

Security-Enhanced and Privacy-Preserving Federated Learning

by

Tianyue Chu

A dissertation submitted in partial fulfilment of the
requirements for the degree of Doctor of Philosophy in

Telematic Engineering

Universidad Carlos III de Madrid

Advisor(s):

Nikolaos Laoutaris

May 2025

This work has been supported by IMDEA Networks Institute.



This thesis is distributed under license “Creative Commons **Attribution - Non Commercial - Non Derivatives**”.



*To my family, for their unconditional love, support, and belief in me through every
step of this journey.*

*To my supervisors and collaborators, whose guidance, patience, and shared passion for
discovery pushed me to grow as a researcher and a person.*

*To everyone who inspired me, challenged me, and walked with me on this path—thank
you for making this journey possible and unforgettable.*

ACKNOWLEDGEMENTS

Pursuing a PhD in Computer Science has been a transformative journey filled with hard work, setbacks, and moments of triumph. This journey has tested my resilience, expanded my knowledge, and deepened my passion for scientific discovery. The challenges encountered along the way have strengthened my determination, while the successes, both big and small, have fueled my motivation to push the boundaries of knowledge.

First and foremost, I extend my deepest gratitude to my supervisor, Prof. Nikolaos Laoutaris, for granting me the opportunity to pursue my PhD at IMDEA Networks. His unwavering support, profound expertise, and patient mentorship have been the guiding light throughout this journey. His inspiring ideas and thoughtful feedback have not only shaped my research but also helped me grow as a scholar and an individual. I will forever cherish the trust he placed in me and the invaluable lessons he imparted. I am also sincerely grateful to Prof. Athina Markopoulou for hosting me at the University of California, Irvine, and offering me a unique perspective on conducting research in a different global setting. Her generous mentorship, insightful advice, and genuine kindness made my time there truly transformative. The lessons I learned during that internship broadened my research horizon and left an indelible mark on my academic journey.

I am deeply thankful to my co-authors for their collaborative spirit, insightful discussions, and valuable contributions to our joint papers. I also owe heartfelt thanks to my teammates in the Data Transparency Group and the entire IMDEA Networks family, whose friendship and support turned challenges into opportunities and made these past years both rewarding and unforgettable.

To my beloved family and lifelong friends—your unconditional love, unwavering belief in me, and constant encouragement have been my anchor through every twist and turn. You provided the emotional strength and motivation that sustained me through the most demanding moments. I am forever indebted to you for being my greatest source of inspiration.

To all of you, thank you from the bottom of my heart.

The works presented in this thesis were partially funded within the framework of the following projects and grants:

- MLEDGE: Cloud and Edge Machine Learning project (REGAGE22e00052829516), funded by the Ministry of Economic Affairs and Digital Transformation and the European Union-NextGenerationEU/PRTR
- DataBri-X: Data Process & Technological Bricks for expanding digital value creation in European Data Spaces, funded by the European Union's Horizon Europe research and innovation programme under grant agreement No. 101070069.

PUBLISHED AND SUBMITTED CONTENT

During the course of this Ph.D., the author submitted, coauthored, and published a number of papers that are included in this thesis. The following list summarises those publications, details their inclusion in this thesis, and outlines the author's contributions to each:

- **Chu, T.**, Garcia-Recuero, A., Iordanou, C., Smaragdakis, G. and Laoutaris, N., 2023. Securing Federated Sensitive Topic Classification against Poisoning Attacks. In the 30th Annual Network and Distributed System Security Symposium, NDSS 2023. San Diego, California, USA. URL: <https://dx.doi.org/10.14722/ndss.2023.23112> [1].
 - This work is fully included and reported in Chapter 2.
 - The author fully participated in the writing of the paper, with her role primarily focused on designing the algorithm, establishing theoretical guarantees, implementing various state-of-the-art comparison algorithms, and analyzing and interpreting the results.
 - The material from this source included in this thesis is not singled out with typographic means and references.
- **Chu, T.** Laoutaris, N., 2024. FedQV: Leveraging Quadratic Voting in Federated Learning. ACM SIGMETRICS/IFIP PERFORMANCE Joint the ACM on Measurement and Analysis of Computing Systems, 8(2), pp.1-36. URL: <https://doi.org/10.1145/3656006> [2].
 - This work is fully included and reported in Chapter 3.
 - The author fully participated in the writing of the paper, with her role primarily focused on designing the methodology, conducting the theoretical analysis, implementing various attacks, and conducting extensive evaluation.
 - The material from this source included in this thesis is not singled out with typographic means and references.
- **Chu, T.**, Yang, M., Laoutaris, N. and Markopoulou, A., 2023. PriPrune: Quantifying and Preserving Privacy in Pruned Federated Learning. ACM Transactions on Modeling and Performance Evaluation of Computing Systems <https://doi.org/10.1145/3702241> [3].
 - This work is fully included and reported in Chapter 4.
 - The author fully participated in the writing of the paper, with her role primarily focused on designing the methodology, conducting the theoretical analysis, implementing various attacks, and conducting extensive evaluation.

- The material from this source included in this thesis is not singled out with typographic means and references.
- **Chu, T., İşler, D., Laoutaris, N.** 2024. Strengthening Privacy in Robust Federated Learning through Secure Aggregation. In Workshop on Artificial Intelligence System with Confidential Computing (AISCC 2024), co-located with NDSS Symposium 2024. DOI: <https://www.ndss-symposium.org/wp-content/uploads/aiscc2024-12-paper.pdf> [4].
 - This work is partially included and reported in Chapter 3.
 - The author fully participated in the writing of the paper, with her role primarily focused on designing the methodology, conducting the theoretical analysis, implementing various attacks, and conducting extensive evaluation.
 - The material from this source included in this thesis is not singled out with typographic means and references.
- **Chu, T., Yang, M., Laoutaris, N. and Markopoulou, A.,** 2024, July. Information-Theoretical Bounds on Privacy Leakage in Pruned Federated Learning. In IEEE ISIT 2024 Workshop on Information-Theoretic Methods for Trustworthy Machine Learning. [5]
 - This work is partially included and reported in Chapter 4.
 - The author fully participated in the writing of the paper, with her role primarily focused on analysing privacy leakage from a theoretical perspective, implementing privacy attacks, and conducting evaluations to validate the theoretical conclusions.
 - The material from this source included in this thesis is not singled out with typographic means and references.

ABSTRACT

Federated Learning (FL) has emerged as a promising machine learning paradigm that enables collaborative model training across decentralized datasets while preserving data privacy. Unlike traditional machine learning, that necessitate centralized data collection, FL trains models locally on client devices and aggregates only model updates, such as parameters or gradients, on a central server. This decentralized approach mitigates privacy risks while reducing data transfer costs, rendering FL well-suited for applications in healthcare, finance, the Internet of Things (IoT), and edge computing.

Despite its potential, FL faces significant challenges in terms of security and privacy. Malicious clients can launch poisoning attacks by manipulating model updates or embedding backdoors. Furthermore, sensitive client data remains vulnerable to privacy attacks, such as gradient inversion attacks, where attackers reconstruct private data from shared gradients. These concerns underscore the need for robust defense mechanisms that ensure both security and privacy within FL systems.

This thesis addresses two critical challenges in FL: enhancing Byzantine robustness to thwart poisoning attacks and advancing privacy-preserving mechanisms to mitigate data leakage risks. To address the first challenge, we develop a robust FL aggregation scheme based on subjective logic and residual-based attack detection. This scheme retains historical data from previous rounds to build client reputations. Employing a combination of theoretical analysis, trace-driven simulation, and experimental validation with a prototype and real users, we show that our FL classifier can detect sensitive content with high accuracy, learn new labels rapidly, and remain robust against poisoning attacks from malicious users as well as imperfect input from non-malicious ones.

