)$, which matches a typical SGD on strongly convex loss functions. In B, the noise-related term $\frac{9m^2}{K^2}\sum_{k=1}^{K}(\sigma_k^T)^2$ decreases as the FL model converges, since $\sigma_k \propto \|\Delta w_k\|$. When convergence occurs, where $\lim_{T\to\infty}\|\Delta w_k^T\| = 0$, we have $\lim_{T\to\infty}\frac{9m^2}{K^2}\sum_{k=1}^{K}(\sigma_k^T)^2 = 0$, which indicates that the proposed scheme converges to the same value as the regular FL scheme under strong convexity.

The Non-convex case. For more general cases, in which the objective function is not necessarily convex, convergence to global optima is not guaranteed, so we will only require convergence to a point of vanishing gradients. We prove the following theorem.

Theorem 3. *For a smooth and non-convex objective function F_k, FedAvg satisfies*

$$\min_{T\in[T_{max}]}\mathbb{E}\|\nabla F(\tilde{w}^t)\|^2 \leq \frac{2(F(w^0)-F(\tilde{w}^*))}{(1+\eta t-2\eta)T} + \frac{\eta^3 L^2 t(t+1)(2t+1)G^2}{6(1+\eta t-2\eta)} + \frac{m^2 L\sum_{k=1}^{K}(\sigma_k^T)^2}{K^2(1+\eta t-2\eta)} + \frac{c^2\eta^2 t^2 G^2}{1-\eta t-2\eta}. \tag{4.10}$$

Proof. See Appendix D.

Remark 6. *Let $\eta = \frac{1}{\sqrt{T}}$ for a sufficiently large T, Eq. 4.10 converges at a rate of $\mathcal{O}(1/\sqrt{T})$, which matches an SGD on non-convex loss functions. The noise-related term, $\frac{m^2 L\sum_{k=1}^{K}(\sigma_k^T)^2}{K^2(1+\eta t-2\eta)}$, decreases as the FL converges due to $\sigma_k \propto \|\Delta w_k\|$. Especially when convergence occurs, where $\lim_{T\to\infty}\|\Delta w_k\| = 0$, we have the noise-related term $\lim_{T\to\infty}\frac{m^2 L\sum_{k=1}^{K}(\sigma_k^T)^2}{K^2(1+\eta t-2\eta)} = 0$. Moreover, since $\sigma_k \propto c$, increasing the number of participating clients K or decreasing c will result in a faster convergence rate.*

4.6 Experiment Setup

4.6.1 Dataset

We evaluate our proposed methods on MNIST, a handwritten digit recognition dataset. The dataset contains 60,000 training data samples and 10,000 testing data samples. Each data sample is a square 28×28 pixel image of a single hand-written digit between 0 and 9.

4.6.2 Evaluation

We evaluate our proposed scheme from both model utility and privacy protection aspects. And we compare our approach with two baselines: (1) non-private FL, in which clients and servers follow standard FL protocol and do not involve any privacy-preserving mechanisms; (2) FL with local DP, in which clients add DP noise to protect the privacy of their local data. As stated in Section 4.3.2, our goal is to protect clients' local privacy against an honest-but-curious server and eavesdroppers, thus we only consider adding perturbations on the client's side. We compare our proposed scheme with the (ϵ, δ)-DP proposed in [1], which is widely used as a noise pattern on the client's side [116, 120]. Specifically, we use a popular choice of $\sigma = \sqrt{2 \log \frac{1.25}{\delta}}/\epsilon$ with a fixed δ of 10^{-5}. The clipping bound is set as the median of the norms of the unclipped local model updates over the course of training.

We evaluate the effectiveness by experimenting with FL with our approach and DP against two state-of-the-art FL privacy inference attacks that we have introduced in Section 4.2.2: the membership inference attack and the label composition inference attack. The convergence and security performance of our proposed perturbation scheme are evaluated using the following four metrics:

1. **Global model accuracy and convergence rate.** We measure the global model accuracy under different choices of parameters c and ρ as a function of the training epoch, and compare the convergence behavior in both dropout and non-dropout scenarios.

2. **Membership inference attack accuracy and F_1-score.** The attack accuracy is defined as the percentage of data samples that are correctly predicted to be presented in the training dataset. And the F_1 score combines precision and recall into a single value, which is defined as

$$F_1\text{-score} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

   A lower accuracy or F_1-score indicates a better protection of privacy.

3. **Accuracy of the label composition inference attack.** The accuracy is measured by the ℓ_2 difference between the true label composition and the inferred label composition. A larger difference indicates better privacy protection.

4. **Signal-to-noise ratio (SNR).** SNR is a popular metric for quantifying the relative amount of noise added to the data.

$$\text{SNR} = \frac{\text{variance of actual data}}{\text{variance of noise}}$$

A lower SNR indicates that there is a greater amount of noise being introduced into the system, leading to better privacy protection. Recovery of the original data becomes erroneous as the SNR drops below 1 [54]. It is also claimed in [70] that privacy can be achieved without affecting learning performance if a small SNR is consistently achieved.

4.6.3 FL System Settings

We implement FL and privacy inference attacks using the PyTorch framework. We conduct our experiments on Google Colab Pro (CPU: Intel(R) Xeon(R) CPU @ 2.20GHz; RAM: 13 GB; GPU: Tesla P100-PCIE-16GB with CUDA 11.2).

The dataset is allocated to 100 clients. The model on each client consists of two convolutional layers and two fully connected layers. In each global training epoch, K clients are randomly selected for the aggregation of the FL model. We use the Dirichlet distribution [82] with the hyper-parameter α to generate different data distributions across clients, in which a smaller α denotes a higher non-i.i.d. level. We set $\alpha = 1$ in the experiments of convergence and membership inference attack, and $\alpha = 0.1, 1, 10, \infty$ in the experiments of label composition inference attack.

For the convergence evaluation, we train the local model with a mini-batch gradient descent with batch size 128, internal epoch $t = 5$, and learning rate $\eta = 0.1$. Ten shadow models and an auxiliary dataset with $3,000$ samples are used in the membership inference attack. The training data composition inference attack is launched on local model updates with full-batch gradient descent. To fairly compare our approach with DP, we choose ϵ in DP so that the accuracy is comparable with that of our approach.

4.7 Experimental Results

Our approach achieves the security goals. Recall that we have four security goals (discussed in Section 4.3.2): utility, dropout-resilience, privacy, and efficiency. Our results show that our approach achieves the four goals.

4.7.1 Utility

The model utility is evaluated in the scenario where there is no attack. We fix $\rho = 0$ and choose the scaling factor of our approach to be $c = 1, 3, 5, 10$, and $\epsilon = 15, 20, 30$ in DP and compute the values of the global model accuracy as a function of the global training epochs. We also include the non-private FL to serve as a baseline. As shown in Figure 4.5, the trend and final accuracy of our approach is similar to that of the non-private FL. For all chosen c, the global model converges to same accuracy as the non-private FL. Such results are inline with **Remark 5**. Even for a large c (e.g., $c = 10$ means the magnitude of the additive noise is 10 times the magnitude of original model updates), the accuracy curve suffers from slight fluctuations, and still achieves the same value as the non-private FL does. As the value of c increases, the convergence becomes slightly slower, due to a higher variance introduced to the global model. This is consistent with our finding in **Remark 3**. We also plot the global model accuracy w.r.t. ρ, shown in Figure 4.6, where the global model converges to the same value but faster with a smaller ρ.

Compared with our approach, DP has a different convergence trend, in which the convergence is notably slower, and it takes much more epochs to reach an accuracy comparable to our approach. FL with our approach converges at epoch 5, whereas DP starts to converge at epoch 10 and the accuracy finally reaches a comparable accuracy at epoch 50 by $\epsilon = 30$.

The final global model accuracy of FL with our approach and DP are presented in Table 4.1. It is suggested that the training accuracy only drops around 1% as we increase c from 5 to 15 in our approach. It is also indicated that $\epsilon = 30$ is a minimum privacy budget in order for the DP to achieve an accuracy comparable to that of $c = 15$ in our approach, since $\epsilon = 20$ reported in the table results in a dropped model accuracy. For fairness, we compare under the setting,

in which a comparable global model accuracy (96%) is achieved by our approach ($c = 15$) and DP ($\epsilon = 30$).

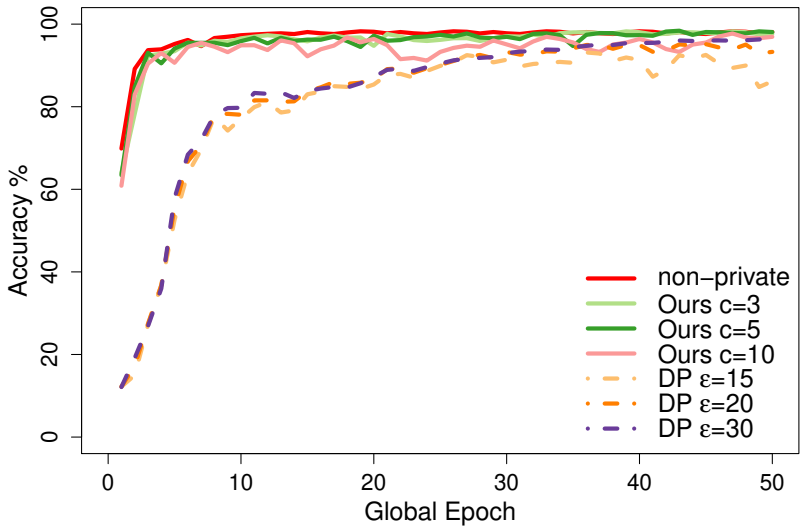


Figure 4.5: The comparison of global model accuracy among the non-private FL, FL with our approach and FL with DP.

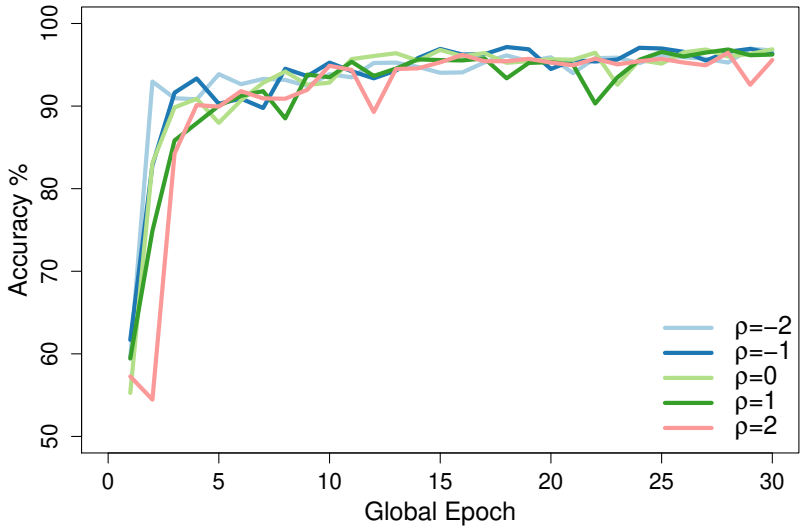


Figure 4.6: The global model accuracy w.r.t. ρ.

Table 4.1: Global model accuracy of FL with our approach , FL with DP and non-private FL.

	Non-private	**DP**			**Our approach**		
		$\epsilon = 15$	$\epsilon = 20$	$\epsilon = 30$	$c = 5$	$c = 10$	$c = 15$
Accuracy (%)	98.26	86.15	93.28	96.56	98.08	97.1	96.92

4.7.2 Dropout-resilience

In Section 4.5, we have shown that our approach can handle up to d client dropouts by setting $c \propto (K - d)$. So in this scenario, the convergence performance is similar to that in the non-dropout scenario where we have $c \propto K$. We also investigate the convergence performance when there are additional client dropouts. In particular, each client has a dropout probability from 0% (non-dropout) to 40%. And we set $c = 15$ in our approach and $\epsilon = 30$ in DP. When dropout occurs, the server will experience an increased variance of the aggregated noises, which might impair the global model's convergence and accuracy. As shown in Figure 4.7, as the dropout probability increases from 10% to 40%, the global model convergence rate and the accuracy of our approach remain similar to that of the non-dropout case. Our theoretical findings in **Remark 4** are consistent with such experimental results. Reducing a limited number of participating clients does not affect the global model accuracy but only results in slightly slower convergence. As for DP, both global model convergence rate and accuracy are severely impacted. Therefore, our approach handles up to d client dropouts by setting $c \leq_{\epsilon} (K - d)$ and the convergence performance of the global model is stable even with additional client dropouts.

4.7.3 Privacy

Defending against Membership Inference Attack

We continue to use the setting of $c = 10, 15$ in our approach and $\epsilon = 20, 30$ in DP. Figure 4.8 shows the per-class attack accuracy and F_1-score of the membership inference attack against FL with our approach, FL with DP as well as the non-private FL. As expected, the non-private FL leaks a considerable amount of information about the training dataset, resulting in an attack success rate as high as 87% on average. Both DP and our approach can reduce the attack accuracy and F_1-score against the membership inference attack. There is no significant difference in the attack accuracy. As for the F_1-score, authors in [96] set the baseline F_1-score to be 0.67

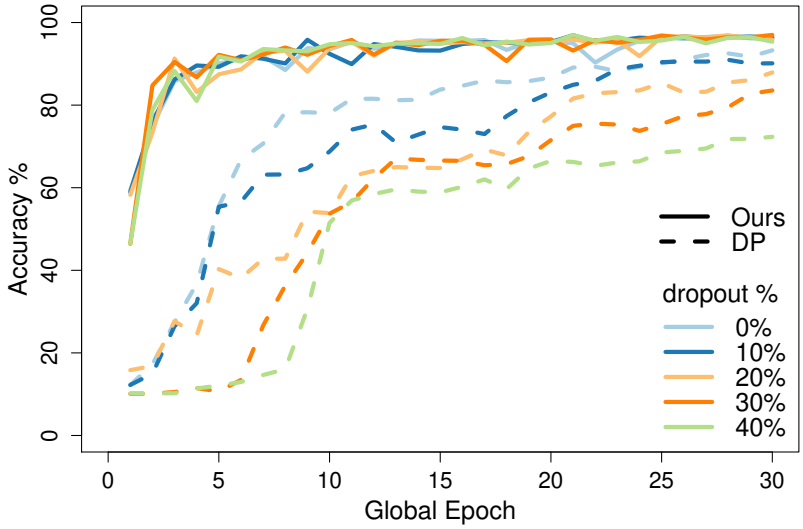
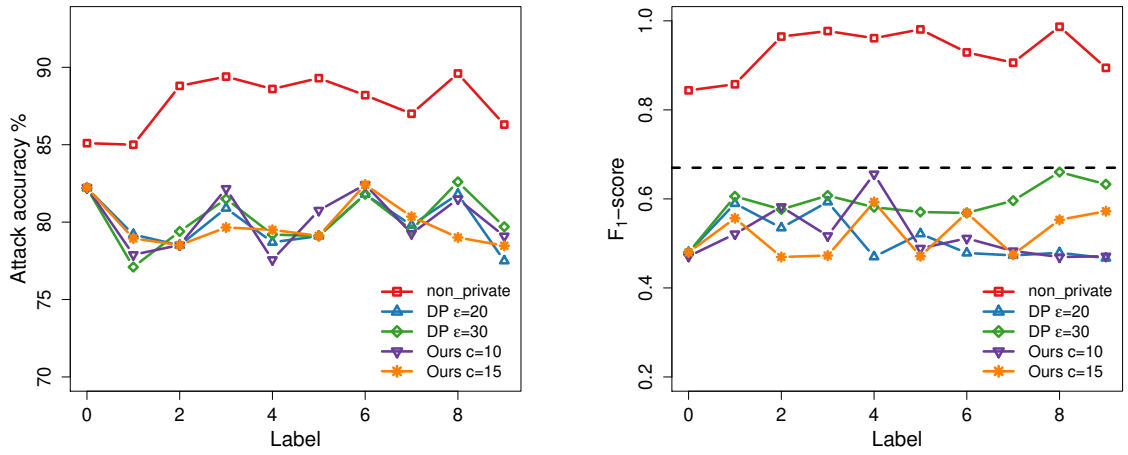


Figure 4.7: The global model accuracy w.r.t. dropout probability.



(a) Attack accuracy.

(b) Attack F_1-score.

Figure 4.8: Per-class accuracy and F_1-score of the membership inference attack against FL with DP, FL with our approach, and the non-private FL.

(dotted line in Figure 4.8(b)), since there are equal number of members and non-members in the attack test dataset. The F_1-score of all private models are below the baseline. The F_1-score for DP with $\epsilon = 30$ presents a higher pattern, whereas there is no significant difference among the rest of privacy-preserving FL models.

In addition, Table 4.2 indicates that our approach with $c = 10, 15$ is as effective as DP with $\epsilon = 20$. Referring back to Table 4.1, we see that the global model accuracy of FL with

$\epsilon = 20$ in DP is 3% lower than FL with $c = 10, 15$ in our approach. Therefore, given the same strength to defend against the membership inference attack, FL with our approach achieves a higher global model accuracy.

Further, Table 4.3 provides the global model accuracy, attack accuracy and F_1-score for a fixed $c = 15$ and different value of ρ. It is suggested that increasing ρ results in a slightly dropped global model accuracy, but more privacy protection in terms of the F_1-score.

Table 4.2: Membership inference attack accuracy and F_1-score for DP and our approach with different settings of ϵ and c as well as the non-private model.

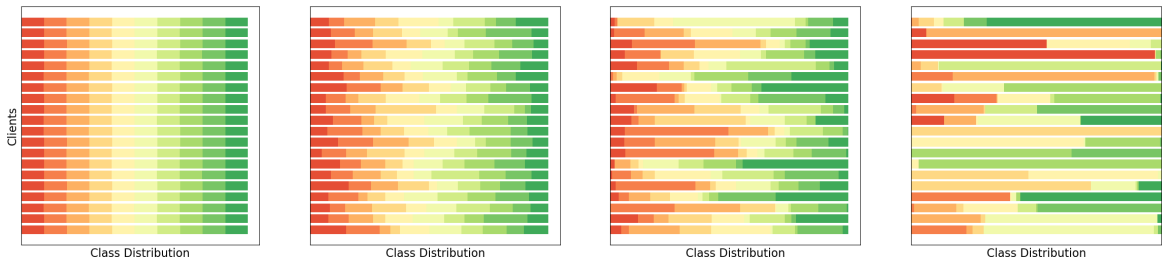
	Non-private	**DP**		**Our approach**		
		$\epsilon = 20$	$\epsilon = 30$	$c = 5$	$c = 10$	$c = 15$
Attack accuracy (%)	87.36	79.9	80.12	80.48	80.03	79.82
F_1-score	0.87	0.52	0.57	0.61	0.52	0.53

Table 4.3: Membership inference accuracy and F_1-score for DP and our approach with $c = 20$ and different value of ρ.

	Our approach				
	$\rho = 2$	$\rho = 1$	$\rho = 0$	$\rho = -1$	$\rho = -2$
Main accuracy (%)	94.76	95.26	96.92	97.56	98.10
Attack accuracy (%)	79.12	79.75	79.82	78.84	79.72
F_1-score	0.49	0.50	0.53	0.52	0.53

Defending against Label Composition Inference Attack

To compare privacy protection in different local label composition scenarios, we consider four local distribution settings, including one i.i.d. ($\alpha = \infty$) and three non-i.i.d. local distribution settings ($\alpha = 10, 1, 0.1$). Figure 4.9 visualizes the change in label composition w.r.t. α.



(a) $\alpha = \infty$ (b) $\alpha = 10$ (c) $\alpha = 1$ (d) $\alpha = 0.1$

Figure 4.9: Local label composition w.r.t. α.