Additionally, we identify a major weakness in FL—its vulnerability to poisoning attacks—as a consequence of the *one person one vote* (henceforth *1p1v*) principle underlying most contemporary aggregation rules. To overcome this, we introduce FEDQV, a novel aggregation algorithm built upon the *quadratic voting* scheme, recently proposed as a better alternative to *1p1v*-based elections. Our theoretical analysis establishes that FEDQV is a truthful mechanism in which bidding according to one’s true valuation is a dominant strategy that achieves a convergence rate matching that of state-of-the-art methods. Furthermore, our empirical analysis using multiple real-world datasets validates the superior performance of FEDQV against poisoning attacks. It also shows that combining FEDQV with unequal voting “budgets” according to a reputation score increases its performance benefits even further. FEDQV can also be easily combined with Byzantine-robust privacy-preserving mechanisms to enhance its robustness against both poisoning and privacy attacks.

To address the second challenge, we first characterize the privacy offered by pruning. We establish information-theoretic upper bounds on the information leakage from

pruned FL and we experimentally validate them under state-of-the-art privacy attacks across different FL pruning schemes. Furthermore, we introduce *PriPrune*— a privacy-aware algorithm for pruning in FL. *PriPrune* uses defense pruning masks, which can be applied locally after *any* pruning algorithm, and adapts the defense pruning rate to jointly optimize privacy and accuracy. A key idea in the design of *PriPrune* is *pseudo-pruning*: it undergoes defense pruning within the local model and only sends the pruned model to the server; while the weights pruned out by defense mask are withheld locally for future local training rather than being removed. We show that *PriPrune* significantly improves the privacy-accuracy tradeoff compared to state-of-the-art pruned FL schemes.

Addressing the critical challenges of robust aggregation and privacy-aware model training is crucial for integrating FL into practical scenarios. Through rigorous theoretical analysis and comprehensive empirical evaluations, this thesis demonstrates that the proposed methods significantly enhance FL’s robustness and privacy preservation. By providing innovative tools and strategies to overcome these obstacles, this thesis establishes FL as a secure and privacy-preserving learning paradigm, paving the way for its widespread adoption across diverse domains and applications.

CONTENTS

1. INTRODUCTION.	1
1.1. Federated Learning.	2
1.2. Challenges in Federated Learning	6
1.2.1. Enhancing Byzantine Robustness and Security in FL.	6
1.2.2. Privacy leakage in FL	8
1.3. Problem Statement	8
1.4. Contributions of the Thesis	10
2. BYZANTINE-ROBUST AGGREGATION MECHANISM IN FEDERATED LEARNING.	13
2.1. Introduction.	13
2.2. Background.	15
2.2.1. A Centralised Offline Classifier for Sensitive Content	15
2.2.2. Challenges in Developing a Practical Classifier for Users	15
2.2.3. Related Work	16
2.3. A Robust FL Method for Classifying Sensitive Content on the Web	18
2.3.1. FL Framework for Classifying Sensitive Content	18
2.3.2. A Reputation score for Thwarting Poisoning Attacks.	19
2.3.3. Theoretical Guarantees	26
2.4. Performance Evaluation	33
2.4.1. Experimental Setup	33
2.4.2. Convergence and Accuracy	38
2.4.3. Resilience to Attacks.	39
2.4.4. Evaluation with an Increased Number of Participating Clients	42
2.4.5. Supplementary Dataset For Validation	43
2.4.6. Stability of Hyper-parameters.	44
2.4.7. Comparison Against the Residual-based Method	45
2.5. The <i>EITR</i> system.	46
2.5.1. System Architecture and Implementation	46

2.5.2. Real Users Experimental Setup	47
2.5.3. Validation with Real Users	48
2.6. Conclusions.	49
3. SECURE FEDERATED LEARNING AGGREGATION LEVERAGING QUADRATIC VOTING	50
3.1. Introduction.	50
3.2. Background.	53
3.2.1. Election Mechanisms in FL	53
3.2.2. FL Aggregation against Poisoning and Privacy Attacks	53
3.3. FEDQV: Quadratic Voting in FL.	54
3.3.1. Federated Learning Setting	54
3.3.2. Motivations	55
3.3.3. FEDQV Design	58
3.3.4. FEDQV with Adaptive Budgets	62
3.4. Theoretical Analysis	64
3.4.1. Convergence	64
3.4.2. Truthfulness	75
3.5. Experiments	77
3.5.1. Experimental Setting.	77
3.5.2. Threat Model	77
3.5.3. Evaluated Poisoning Attacks	78
3.5.4. Performance Metrics.	80
3.5.5. Convergence	80
3.5.6. Defence against Poisoning Attacks	81
3.5.7. Adaptive Budget	83
3.5.8. Integration with Byzantine-robust Aggregation	84
3.5.9. Comparison with Alternative Voting Strategies	86
3.5.10. Non-IID Degree.	87
3.5.11. Impact of Hyper-parameters	88
3.6. Defence against Privacy Attacks	89
3.6.1. Secure Aggregation for FL	89