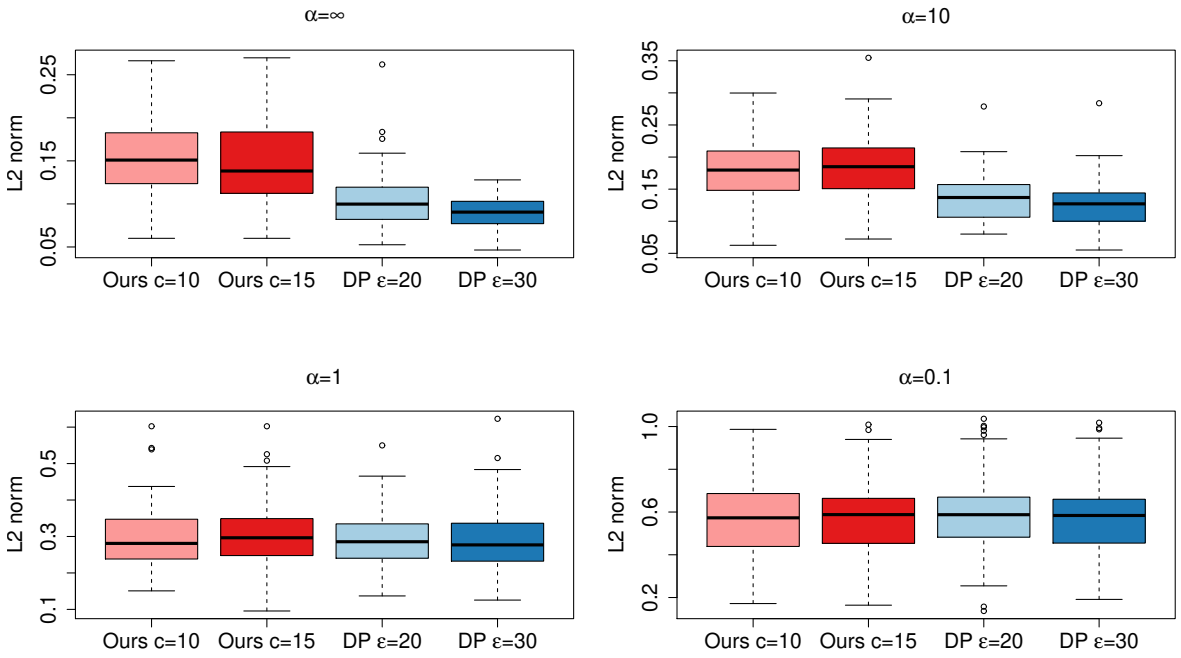


Figure 4.10: Box plot of the ℓ_2 distance between the original label composition and the inferred label composition by our approach and DP.

The results of the label composition inference attack are presented in Figure 4.10, which shows a box plot of ℓ_2 distance between the original label composition and the inferred label composition of FL with our approach and DP. Our approach is more effective in defending distribution inference attack compared to DP as the local distribution becomes more i.i.d ($\alpha = \infty, 10$), whereas our approach and DP reach a comparable protection as local distributions become more dissimilar ($\alpha = 1, 0.1$).

Signal-to-Noise Ratio (SNR)

Finally, we present the SNR of FL with our approach and DP as a function of the training epochs in Figure 4.11. Similarly as in previous experiments, the ϵ for DP and the c in our approach are chosen such that a similar global model accuracy is achieved. The results show that the SNR of DP is high at the beginning of the training and decreases as the convergence occurs, while our approach achieves a consistently low SNR. Referring to [70], such a consistently low SNR also explains our results in Section 4.7.1 that our approach has a minor impact on the global model's convergence and accuracy.

In addition, results in [54] showed that the original data could be harder to recover from a lower SNR. As shown in Figure 4.11, FL with DP has a higher chance to be recovered at the early training stage, due to their greater SNR values.

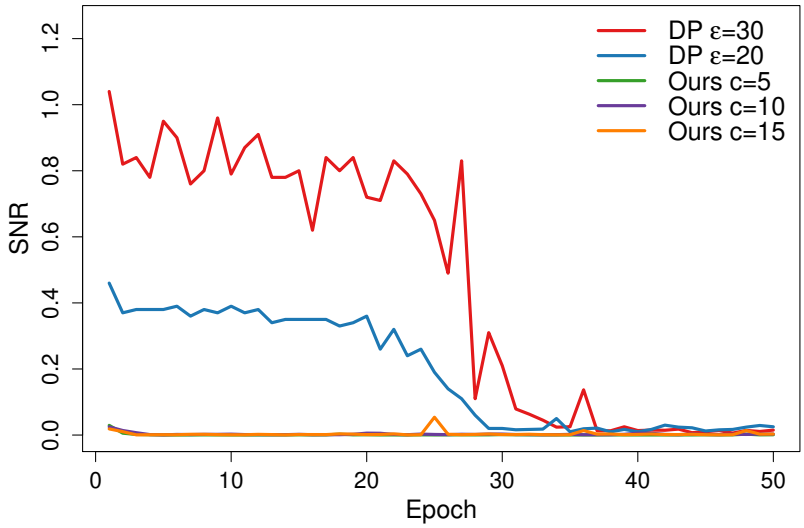


Figure 4.11: The SNR of our approach and DP along the training course.

4.7.4 Efficiency

We analyze the efficiency based on both the communication and computation overhead. FL with our approach converges as fast as the non-private FL, and much faster compared to FL with DP. Especially, for the MNIST task, both the FL with our approach and non-private FL converge at epoch 5, but FL with DP requires extra epochs to reach a similar global model accuracy, indicating that extra communication is needed for DP. Hence, the communication overhead of our proposed scheme is similar to the non-private FL, and much less than that of DP.

In addition, compared with the non-private FL, the only extra computation cost of our approach lies in the random noise generation, specifically in the direction-based filtering. Table 4.4 shows the time complexity analysis of our proposed noise perturbation scheme w.r.t. ρ in terms of multiples (m_ρ) of that of DP. The time complexity of our approach is inversely related to ρ. In DP, generating a noise vector for a vector of model updates with n parameters costs $\mathcal{O}(n)$. Therefore, the time complexity of our approach is $m_\rho \times \mathcal{O}(n)$. Since practically m_ρ

is much less than n, the time complexity of our proposed method is still $\mathcal{O}(n)$. The real time spent on generating the noise vector for one client's local updates w.r.t. ρ is presented in Table 4.5. The time shows an increasing pattern with a decreasing value of ρ. Even for a small ρ (e.g., $\rho = -3$), the time spent generating the noise vector is 0.52 seconds, which is minor compared with the local training time, which is 3 seconds in our experiments. Thus, we claim that our approach does not introduce extra communication and computation overhead.

Table 4.4: Time complexity of different ρ values in terms of multiples of that of DP.

Value of ρ	3	2	1	0	-1	-2	-3
Multiples of DP cost (m_ρ)	1.0	1.0	1.1	2.0	6.3	43.9	333.3

Table 4.5: Real time spent on noise vector generation w.r.t ρ.

	DP	**value of ρ**						
		3	2	1	0	-1	-2	-3
Time (s)	0.015	0.018	0.018	0.019	0.021	0.034	0.143	0.520

4.7.5 Generalization to More Complex Datasets

To explore if the above findings still hold for more complex datasets and neural network architectures, we conduct several experiments using ResNet 18 [44] on the CIFAR-10 [57] datasets. CIFAR-10 consists of 60,000 32×32 color images containing one of ten object classes, with 6000 images per class. ResNet 18 is a convolutional neural network that is 18 layers deep and contains around 11 million parameters.

Data are distributed to 50 clients with a non-i.i.d. parameter $\alpha = 10$ and 10 clients are selected in each training round. We use SGD with a learning rate of 0.1 and 200 training epochs. We compare FL with the proposed method ($c = 10$) with non-private FL and FL with DP ($\epsilon = 100$ and $\delta = 10^{-5}$). We report the training accuracy, the attack accuracy and F_1-score of the membership inference attack, and the accuracy of the label composition inference attack. These experiments are representative in verifying the impact of our proposed method on FL convergence and accuracy and the privacy protection against state-of-the-art privacy inference attacks.

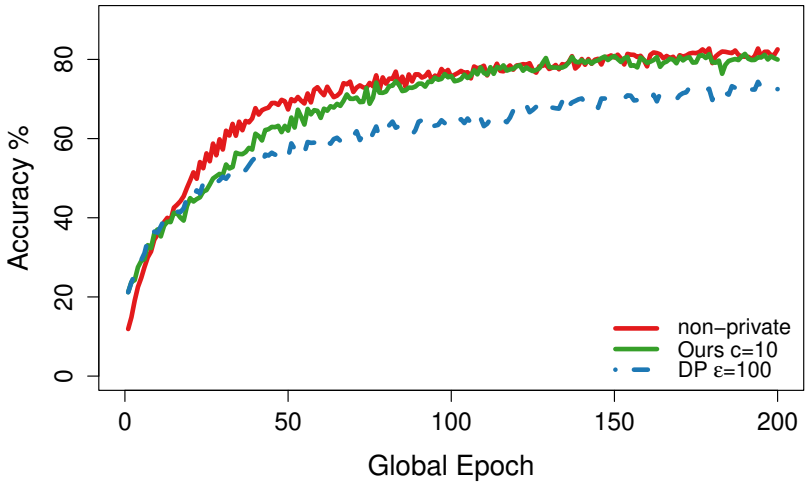


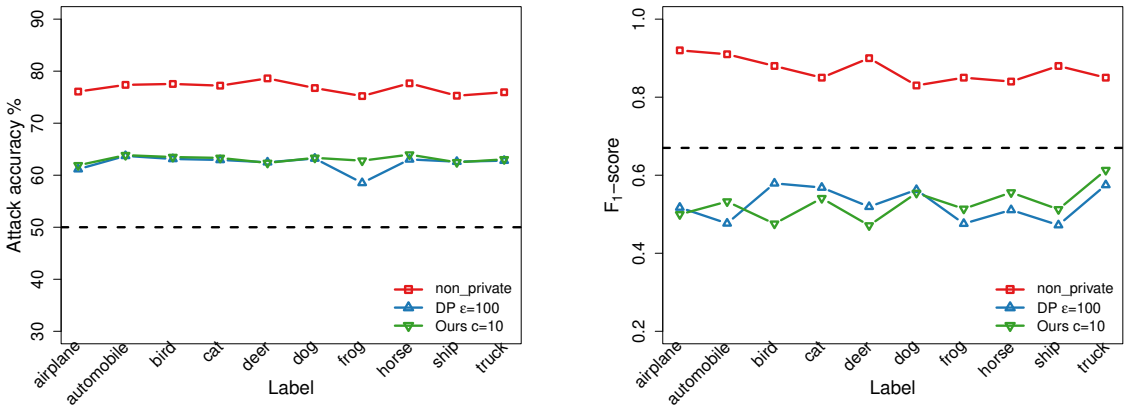
Figure 4.12: Global model accuracy of non-private FL, FL with DP, FL with our method on CIFAR-10, respectively.

The FL model accuracies are presented in Figure 4.12. The FL with the proposed method converges slightly slower than the non-private FL, but still converges to a similar accuracy of 80% around epoch 100. The slower convergence rate is due to higher non-convexity in the ResNet 18 model, which is consistent with the convergence analysis of the non-convex case (**Remark 5**). For FL with DP, even for a large ϵ of 100, the FL model still suffers accuracy loss and can only reach an accuracy of 74%.

Table 4.6: FL model accuracy, overall attack accuracy and F_1-score, and mean ℓ_2 distance on CIFAR-10.

	Non-private	**Ours** $c = 10$	**DP** $\epsilon = 100$
Model accuracy (%)	82.54	81.16	74.3
Attack accuracy (%)	76.77	63.06	62.36
Attack F_1-score	0.871	0.527	0.526
Attack ℓ_2 distance	0.023	0.101	0.099

We continue to evaluate the effectiveness of privacy protection on CIFAR-10. Table 4.6 summarizes the FL model accuracy and overall attack accuracy and F_1-score against the membership inference attack, as well as the ℓ_2 distance against the label composition inference attack. More specifically, Figure 4.13 provides the per-class attack accuracy and F_1-score. Figure 4.14 presents the results for the label composition inference attack, which shows a box plot of ℓ_2 distance of the true label composition and the inferred ones. Compared to non-private FL, both DP and our method can significantly lower the strength of two attacks, since the accuracy



(a) Attack accuracy.

(b) Attack F_1-score.

Figure 4.13: Per-class attack accuracy and F_1-score of the membership inference attack against FL with DP, FL with our approach, and regular FL on CIFAR-10. The dotted lines are baselines, where there is no privacy-preserving mechanism.

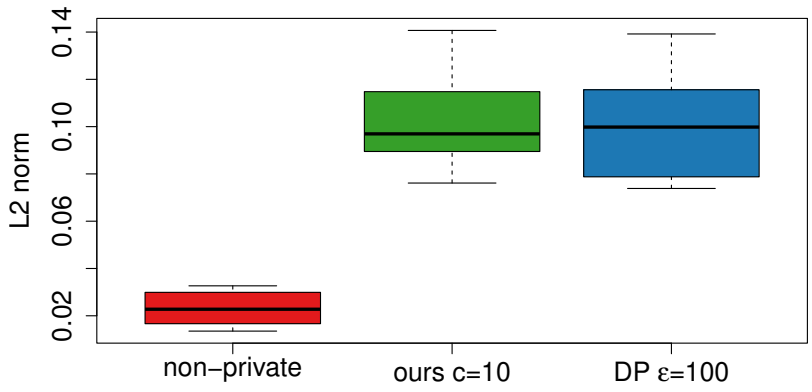


Figure 4.14: Box plot of ℓ_2 distance between the true and inferred label composition on CIFAR-10 with non-i.i.d. $\alpha = 10$.

of the attack, F_1 score, and the ℓ_2 distance are reduced by 17% percent, 0.3 and 0.078, respectively. There is no significant difference between our method and DP in both per-class attack accuracy and attack F_1-score, as well as the attack ℓ_2 norm. However, the gain in privacy protection by DP comes at the cost of 8% model accuracy loss, while our method enjoys a lossless accuracy.

4.8 Conclusion

In this work, we have proposed a novel adaptive perturbation-based scheme that protects local privacy in FL but without sacrificing global model accuracy. The key difference between our

approach and differential privacy is that we considered both the magnitude and direction when generating the random noises. In particular, we introduced adaptive noise scaling and direction-based filtering methods to reduce the negative impact of noises on the global model. We have provided theoretical convergence analyses of our proposed scheme with both non-convex and convex FL loss functions. Numerical experiments on the MNIST dataset have shown that our approach can achieve a convergence performance that is comparable to the non-private FL. And our proposed noise perturbation scheme can achieve a comparable or, in many cases, stronger privacy protection than DP in defending against state-of-the-art membership inference attack and label composition inference attack.

Chapter 5

Future Work

This dissertation reflects the robustness and privacy issues of distributed learning systems, which have not been widely investigated in previous work. I hope that this work can draw more attention to design a safer and more efficient distributed learning system. This dissertation also opens new opportunities for future security analysis of distributed learning schemes. In particular, three different branches deserve further investigation.

Firstly, the first work demonstrates that noise-like false data injection to a set of electricity meters is effective in misleading the state estimation result and leaves some future questions: (1) Further exploration of a wider class of attack models. Our work assumes that the attacker has full knowledge of the neural network settings (white-box attack), yet there are more realistic attack types, e.g., the attacker has partial knowledge (gray-box attack) or no knowledge (black-box attack) about the state estimator settings. (2) Model robustness to scalability challenges. The proposed attack has been tested on a 30-bus system. However, realistic power grids may have more than thousands of lines. A real-world test system is needed to accommodate complicated attack scenarios. Since power grid data are time sensitive and must be analyzed in a real-time manner, the state estimation for a large grid will be performed locally or regionally. On the one hand, a local attack that circumvents detection could lead to cascading failures of the entire grid. On the other hand, when a false alarm is triggered, the collected data will be discarded, and the control center will be blind to the status of the grid. Therefore, how to coordinate the state estimation results from subareas and the scalability of defense mechanisms should be considered in designing a robust state estimator for large-scale power grids.

Secondly, current FL research communities are aware of the threats of backdoor attacks, privacy attacks, and are dedicating themselves to robustify the learning algorithms and improving privacy protection. However, interactions between privacy leakage and backdoor attacks have not yet been thoroughly investigated. We hope that our work could inspire future enhancements to FL against both attacks. (1) Optimization of trigger design. Current triggers are visible and designed in a heuristic way. How to optimize the trigger pattern, e.g., reduce the number of pixels, or make the trigger less visible, is still an important open question. Information leakage from the shared model could help optimize the trigger pattern to reach the goal of invisibility and generalization. (2) Effective defenses. Most of existing defense mechanisms are built upon statistic methods, which may falsely reject or reduce the weight for a model that is normal but trained from highly non-i.i.d. data, or fail to reject a backdoored model that is deliberately designed. Furthermore, current backdoor attacks still have the potential to bypass these defense mechanisms. More efforts should be made to analyze weaknesses and design more effective defenses to keep up with the fast pace of the development of backdoor attacks.

Lastly, FL combined with other privacy-preserving methods has made great progress in protecting data privacy. However, in the area of mobile edge computing, in which most devices are IoT devices, additional key challenges come from computational and power constraints. Furthermore, due to the heterogeneity of IoT devices, privacy constraints may differ between devices or even between data samples on a single device. There is still a gap between FL techniques and real applications. Future research should focus on reducing the computation and communication overhead, retaining the model accuracy, and enabling the capability to handle mixed privacy constraints.

References

[1] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, Vienna, Austria, 2016.

[2] M. Abbasi and C. Gagné. Robustness to adversarial examples through an ensemble of specialists. *arXiv preprint arXiv:1702.06856*, 2017.

[3] M. Abdel-Nasser, K. Mahmoud, and H. Kashef. A novel smart grid state estimation method based on neural networks. *IJIMAI*, 5(1):92–100, 2018.

[4] M. M. Amiri, D. Gündüz, S. R. Kulkarni, and H. Vincent Poor. Convergence of federated learning over a noisy downlink. *IEEE Transactions on Wireless Communications*, pages 1–1, 2021.

[5] G. Andrew, O. Thakkar, H. B. McMahan, and S. Ramaswamy. Differentially private learning with adaptive clipping. *arXiv preprint arXiv:1905.03871*, 2019.

[6] ANSI. ANSI C12.1-2008: American National Standard for Electric Meters: Code for Electricity Metering. 2008.

[7] G. Ateniese, L. V. Mancini, A. Spognardi, A. Villani, D. Vitali, and G. Felici. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *International Journal of Security and Networks*, 10(3):137–150, 2015.

[8] M. Backes, P. Berrang, M. Humbert, and P. Manoharan. Membership privacy in microrna-based studies. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 319–330, Vienna, Austria, 2016.

[9] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov. How to backdoor federated learning. *arXiv preprint arXiv:1807.00459*, 2018.

[10] T. Barnett, S. Jain, U. Andra, and T. Khurana. Cisco visual networking index (vni) complete forecast update, 2017–2022. *Americas/EMEAR Cisco Knowledge Network (CKN) Presentation*, pages 1–30, 2018.

[11] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[12] R. B. Bobba, K. M. Rogers, Q. Wang, H. Khurana, K. Nahrstedt, and T. J. Overbye. Detecting false data injection attacks on dc state estimation. In *Preprints of the First Workshop on Secure Control Systems, CPSWEEK*, volume 2010, 2010.

[13] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, Dallas, USA, 2017.

[14] J. Bradshaw, A. G. d. G. Matthews, and Z. Ghahramani. Adversarial examples, uncertainty, and transfer testing robustness in gaussian process hybrid deep networks. *arXiv preprint arXiv:1707.02476*, 2017.

[15] T. S. Brisimi, R. Chen, T. Mela, A. Olshevsky, I. C. Paschalidis, and W. Shi. Federated learning of predictive models from federated electronic health records. *International Journal of Medical Informatics*, 112:59–67, 2018.

[16] R. H. Byrd, G. M. Chin, J. Nocedal, and Y. Wu. Sample size selection in optimization methods for machine learning. *Mathematical programming*, 134(1):127–155, 2012.

[17] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57, 2017.

[18] H. Chabanne, A. de Wargny, J. Milgram, C. Morel, and E. Prouff. Privacy-preserving classification on deep neural network. Cryptology ePrint Archive, Report 2017/035, 2017.

[19] Y. Chakhchoukh, H. Lei, and B. K. Johnson. Diagnosis of outliers and cyber attacks in dynamic pmu-based power state estimation. *IEEE Transactions on Power Systems*, 35(2):1188–1197, 2020.

[20] D. Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of Cryptology*, 1(1):65–75, 1988.

[21] L. Che, X. Liu, Z. Li, and Y. Wen. False data injection attacks induced sequential outages in power systems. *IEEE Transactions on Power Systems*, 34(2):1513–1523, 2019.