3.6.2. Implementing SECAGG in FEDQV	90
3.6.3. Privacy Attacks against FEDQV with SECAGG	92
3.7. Conclusion	94
4. QUANTIFYING AND PRESERVING PRIVACY IN PRUNED FEDERATED LEARNING	95
4.1. Introduction.	95
4.2. Related Work	97
4.3. Problem Setup	99
4.4. Quantifying Privacy In FL Model Pruning.	101
4.4.1. Theoretical Quantification	102
4.4.2. Empirical Quantification.	107
4.5. Design of Defense Pruning	114
4.5.1. Insights from Privacy Quantification	114
4.5.2. The <i>PriPrune</i> Mechanism	118
4.6. Experimental Evaluation	120
4.6.1. Experimental Setup	120
4.6.2. Performance of <i>PriPrune</i>	121
4.6.3. Hyper-parameter Selection	125
4.7. Conclusion and Future Work	125
5. DISCUSSION	127
5.1. Enhancing Byzantine Robustness and Security in FL.	127
5.1.1. Reputation-based Robust Aggregation	128
5.1.2. A Superior Fundamental Aggregation	129
5.2. Privacy leakage in FL	130
5.3. Open-Source Code Contribution	132
5.4. Conclusion	132
BIBLIOGRAPHY.	134

LIST OF FIGURES

1.1	Overview of the Federated Learning training process. Step 1: The server broadcasts the global model to the selected clients. Step 2: Each selected client performs local training and sends its local model updates to the server. Step 3: The server aggregates the local model updates. Step 4: The server updates the global model. This process is repeated over multiple training rounds, with the goal of minimizing the model’s loss and improving its fit to the data.	5
2.1	Accuracy of FL classifiers and centralised classifiers in Health, Religion and all categories.	19
2.2	Overview of reputation-based aggregation algorithm.	20
2.3	The decay of reputation score in Client (X) with X model parameters when they (a) attack once at 3rd iteration and (b) attack continuously at and after the 3rd iteration.	24
2.4	The decay of reputation score in Client X with the same model parameters when they (a) attack once at X iteration and (b) attack continuously after starting to attack at X iteration.	25
2.5	The decay of reputation scores of malicious clients under multi-iteration poisoning attacks. (Left) Four malicious clients with varying model sizes (1k, 10k, 100k, and 1M parameters) perform attacks at iterations 10, 50, and 90. (Right) Four malicious clients, each with 1M parameters, perform two attacks: the first at iteration 10 and the second at iteration $10 + X$, where $X \in \{40, 50, 60, 70\}$	25
2.6	Training Loss (TL) and Average Accuracy (AA) for 100 epochs of Reputation-based, FedAvg, Residual-based, Median, Trimmed-mean, FoolsGold and FLTrust methods in SURL Dataset under no attack (left) scenario, under label flipping attack (middle) and backdoor attack (right) scenarios with 30% malicious clients.	38
2.7	Average accuracy (ACC) and attack success rate (ASR) for varying percentage of attackers from 10% to 50% under label flipping (left) and backdoor (right) attack for Reputation-based, FedAvg, Residual-based, Median, Trimmed-mean, FoolsGold and FLTrust methods in SURL Dataset.	39
2.8	ACC and ASR as we vary the percentage of flipped labels from 10% to 90% (left) for label flipping attack, and the number of the frequent words as the trigger pattern from 5 to 25 for backdoor attack (right).	41

2.9	Average accuracy (ACC) and attack success rate (ASR) for varying percentage of attackers from 10% to 50% under label flipping (left) and backdoor (right) attack with a looping attack in which an attacker attacks every 30 epochs.	41
2.10	Average accuracy (ACC) and attack success rate (ASR) for varying the number of clients from 10 to 200 under label flipping and backdoor attack with 30% malicious clients.	42
2.11	Average accuracy (ACC) with no attack (left) and 30% percentage of attackers under label flipping (middle) and backdoor (right) attack for Reputation, FedAvg, Residual-based, Median, Trimmed-mean, Foolsgold, FLTrust in SURL dataset with 100 clients.	43
2.12	Attack success rate (ASR) for varying percentage of attackers from 10% to 50% under label flipping (left) and backdoor (right) attack for Reputation, FedAvg, Residual-based, Median, Trimmed-mean, Foolsgold, FLTrust in SURL dataset with 100 clients.	43
2.13	Average accuracy (ACC) and attack success rate (ASR) for varying percentage of attackers from 10% to 50% under label flipping (left) and backdoor (right) attack for Reputation, FedAvg, Residual-based, Median, Trimmed-Median, FoolsGold and FLTruts in CIFAR-10 dataset.	44
2.14	Average accuracy and attack success rate as we vary rewarding weight κ and prior probability a	44
2.15	The aggregation weights of clients from our reputation-based (left) and residual-based method (right) for 10 communication rounds under label flipping attack with 80% attackers.	45
2.16	<i>EITR</i> extension in action. The letter “H” inside the red frame at the bottom right of the extension’s icon indicates that a health-related page has been detected.	46
2.17	Results of real-user experiment for COVID-19 related URLs with 50 users over 100 iterations. (left): the reputation score of the real users at the end of experiment, (middle): the accuracy of the centralised classifier and the global model of our reputation-based FL approach for COVID-19 URLs, and (right): the ROC curve of real-user experiment with 0.79 area under the ROC curve (AUC).	47
3.1	Visualization of Aggregation Weights and Robustness Performance of FEDQV	57
3.2	Overview of FEDQV algorithm.	59

3.3	Convergence comparison of FEDAVG and FEDQV in Training Loss (TL) and Accuracy (ACC) over 100 epochs across four benchmark datasets, both under no attack and under attack scenarios.	81
3.4	Performance comparison of FEDQV and FEDAVG in terms of Accuracy (ACC) and Attack Success Rate (ASR) over 100 epochs across four benchmark datasets under two targeted attacks, with varying percentages of attackers m from 10% to 50%.	83
3.5	Performance comparison of FEDAVG, FEDQV, and FEDQV+REP (FEDQV with adaptive budgets based on reputation model). Performance comparison of Krum alone and Multi-Krum integrated with FEDQV.	84
3.6	Impact of varying hyper-parameters budget B and similarity threshold θ of FEDQV on ACC and ASR under a backdoor attack scenario with 30% malicious parties, evaluated on the MNIST dataset.	88
3.7	Overview of implementation of the SECAGG protocol in FEDQV.	91
3.8	Visual Comparison of Images Reconstructed by Privacy Attacks (DLG and GI) on FEDQV with and without SECAGG. The first row displays the original private training images from the targeted client. The second row illustrates reconstructed images resulting from the privacy attacks (DLG and GI) on FEDQV without SECAGG. In the third row, reconstructed images from the privacy attacks (DLG and GI) on FEDQV with SECAGG are presented. The use of SECAGG is observed to contribute significantly to mitigating information leakage from the images.	93
4.1	FL with model pruning, under a privacy attack by an honest-but-curious server. Step 1: the server broadcasts the global model with weight w^t to users. Step 2: After local training, user i employs base pruning with pruning mask $\bar{m}_i^t$, resulting in a pruned model with weights $\bar{w}_i^t$. The user then applies a defense pruning technique, yielding a pruned model with weights $\hat{w}_i^t$. Step 3, the server aggregates the model updates and launches a gradient inversion attack upon receiving $\hat{w}_i^t$. Step 4: the server applies global pruning with pruning mask $\bar{m}^t$, and updates the global model accordingly. Notice <i>Base</i> pruning can be <i>any</i> FL pruning scheme. <i>Defense</i> pruning is introduced, in this paper, specifically to protect against privacy attacks.	97
4.2	Impact of varying pruning rate on privacy leakage using key metrics: Normalized Mutual Information (NMI) displayed on (a) and Peak Signal-to-Noise Ratio (PSNR) shown on (b) under a Sparse Gradient Inversion attack (SGI) in FEMNIST for five pruning methods under varying the base pruning rate.	108