[22] M. Chen, R. Mathews, T. Ouyang, and F. Beaufays. Federated learning of out-of-vocabulary words. *arXiv preprint arXiv:1903.10635*, 2019.

[23] P. Chen, S. Cheng, and K. Chen. Smart attacks in smart grid communication networks. *IEEE Communications Magazine*, 50(8):24–29, 2012.

[24] X. Chen, S. Z. Wu, and M. Hong. Understanding gradient clipping in private sgd: A geometric perspective. In *Advances in Neural Information Processing Systems*, volume 33, pages 13773–13782, 2020.

[25] Y. Chen, Y. Tan, and D. Deka. Is machine learning in power systems vulnerable? In *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, pages 1–6, 2018.

[26] Y. Chen, Y. Tan, and B. Zhang. Exploiting vulnerabilities of load forecasting through adversarial attacks. In *Proceedings of the Tenth ACM International Conference on Future Energy Systems*, pages 1–11, 2019.

[27] K. Deb. An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering*, 186(2):311–338, 2000.

[28] C. Dwork and M. Naor. On the difficulties of disclosure prevention in statistical databases or the case for differential privacy. *Journal of Privacy and Confidentiality*, 2(1), 2010.

[29] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Foundations Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[30] C. Dwork, A. Smith, T. Steinke, J. Ullman, and S. Vadhan. Robust traceability from trace amounts. In *Proceedings of 2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 650–669, 2015.

[31] E. M. El Mhamdi, R. Guerraoui, and S. Rouault. The hidden vulnerability of distributed learning in Byzantium. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3521–3530. PMLR, 10–15 Jul 2018.

[32] M. Esmalifalak, L. Liu, N. Nguyen, R. Zheng, and Z. Han. Detecting stealthy false data injection using machine learning in smart grid. *IEEE Systems Journal*, 11(3):1644–1652, 2017.

[33] M. Esmalifalak, H. Nguyen, R. Zheng, and Zhu Han. Stealth false data injection using independent component analysis in smart grid. In *2011 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 244–248, 2011.

[34] M. Fredrikson, S. Jha, and T. Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333, Denver, USA, 2015.

[35] C. Fung, C. J. Yoon, and I. Beschastnikh. Mitigating sybils in federated learning poisoning. *arXiv preprint arXiv:1808.04866*, 2018.

[36] K. Ganju, Q. Wang, W. Yang, C. A. Gunter, and N. Borisov. 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*, CCS '18, pages 619–633, New York, NY, USA, 2018. Association for Computing Machinery.

[37] R. C. Geyer, T. Klein, and M. Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.

[38] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[39] Z. Guo, D. Shi, D. E. Quevedo, and L. Shi. Secure state estimation against integrity attacks: A gaussian mixture model approach. *IEEE Transactions on Signal Processing*, 67(1):194–207, 2019.

[40] X. Han, H. Yu, and H. Gu. Visual inspection with federated learning. In F. Karray, A. Campilho, and A. Yu, editors, *Image Analysis and Recognition*, pages 52–64, Cham, 2019. Springer International Publishing.

[41] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

[42] S. Hardy, W. Henecka, H. Ivey-Law, R. Nock, G. Patrini, G. Smith, and B. Thorne. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. *arXiv preprint arXiv:1711.10677*, 2017.

[43] J. Hayes, L. Melis, G. Danezis, and E. De Cristofaro. Logan: Membership inference attacks against generative models. In *Proceedings of the Privacy Enhancing Technologies*, pages 133–152, Barcelona, Spain, 2019.

[44] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[45] Y. He, G. J. Mendis, and J. Wei. Real-time detection of false data injection attacks in smart grid: A deep learning-based intelligent mechanism. *IEEE Transactions on Smart Grid*, 8(5):2505–2516, 2017.

[46] B. Hitaj, G. Ateniese, and F. Perez-Cruz. Deep models under the gan: Information leakage from collaborative deep learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 603–618, New York, NY, USA, 2017. Association for Computing Machinery.

[47] N. Homer, S. Szelinger, M. Redman, D. Duggan, W. Tembe, J. Muehling, J. V. Pearson, D. A. Stephan, S. F. Nelson, and D. W. Craig. Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays. *PLoS Genetics*, 4(8):e1000167, 2008.

[48] R. Huang, B. Xu, D. Schuurmans, and C. Szepesvári. Learning with a strong adversary. *arXiv preprint arXiv:1511.03034*, 2015.

[49] Y. Huang, H. Li, K. A. Campbell, and Z. Han. Defending false data injection attack on smart grid network using adaptive cusum test. In *2011 45th Annual Conference on Information Sciences and Systems*, pages 1–6. IEEE, 2011.

[50] G. Hug and J. A. Giampapa. Vulnerability assessment of ac state estimation with respect to false data injection cyber-attacks. *IEEE Transactions on Smart Grid*, 3(3):1362–1370, 2012.

[51] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, and B. Li. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 19–35, 2018.

[52] A. Jain, R. Balasubramanian, and S. C. Tripathy. Topological observability: Artificial neural network application based solution for a practical power system. In *2008 40th North American Power Symposium*, pages 1–6, 2008.

[53] L. Jia, R. J. Thomas, and L. Tong. On the nonlinearity effects on malicious data attack on power system. In *2012 IEEE Power and Energy Society General Meeting*, pages 1–8, 2012.

[54] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar. On the privacy preserving properties of random data perturbation techniques. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, pages 99–106, 2003.

[55] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[56] D. Kraft. A software package for sequential quadratic programming. *Forschungsbericht Deutsche Forschungs und Versuchsanstalt für Luft und Raumfahrt*, 88:33, 1988.

[57] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[58] D. V. Kumar, S. Srivastava, S. Shah, and S. Mathur. Topology processing and static state estimation using artificial neural networks. *IEE Proceedings-Generation, Transmission and Distribution*, 143(1):99–105, 1996.

[59] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[60] Y. LeCun, C. Cortes, and C. Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[61] R. Lee, M. Assante, and T. Conway. Analysis of the cyber attack on the ukrainian power grid. 2016.

[62] B. Li, R. Lu, W. Wang, and K.-K. R. Choo. Distributed host-based collaborative detection for false data injection attacks in smart grid cyber-physical system. *Journal of Parallel and Distributed Computing*, 103:32–41, 2017.

[63] J. Li, J. Y. Lee, Y. Yang, J. S. Sun, and K. Tomsovic. Conaml: Constrained adversarial machine learning for cyber-physical systems. *arXiv preprint arXiv:2003.05631*, 2020.

[64] S. Li, Y. Cheng, W. Wang, Y. Liu, and T. Chen. Learning to detect malicious clients for robust federated learning. *arXiv preprint arXiv:2002.00211*, 2020.

[65] S. Li, Y. Yılmaz, and X. Wang. Quickest detection of false data injection attack in wide-area smart grids. *IEEE Transactions on Smart Grid*, 6(6):2725–2735, 2015.

[66] G. Liang, S. R. Weller, J. Zhao, F. Luo, and Z. Y. Dong. The 2015 ukraine blackout: Implications for false data injection attacks. *IEEE Transactions on Power Systems*, 32(4):3317–3318, 2017.

[67] J. Liang, O. Kosut, and L. Sankar. Cyber attacks on ac state estimation: Unobservability and physical consequences. In *2014 IEEE PES General Meeting | Conference Exposition*, pages 1–5, 2014.

[68] J. Liang, L. Sankar, and O. Kosut. Vulnerability analysis and consequences of false data injection attack on power system state estimation. *IEEE Transactions on Power Systems*, 31(5):3864–3872, 2016.

[69] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao. A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications. *IEEE Internet of Things Journal*, 4(5):1125–1142, 2017.

[70] D. Liu and O. Simeone. Privacy for free: Wireless federated learning via uncoded transmission with adaptive power control. *IEEE Journal on Selected Areas in Communications*, 39(1):170–185, 2021.

[71] L. Liu, M. Esmalifalak, Q. Ding, V. A. Emesih, and Z. Han. Detecting false data injection attacks on power grid by sparse optimization. *IEEE Transactions on Smart Grid*, 5(2):612–621, 2014.

[72] T. Liu, X. Hu, and T. Shu. Assisting backdoor federated learning with global knowledge alignment. https://drive.google.com/file/d/1L3694k1GXGnByfcREZUE0P-gwjzGkU3T/view?usp=sharing, 2021.

[73] Y. Liu, X. Chen, C. Liu, and D. Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.

[74] Y. Liu, P. Ning, and M. K. Reiter. False data injection attacks against state estimation in electric power grids. *ACM Transactions on Information and System Security (TISSEC)*, 14(1):13, 2011.

[75] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[76] K. Manandhar, X. Cao, F. Hu, and Y. Liu. Detection of faults and attacks including false data injection attack in smart grid using kalman filter. *IEEE Transactions on Control of Network Systems*, 1(4):370–379, 2014.

[77] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In A. Singh and J. Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 20–22 Apr 2017.

[78] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54, pages 1273–1282, Sydney, Australia, 20–22 Apr 2017.

[79] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 691–706, 2019.

[80] J.-H. Menke, N. Bornhorst, and M. Braun. Distribution system monitoring for smart power grids with distributed generation using artificial neural networks. *International Journal of Electrical Power & Energy Systems*, 113:472–480, 2019.

[81] K. R. Mestav, J. Luengo-Rozas, and L. Tong. State estimation for unobservable distribution systems via deep neural networks. In *2018 IEEE Power Energy Society General Meeting (PESGM)*, pages 1–5, 2018.

[82] T. Minka. Estimating a dirichlet distribution, 2000.

[83] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2574–2582, June 2016.

[84] H. Mosbah and M. El-Hawary. Multilayer artificial neural networks for real time power system state estimation. In *2015 IEEE Electrical Power and Energy Conference (EPEC)*, pages 344–351, 2015.

[85] M. Naseri, J. Hayes, and E. De Cristofaro. Local and central differential privacy for robustness and privacy in federated learning. *arXiv preprint arXiv:2009.03561*, 2020.

[86] M. Nasr, R. Shokri, and A. Houmansadr. 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.

[87] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, pages 75–84, San Diego, USA, 2007.

[88] A. Onwuachumba and M. Musavi. New reduced model approach for power system state estimation using artificial neural networks and principal component analysis. In *2014 IEEE Electrical Power and Energy Conference*, pages 15–20, 2014.

[89] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 582–597. IEEE, 2016.

[90] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security*, 13(5):1333–1345, 2018.

[91] V. Pichapati, A. T. Suresh, F. X. Yu, S. J. Reddi, and S. Kumar. Adaclip: Adaptive clipping for private sgd. *arXiv preprint arXiv:1908.07643*, 2019.

[92] K. Pillutla, S. M. Kakade, and Z. Harchaoui. Robust aggregation for federated learning. *arXiv preprint arXiv:1912.13445*, 2019.

[93] A. Pyrgelis, C. Troncoso, and E. De Cristofaro. Knock knock, who's there? membership inference on aggregate location data. *arXiv preprint arXiv:1708.06145*, 2017.

[94] B. Qolomany, K. Ahmad, A. Al-Fuqaha, and J. Qadir. Particle swarm optimized federated learning for industrial iot and smart city services. In *Proceeding of the 2020 IEEE Global Communications Conference*, pages 1–6, 2020.

[95] M. A. Rahman and H. Mohsenian-Rad. False data injection attacks against nonlinear state estimation in smart power grids. In *2013 IEEE Power Energy Society General Meeting*, pages 1–5, 2013.

[96] M. A. Rahman, T. Rahman, R. Laganière, N. Mohammed, and Y. Wang. Membership inference attack against differentially private deep learning model. *Transactions on Data Privacy*, 11(1):61–79, 2018.

[97] S. Ramaswamy, R. Mathews, K. Rao, and F. Beaufays. Federated learning for emoji prediction in a mobile keyboard. *arXiv preprint arXiv:1906.04329*, 2019.

[98] A. Rozsa, E. M. Rudd, and T. E. Boult. Adversarial diversity and hard positive generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 25–32, 2016.

[99] H. Sandberg, A. Teixeira, and K. H. Johansson. On security indices for state estimators in power networks. In *Preprints of the First Workshop on Secure Control Systems, CPSWEEK 2010, Stockholm, Sweden*, 2010. QC 20120206.

[100] H. Sedghi and E. Jonckheere. Statistical structure learning of smart grid for detection of false data injection. In *2013 IEEE Power Energy Society General Meeting*, pages 1–5, 2013.

[101] A. Shamir. How to share a secret. *Communication of the ACM*, 22(11):612–613, 1979.

[102] M. Shayan, C. Fung, C. J. Yoon, and I. Beschastnikh. Biscotti: A ledger for private and secure peer-to-peer machine learning. *arXiv preprint arXiv:1811.09904*, 2018.

[103] R. Shokri and V. Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1310–1321, Denver, Colorado, USA, 2015.

[104] R. Shokri, M. Stronati, C. Song, and V. Shmatikov. Membership inference attacks against machine learning models. In *Proceedings of the 2017 IEEE Symposium on Security and Privacy*, pages 3–18, San Jose, USA, 2017.

[105] D. Singh, J. P. Pandey, and D. S. Chauhan. Radial basis neural network state estimation of electric power networks. In *2004 IEEE International Conference on Electric Utility Deregulation, Restructuring and Power Technologies. Proceedings*, volume 1, pages 90–95 Vol.1, 2004.

[106] R. Storn and K. Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359, 1997.

[107] J. Su, D. V. Vargas, and K. Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, 2019.

[108] Z. Sun, P. Kairouz, A. T. Suresh, and H. B. McMahan. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963*, 2019.

[109] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

[110] A. Teixeira, G. Dán, H. Sandberg, and K. H. Johansson. A Cyber Security Study of a SCADA Energy Management System: Stealthy Deception Attacks on the State Estimator*. *IFAC Proceedings Volumes*, 44(1):11271–11277, 2011.

[111] F. Tramèr, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel. The space of transferable adversarial examples. *arXiv preprint arXiv:1704.03453*, 2017.

[112] S. Truex, L. Liu, M. E. Gursoy, L. Yu, and W. Wei. Demystifying membership inference attacks in machine learning as a service. *IEEE Transactions on Services Computing*, 14(6):2073–2089, 2021.

[113] H. Wang, K. Sreenivasan, S. Rajput, H. Vishwakarma, S. Agarwal, J.-y. Sohn, K. Lee, and D. Papailiopoulos. Attack of the tails: Yes, you really can backdoor federated learning. *arXiv preprint arXiv:2007.05084*, 2020.

[114] J. Wang, L. C. K. Hui, and S. M. Yiu. System-state-free false data injection attack for nonlinear state estimation in smart grid. *International Journal of Smart Grid and Clean Energy*, 4(3), 2015.

[115] L. Wang, S. Xu, X. Wang, and Q. Zhu. Eavesdrop the composition proportion of training labels in federated learning. *arXiv preprint arXiv:1910.06044*, 2019.

[116] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. S. Quek, and H. V. Poor. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, 15:3454–3469, 2020.

[117] A. J. Wood and B. F. Wollenberg. *Power generation, operation, and control Second edition*. John Wiley & Sons, Inc., 1996.

[118] C. Xie, K. Huang, P.-Y. Chen, and B. Li. Dba: Distributed backdoor attacks against federated learning. In *International Conference on Learning Representations*, 2020.

[119] C. Xie, Z. Zhang, Y. Zhou, S. Bai, J. Wang, Z. Ren, and A. L. Yuille. Improving transferability of adversarial examples with input diversity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[120] Z. Xiong, Z. Cai, D. Takabi, and W. Li. Privacy threat and defense for federated learning with non-i.i.d. data in aiot. *IEEE Transactions on Industrial Informatics*, 18(2):1310–1321, 2022.

[121] G. Xu, H. Li, S. Liu, K. Yang, and X. Lin. Verifynet: Secure and verifiable federated learning. *IEEE Transactions on Information Forensics and Security*, 15:911–926, 2020.

[122] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang. Federated learning for healthcare informatics. *Journal of Healthcare Informatics Research*, 5(1):1–19, 2021.

[123] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage, and F. Beaufays. Applied federated learning: Improving google keyboard query suggestions. *arXiv preprint arXiv:1812.02903*, 2018.

[124] D. Yin, Y. Chen, R. Kannan, and P. Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5650–5659. PMLR, 10–15 Jul 2018.

[125] J. J. Q. Yu, Y. Hou, and V. O. K. Li. Online false data injection attack detection with wavelet transform and deep neural networks. *IEEE Transactions on Industrial Informatics*, 14(7):3271–3280, 2018.

[126] X. Yuan, P. He, Q. Zhu, and X. Li. Adversarial examples: Attacks and defenses for deep learning. *IEEE Transactions on Neural Networks and Learning Systems*, 30(9):2805–2824, 2019.

[127] A. S. Zamzam, X. Fu, and N. D. Sidiropoulos. Data-driven learning-based optimization for distribution system state estimation. *IEEE Transactions on Power Systems*, 34(6):4796–4805, 2019.

[128] A. S. Zamzam and N. D. Sidiropoulos. Physics-aware neural networks for distribution system state estimation. *IEEE Transactions on Power Systems*, 35(6):4347–4356, 2020.

[129] F. Zhang, H. A. D. E. Kodituwakku, J. W. Hines, and J. Coble. Multilayer data-driven cyber-attack detection system for industrial control systems based on network, system, and process data. *IEEE Transactions on Industrial Informatics*, 15(7):4362–4369, 2019.

[130] J. Zhang, T. He, S. Sra, and A. Jadbabaie. Why gradient clipping accelerates training: A theoretical justification for adaptivity. In *International Conference on Learning Representations*, 2020.

[131] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

[132] L. Zhu, Z. Liu, and S. Han. Deep leakage from gradients. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 32, Vancouver, Canada, 2019.

[133] R. D. Zimmerman, C. E. Murillo-Sánchez, and R. J. Thomas. Matpower: Steady-state operations, planning, and analysis tools for power systems research and education. *IEEE Transactions on Power Systems*, 26(1):12–19, 2011.

Appendices

A Proof of Proposition 1

Proof.

$$
\begin{aligned}
&\|w_k^{T,t} - w_{cen}^{T,t}\| \\
=&\Big\|w_k^{T,t-1} - \eta \underbrace{\sum_{c=1}^{C} p_k(y=c)\nabla \mathbb{E}_{x\in D_k|y=c}[\log f_c(x; w_k^{T,t-1})]}_{A_1} \\
&- w_{cen}^{T,t-1} + \eta \underbrace{\sum_{c=1}^{C} p(y=c)\nabla \mathbb{E}_{x\in D|y=c}[\log f_c(x; w_{cen}^{T,t-1})]}_{A_2} \\
&- \eta \underbrace{\sum_{c=1}^{C} p_k(y=c)\nabla \mathbb{E}_{x\in D|y=c}[\log(f_c(x; w_{cen}^{T,t-1})]}_{A_3} + \eta \underbrace{\sum_{c=1}^{C} p_k(y=c)\nabla \mathbb{E}_{x\in D|y=c}[\log(f_c(x; w_{cen}^{T,t-1})]}_{A_3} \Big\| \\
\overset{(1)}{\leq}&\|w_k^{T,t-1} - w_{cen}^{T,t-1}\| + \eta\|A_1 - A_3\| + \eta\|A_2 - A_3\| \\
=&\|w_k^{T,t-1} - w_{cen}^{T,t-1}\| + \eta\Big\|\sum_{c=1}^{C} p_k(y=c)\big[\nabla \mathbb{E}_{x\in D_k|y=c}[\log f_c(x; w_k^{T,t-1})] - \nabla \mathbb{E}_{x\in D|y=c}[\log(f_c(x; w_{cen}^{T,t-1})]\big]\Big\| \\
&+ \eta\Big\|\sum_{c=1}^{C}\big[(p(y=c) - p_k(y=c)\big]\nabla \mathbb{E}_{x\in D|y=c}[\log(f_c(w_k^{T,t-1})]\Big\|, \qquad (5.1)
\end{aligned}
$$

where the inequality (1) holds due to the Cauchy–Schwarz inequality. By induction, we have:

$$
\begin{aligned}
&\|w_k^{T+1} - w_{cen}^{T+1}\| \\
\leq &\|w_k^T - w_{cen}^T\| + \eta \sum_{\tau=1}^{t} \Big\| \sum_{c=1}^{C} \big[(p(y=c) - p_k(y=c)\big] \nabla \mathbb{E}_{x \in D|y=c}[\log(f_c(w_k^{T,\tau-1})] \Big\| \\
&+ \eta \sum_{\tau=1}^{t} \Big\| \sum_{c=1}^{C} p_k(y=c) \big[\nabla \mathbb{E}_{x \in D_k|y=c}[\log f_c(x; w_k^{T,\tau-1})] - \nabla \mathbb{E}_{x \in D|y=c}[\log(f_c(x; w_{cen}^{T,\tau-1})] \big] \Big\|.
\end{aligned} \tag{5.2}
$$