4.3	Impact of varying pruning rates on privacy leakage for PruneFL in FEMNIST under Gradient Inversion (left) and DLG (right) attacks.	109
4.4	Impact of varying pruning rate on image reconstruction in FEMNIST using different base pruning methods under SGI attack.	111
4.5	Comparison between the SGI (Sparse Gradient Inversion) attack and the GI (Gradient Inversion) attack on the FEMNIST dataset for five pruning methods under varying the base pruning rate. In addition to the NMI metric, we show the corresponding recovered images, which show the degradation with an increasing pruning rate.	111
4.6	Impact of varying base pruning rate $\bar{p}$ and batch size B on privacy leakage. We consider the Normalized Mutual Information (NMI) as a privacy metric. We launch a Sparse Gradient Inversion (SGI) attack against 5 base pruning methods (Random, SNIP, SynFlow, FedDST, and PruneFL), over two datasets (FEMNIST in the First Row and CIFAR10 in the (Second Row)).	112
4.7	Impact of varying model size d and training rounds T on privacy leakage using Normalized Mutual Information (NMI) under a Sparse Gradient Inversion (SGI) attack for six pruning methods: Random, SNIP, SynFlow, GraSP, FedDST, and PruneFL in FEMNIST dataset (First Row) and CIFAR10 dataset (Second Row).	113
4.8	Tradeoff of privacy vs. accuracy with three defense strategies: <i>Largest</i> (red), <i>Random</i> (green), and <i>Mix</i> (yellow), and their enhanced versions incorporating <i>Pseudo-Pruning</i> (<i>Pseudo</i>). <i>Pseudo</i> could improve the overall tradeoff by elevating model accuracy and <i>Largest</i> offers the highest level of privacy protection among all defense methods.	115
4.9	Comparative performance of four defense rate combinations in the MIX strategy over 20,000 FL rounds, evaluated in terms of accuracy, NMI, and attack loss.	116
4.10	Comparison of Privacy-Accuracy Trade-Off with Varying Defense Rate (0% to 50%) in Largest Method based on PruneFL using NMI and Attack Loss Privacy Metrics	117
4.11	Illustration of local pruning at user i with <i>Pseudo-Pruning</i> defense: First user applies base pruning with base mask $\bar{\mathbf{m}}_i$ to derive a pruned model $\bar{\mathbf{w}}_i^t$. Then, defense pruning is conducted with mask $\hat{\mathbf{m}}_i$. Defense pruning is implemented as <i>Pseudo-Pruning</i> , where the remaining weights $\hat{\mathbf{w}}_i^t$ ($= \bar{\mathbf{w}}_i^t \odot \hat{\mathbf{m}}_i^t$) are sent to the server; while the pruned weights $\tilde{\mathbf{w}}_i^t$ ($= \bar{\mathbf{w}}_i^t \odot \tilde{\mathbf{m}}_i^t$) are locally saved for future local training. Notably, these weights pruned by defense are not actually pruned, hence the term "Pseudo-Pruning." . .	118

4.12	Performance evaluation of <i>PriPrune</i> with adaptive defense rate and <i>Largest (Pseudo)</i> with 6 fixed defense rates, as defense strategies, integrated with PruneFL (as the base pruning method), conducted over 20,000 training rounds across the FEMNIST dataset. (a) Comparison of ACC over training rounds under these different defense strategies. (b) Comparison of Privacy (NMI) over training rounds under these different defense strategies. (c) The Privacy-Utility tradeoff, measured in terms of NMI and ACC, respectively.(d) Change in the adaptive defense rate of <i>PriPrune</i> over training rounds.	121
4.13	Comparison of the communication costs of PruneFL with and without <i>PriPrune</i> during the training process with (a) representing FL training using the Conv-2 model on the FEMNIST dataset and (b) the VGG-11 model on the CIFAR-10 dataset.	123
4.14	Comparison of the training speed of PruneFL with and without <i>PriPrune</i> during the training process with (a) representing FL training on the FEMNIST dataset and (b) on the CIFAR-10 dataset.	123
4.15	Comparison of tradeoffs between privacy-vs-accuracy among four base pruning mechanisms, both with and without the integration of <i>PriPrune</i> , across the FEMNIST and CIFAR-10 datasets.	124