Hence, Eq. 3.8 has been proved. And the proof of Eq. 3.9 follows similar steps, and hence we omit the proof here. □

B Proof of Theorem 1

Proof. Recall that in the process of generating random noise, r_k is first randomly chosen from $\mathcal{N}(0, I)$ and then scaled by $\frac{c\|\Delta w_k\|}{\|r_k\|}$. Therefore, the i -th element r_{ki} follows a Gaussian distribution $\mathcal{N}(0, \sigma_k^2)$ with $\sigma_k = \frac{c\|\Delta w_k\|}{\|r_k\|}$. For the sequence $\{r_{ki}\}$ for $k \in [K]$, if the Lindeberg's condition holds, then $\frac{1}{K}\sum_{k=1}^{K} r_{ki} \to \mathcal{N}(0, \frac{1}{K^2}\sum_{k=1}^{K} \sigma_k^2)$. Thus, we must verify that for any $\epsilon > 0$,

$$
\lim_{K \to +\infty} \frac{1}{K} \sum_{k=1}^{K} \mathbb{E}[r_{ki}^2 \cdot \mathbf{1}\{|r_{ki}|^2 \geq \epsilon\sqrt{K}] = 0, \tag{5.3}
$$

where $\mathbf{1}$ is the indicator function. Note that r_{ki} can be represented by $\sigma_k \cdot x$, where x denotes a standard Gaussian random variable. Then we have

$$
\mathbb{E}\big[r_{ki}^2 \cdot \mathbf{1}\{|r_{ki}|^2 \geq \sum_{k=1}^{K} \epsilon\sqrt{K}\}|\big] \leq \sigma_k^2 \mathbb{E}\big[x^2 \cdot \mathbf{1}\{|x|^2 \geq \sum_{k=1}^{K} \epsilon\sqrt{K}\big]. \tag{5.4}
$$

And Eq. 5.4 goes to 0 when K is sufficiently large. □

C Proof of Theorem 2

This proof is deeply inspired by the proof developed in [4], and we roughly follow the same proof procedure.

Proof. The noise-perturbed global model parameter is updated as

$$\tilde{w}^{T+1} = \tilde{w}^T - \sum_{k=1}^{K} \frac{n_k}{n} \Delta \tilde{w}_k^T + R^T. \tag{5.5}$$

Assuming that w^* is the optimal parameter, we have the following.

$$\begin{aligned}
&\mathbb{E}\big[\ \|\tilde{w}^{T+1} - w^*\|^2\big] \\
=&\mathbb{E}\big[\|\tilde{w}^T - w^*\|^2\big] \underbrace{-2\mathbb{E}\big[\langle \sum_{k=1}^{K} \frac{n_k}{n} \Delta \tilde{w}_k^T, R^T \rangle\big]}_{B_1} + \underbrace{\mathbb{E}\big[\| \sum_{k=1}^{K} \frac{n_k}{n} \Delta \tilde{w}_k^T \|^2\big]}_{B_2} + \underbrace{\mathbb{E}\big[\|R^T\|^2\big]}_{B_3} \\
&+ \underbrace{2\mathbb{E}\big[\langle \tilde{w}^T - w^*, R^T \rangle\big]}_{B_4} \underbrace{-2\mathbb{E}\big[\langle \tilde{w}^T - w^*, \sum_{k=1}^{K} \frac{n_k}{n} \Delta \tilde{w}_k^T \rangle\big]}_{B_5}
\end{aligned} \tag{5.6}$$

Next, we bound the terms on the RHS of (5.6). By the Young's inequality, we have $B_1 \leq B_2 + B_3$. By the Cauchy-Schwarz inequality, we have

$$B_2 = \mathbb{E}\big[\big\| \sum_{k=1}^{K} \frac{n_k}{n} \Delta \tilde{w}_k^T \big\|^2\big] \leq \sum_{k=1}^{K} \frac{n_k}{n} \mathbb{E}\big[\big\| \Delta \tilde{w}_k^T \big\|^2\big] \tag{5.7}$$

$$= \eta^2 \sum_{k=1}^{K} \frac{n_k}{n} \mathbb{E}\big[\big\| \sum_{\tau=0}^{t-1} \nabla F_k(w_k^{T,\tau}, \xi_k^{T,\tau}) \big\|^2\big] \tag{5.8}$$

$$\leq \eta^2 t \sum_{\tau=0}^{t-1} \sum_{k=1}^{K} \frac{n_k}{n} \mathbb{E}\big[\big\| \nabla F_k(w_k^{T,\tau}, \xi_k^{T,\tau}) \big\|^2\big] \leq \eta^2 t^2 G^2, \tag{5.9}$$

$$B_3 = \mathbb{E}\big[\|R^T\|^2\big] \leq \frac{9m^2}{K^2} \sum_{k=1}^{K} (\sigma_k^T)^2, \tag{5.10}$$

where m is the dimension the model parameter and the inequality holds by Theorem 1 for a large enough K. Again, by the Cauchy-Schwarz inequality, we have

$$B_4 = 2\mathbb{E}\big[\langle \tilde{w}^T - w^*, R^T \rangle\big] \leq \mathbb{E}\big[\|\tilde{w}^T - w^*\|^2\big] + B_3. \tag{5.11}$$

$$
\begin{aligned}
B_5 =& 2\mathbb{E}\big[\langle w^* - \tilde{w}^T, \sum_{k=1}^{K} \frac{n_k}{n} \Delta \tilde{w}_k^T \rangle\big] \\
\leq& 2\eta \sum_{k=1}^{K} \sum_{\tau=0}^{t-1} \frac{n_k}{n} \mathbb{E}\big[\langle w^* - \tilde{w}^T, \nabla F_k(\tilde{w}_k^{T,\tau}, \xi_k^{T,\tau}) \rangle\big] \\
\leq& \underbrace{2\eta \sum_{k=1}^{K} \sum_{\tau=0}^{t-1} \frac{n_k}{n} \mathbb{E}\big[\langle \tilde{w}_k^{T,\tau} - \tilde{w}^T, \nabla F_k(\tilde{w}_k^{T,\tau}, \xi_k^{T,\tau}) \rangle\big]}_{C_1} + \underbrace{2\eta \sum_{k=1}^{K} \sum_{\tau=0}^{t-1} \frac{n_k}{n} \mathbb{E}\big[\langle w^* - \tilde{w}_k^{T,\tau}, \nabla F_k(\tilde{w}_k^{T,\tau}, \xi_k^{T,\tau}) \rangle\big]}_{C_2}
\end{aligned} \tag{5.12}
$$

$$
\begin{aligned}
C_1 =& \sum_{k=1}^{K} \sum_{\tau=0}^{t-1} \frac{n_k}{n} \mathbb{E}\big[\|\tilde{w}_k^{T,\tau} - \tilde{w}^T\|^2\big] + \eta^2 \sum_{k=1}^{K} \sum_{\tau=0}^{t-1} \frac{n_k}{n} \mathbb{E}\big[\|\nabla F_k(\tilde{w}_k^{T,\tau}, \xi_k^{T,\tau}\|^2\big] \\
\leq& \sum_{k=1}^{K} \sum_{\tau=0}^{t-1} \frac{n_k}{n} \mathbb{E}\left[\big\|\eta \sum_{i=0}^{\tau} \nabla F_k(\tilde{w}_k^{T,i}, \xi_k^{T,i})\big\|^2\right] + \eta^2 t G^2 \\
\leq& \frac{t(t+1)(2t+1)\eta^2 G^2}{6} + \eta^2 t G^2,
\end{aligned} \tag{5.13}
$$

$$
C_2 \overset{(e)}{\leq} 2\eta \sum_{k=1}^{K} \sum_{\tau=0}^{t-1} \frac{n_k}{n} \mathbb{E}\big[\langle w^* - \tilde{w}_k^{T,\tau}, \nabla F_k(\tilde{w}_k^{T,\tau}) \rangle\big] \tag{5.14}
$$

$$
\begin{aligned}
\overset{(f)}{\leq}& 2\eta \sum_{k=1}^{K} \sum_{\tau=0}^{t-1} \frac{n_k}{n} \mathbb{E}\big[F_k(w^*) - F_k(\tilde{w}_k^{T,\tau}) - \frac{\mu}{2}\|\tilde{w}_k^{T,\tau} - w^*\|^2\big] \\
\leq& 2\eta \sum_{k=1}^{K} \sum_{\tau=0}^{t-1} \frac{n_k}{n} \mathbb{E}\big[F_k(w^*) - F_k^* + F_k^* - F_k(\tilde{w}_k^{T,\tau}) - \frac{\mu}{2}\|\tilde{w}_k^{T,\tau} - w^*\|^2\big] \\
=& 2\eta t \Gamma + 2\eta \sum_{k=1}^{K} \sum_{\tau=0}^{t-1} \frac{n_k}{n} (F_k^* - F_k(\tilde{w}_k^{T,\tau}) - \mu\eta \sum_{k=1}^{K} \sum_{\tau=0}^{t-1} \frac{n_k}{n} \mathbb{E}\underbrace{\big[\|\tilde{w}_k^{T,\tau} - w^*\|^2\big]}_{C_3},
\end{aligned} \tag{5.15}
$$

where (e) and (f) are due to Assumptions 4 and 2, respectively.

$$
\begin{aligned}
C_3 &= \|\tilde{w}_k^{T,\tau} - \tilde{w}^T\|^2 + \|\tilde{w}^T - w^*\|^2 + 2\langle \tilde{w}_k^{T,\tau} - \tilde{w}^T, \tilde{w}^T - w^* \rangle \\
&\leq \|\tilde{w}_k^{T,\tau} - \tilde{w}^T\|^2 + \|\tilde{w}^T - w^*\|^2 - \frac{1}{\eta}\|\tilde{w}_k^{T,\tau} - \tilde{w}^T\|^2 - \eta\|\tilde{w}^T - w^*\|^2 \\
&= (1-\eta)\|\tilde{w}_k^T - w^*\|^2 - (\frac{1}{\eta} - 1)\|\tilde{w}_k^{T,\tau} - \tilde{w}^T\|^2, \qquad (5.16)
\end{aligned}
$$

Substituting C_3 into C_2, we have

$$
\begin{aligned}
C_2 =& 2\eta t\Gamma + 2\eta \sum_{k=1}^{K} \sum_{\tau=0}^{t-1} \frac{n_k}{n}(F_k^* - F_k(\tilde{w}_k^{T,\tau})) - \mu\eta t(1-\eta)\mathbb{E}\big[\|\tilde{w}^T - w^*\|^2\big] \\
&+ \mu(1-\eta)\eta^2 G^2 \frac{t(t+1)(2t+1)}{6}. \qquad (5.17)
\end{aligned}
$$

Substituting C_1 and C_2 into B_5, we have

$$
\begin{aligned}
B_5 \leq& -\mu\eta t(1-\eta)\mathbb{E}\big[\|\tilde{w}^T - w^*\|_2^2\big] + (1+\mu(1-\eta))\frac{t(t+1)(2t+1)\eta^2 G^2}{6} + \eta^2 t G^2 + 2\eta t\Gamma \\
&+ 2\eta \sum_{k=1}^{K} \sum_{\tau=0}^{t-1} \frac{n_k}{n}(F_k^* - F_k(\tilde{w}_k^{T,\tau})). \qquad (5.18)
\end{aligned}
$$

Substituting B_1-B_5 into Eq. 5.6, we have

$$
\begin{aligned}
\mathbb{E}\big[\ \|\tilde{w}^{T+1} - w^*\|^2\big] \overset{(g)}{\leq}& (2 - \mu\eta t(1-\eta))\mathbb{E}\big[\|\tilde{w}^T - w^*\|_2^2\big] + 2\eta t\Gamma + (1+2t)t\eta^2 G^2 \\
&+ (1+\mu(1-\eta))\frac{t(t+1)(2t+1)\eta^2 G^2}{6} + \frac{9m^2}{K^2}\sum_{k=1}^{K}(\sigma_k^T)^2, \qquad (5.19)
\end{aligned}
$$

where (g) follows from $F_k^* - F_k(\tilde{w}_k^{T,\tau}) \leq 0$. Rearranging Eq. 5.19 and summing from 0 to T, we have proved Theorem 2. □

D Proof of Theorem 3

Proof. We denote the global model parameter at aggregation T by $\tilde{w}^{T+1} = \tilde{w}^T - \Delta w^T + R^T$, where $\Delta w^T = \eta \sum_{k=1}^{K} \sum_{\tau=0}^{t-1} \frac{n_k}{n} \nabla F_k(\tilde{w}_k^{T,\tau}, \xi_k^{T,\tau})$. Due to the smoothness of Assumption 1 and

taking the expectation of $F_k(\tilde{w}^{T+1})$ over randomness at the T-th aggregation, we have

$$\begin{aligned}
\mathbb{E}[F(\tilde{w}^{T+1}] \leq & F(\tilde{w}^T) + \langle \nabla F(\tilde{w}^T), \mathbb{E}[R^T - \Delta w^T] \rangle + \frac{L}{2}\mathbb{E}[\|R^T - \Delta w^T)\|^2] && (5.20)\\
\leq & F(\tilde{w}^T) + \langle \nabla F(\tilde{w}^T), \mathbb{E}[R^T - \Delta w^T + \eta \nabla F(\tilde{w}^T) - \eta \nabla F(\tilde{w}^T)] \rangle \\
& + \frac{L}{2}\mathbb{E}[\|R^T - \Delta w^T\|^2] && (5.21)\\
\leq & F_k(\tilde{w}^T) + \underbrace{\langle \nabla F(\tilde{w}^T), \mathbb{E}[\eta \nabla F(\tilde{w}^T) - \Delta w^T] \rangle}_{A_1} + \underbrace{\frac{L}{2}\mathbb{E}[\|R^T - \Delta w^T\|^2}_{A_2} \\
& + \underbrace{\frac{1}{2}\mathbb{E}\|R^T\|^2}_{A_3} + \frac{1}{2}\|\nabla F(\tilde{w}^T)\|^2 - \eta \|\nabla F(\tilde{w}^T)\|^2. && (5.22)
\end{aligned}$$

$$\begin{aligned}
A_1 = & \langle \nabla F(\tilde{w}^T), \mathbb{E}\big[\eta \nabla F(\tilde{w}^T) - \Delta w^T\big] \rangle && (5.23)\\
= & \langle \sqrt{\eta t} \nabla F(\tilde{w}^T), \frac{\sqrt{\eta}}{\sqrt{t}} \mathbb{E}[\sum_{k=1}^{K} \sum_{\tau=0}^{t-1} \frac{n_k}{n} (\nabla F_k(\tilde{w}^T) - \nabla F_k(\tilde{w}_k^{T,\tau}, \xi_k^{T,\tau}))] \rangle && (5.24)\\
\overset{(b)}{\leq} & \frac{\eta t}{2} \|\nabla F(\tilde{w}^T)\|^2 + \frac{\eta}{2t} \mathbb{E}\big[\| \sum_{k=1}^{K} \sum_{\tau=0}^{t-1} \frac{n_k}{n} (\nabla F_k(\tilde{w}^T) - \nabla F_k(\tilde{w}_k^{T,\tau}, \xi_k^{T,\tau}))\|\big]^2 && (5.25)\\
\overset{(c)}{\leq} & \frac{\eta t}{2} \|\nabla F(\tilde{w}^T)\|^2 + \frac{\eta L^2}{2} \sum_{k=1}^{K} \sum_{\tau=1}^{t-1} \frac{n_k}{n} \mathbb{E}\big[\|\tilde{w}_k^{T,\tau} - \tilde{w}^T\|^2\big] \\
\leq & \frac{\eta t}{2} \|\nabla F(\tilde{w}^T)\|^2 + \frac{\eta^3 L^2}{2} \sum_{k=1}^{K} \sum_{\tau=0}^{t-1} \frac{n_k}{n} \mathbb{E}\| \sum_{i=0}^{\tau} \nabla F_k(\tilde{w}_k^{T,i}, \xi_k^{T,i})\|^2 && (5.26)\\
\overset{(d)}{\leq} & \frac{\eta t}{2} \|\nabla F(\tilde{w}^T)\|^2 + \frac{\eta^3 L^2 t(t+1)(2t+1)}{12} G^2, && (5.27)
\end{aligned}$$

where (b) follows from the Young inequality, and (c) is due to Assumptions 1 and $\mathbb{E}\| \sum_{i=1}^{n} x_i\|^2 \leq n \sum_{i=1}^{n} \mathbb{E}\|x_i\|^2$, and (d) is due to Assumption 3.

Based on the relationship of the noise and the gradient and following the Efron-Stein inequality, we have

$$A_2 = \frac{L}{2}\mathbb{E}[\|R^T - \Delta w^T\|^2 \leq \frac{m^2 L}{2K^2} \sum_{k=1}^{K} (\sigma_k^T)^2, \tag{5.28}$$

where m is the dimension of r_k.

$$A_3 = \frac{1}{2}\mathbb{E}\big[\|R^T\|^2\big] \leq \frac{1}{2}c^2\eta^2\mathbb{E}\big[\|\sum_{k=1}^{K}\sum_{\tau=0}^{t-1}\frac{n_k}{n}\nabla F_k(\tilde{w}_k^{T,\tau},\xi_k^{T,\tau})\|\big] \tag{5.29}$$

$$\leq \frac{1}{2}c^2\eta^2\sum_{k=1}^{K}\frac{n_k}{n}\mathbb{E}\big[\|\sum_{\tau=0}^{t-1}F_k(\tilde{w}_k^{T,\tau},\xi_k^{T,\tau})\|\big] \tag{5.30}$$

$$\leq \frac{1}{2}c^2\eta^2 t\sum_{k=1}^{K}\frac{n_k}{n}\sum_{\tau=0}^{t-1}\mathbb{E}\big[\|F_k(\tilde{w}_k^{T,\tau},\xi_k^{T,\tau})\|\big] \tag{5.31}$$

$$\leq \frac{1}{2}c^2\eta^2t^2G^2 \tag{5.32}$$

Substituting A_1, A_2 and A_3 into Eq. 5.27, we have

$$\begin{aligned}\mathbb{E}[F(\tilde{w}^{T+1}] \leq & F(\tilde{w}^T) + (\frac{1+\eta t-2\eta}{2})\|\nabla F(\tilde{w}^T)\|^2 + \frac{\eta^3L^2t(t+1)(2t+1)}{12}G^2 \\ & + \frac{m^2L}{2K^2}\sum_{k=1}^{K}(\sigma_k^T)^2 + \frac{1}{2}c^2\eta^2t^2G^2.\end{aligned} \tag{5.33}$$

Rearranging Eq. (5.33) and summing from $0-T$, we have

$$\begin{aligned}\sum_{T=1}^{T_{max}}\frac{1+\eta t-2\eta}{2}\|\nabla F(\tilde{w}^T)\|^2 \leq & F(w^0) - F(\tilde{w}^T) + \frac{\eta^3L^2t(t+1)(2t+1)}{12}TG^2 \\ & + \frac{m^2LT}{2K^2}\sum_{k=1}^{K}(\sigma_k^T)^2 + \frac{1}{2}Tc^2\eta^2t^2G^2,\end{aligned} \tag{5.34}$$

And we get

$$\begin{aligned}\min_{T\in[T_{max}]}\mathbb{E}\|\nabla F(\tilde{w}^T)\|^2 \leq & \frac{2(F(w^0)-F(\tilde{w}^*))}{(1+\eta t-2\eta)T} + \frac{\eta^3L^2t(t+1)(2t+1)G^2}{6(1+\eta t-2\eta)} \\ & + \frac{m^2L\sum_{k=1}^{K}(\sigma_k^T)^2}{K^2(1+\eta t-2\eta)} + \frac{c^2\eta^2t^2G^2}{1+\eta t-2\eta}.\end{aligned} \tag{5.35}$$

□