LIST OF TABLES

2.1	Notations	18
2.2	Comparison of FL aggregation methods in terms of robustness to attacks, efficiency, and scalability	35
2.3	Default Experimental Settings	38
3.1	Comparison of FEDQV and FEDAVG on four benchmark datasets under 10 attack scenarios with 30% malicious parties. The best results are highlighted in bold.	82
3.2	Comparison of Multi-Krum, Trimmed-Mean, Reputation, and their integration with FEDQV under three State-of-the-Art attacks on FEMNIST dataset. The best results are in bold. The results of targeted attacks are in the form of "ACC / ASR".	84
3.3	Comparison of Multi-Krum and Multi-Krum + FEDQV under two targeted attacks with 30% malicious parties in MNIST and Fashion-MNIST dataset. The best results are in bold.	85
3.4	Comparison of Trimmed-Mean and Trimmed-Mean Integrated with FedQV Methods under Targeted Attacks (Backdoor and QV-Adaptive) Across Varying Percentages of Malicious Parties.	86
3.5	Comparison of FEDPV, FEDCV and FEDQV on two benchmark datasets under 5 attack scenarios with 20% malicious parties. The best results are highlighted in bold.	86
3.6	Comparison of Accuracy (ACC) and Attack Success Rate (ASR) for FedQV and FedAvg under Backdoor Attack over 100 epochs with varying Non-IID Degrees on Fashion-MNIST Dataset.	87
4.1	Comparison for Evaluated Methods	110
4.2	Comparison of privacy levels (NMI) between Base and Base + <i>PriPrune</i> for FL pruning methods. A lower NMI indicates a reduced risk of privacy leakage.	124
4.3	Comparison of privacy levels (NMI) during hyper-parameter selection: with regards to our hyperparameters in loss functions, we search λ_{acc} in the list of [1, 5, 10], λ_{pri} in the list of [1, 10, 15] and λ_{sha} in the list of [2e-06, 2e-05, 2e-04].	125

4.4	Parameter settings for λ_{pri} , λ_{sha} , and λ_{acc} in FEMNIST and CIFAR-10 datasets.	125
-----	--	-----

LIST OF ACRONYMS

Acronym	Meaning
ML	Machine Learning
FL	Federated Learning
DNNs	Deep Neural Networks
FTL	Federated Transfer Learning
GDPR	General Data Protection Regulation
CCPA	California Consumer Privacy Act
IoT	Internet of Things
AR	Augmented Reality
TF-IDF	Term Frequency-Inverse Document Frequency
IID	Independent and Identical Distribution
SGD	Stochastic Gradient Descent
IRLS	Iteratively Reweighted Least Squares
SMPC	Secure Multi-Party Computation
ASR	Attack Success Rate
EITR	Elephant In the Room
1p1v	one person one vote
QV	Quadratic Voting
PGD	Projected Gradient Descent
DP	Differential Privacy
SA	Secure Aggregation
TEE	Trusted-execution Environment
DLG	Deep Leakage from Gradients
iDLG	Improved Deep Leakage from Gradients
GI	Gradient Inversion
ROG	Reconstruction From Obfuscated Gradient
SGI	Sparse Gradient Inversion
zCDP	Zero-concentrated Differential Privacy
MI	Mutual Information
NMI	Normalized Mutual Information
PSNR	Peak Signal-to-Noise Ratio

1. INTRODUCTION

Machine learning (ML) [6]–[9] is a field of artificial intelligence that enables computer systems to learn patterns and make decisions from data without being explicitly programmed. By applying algorithms to data, ML models can recognize patterns, make predictions, and adapt to new information. Traditional ML relies on a centralized approach where data from various sources is collected into a central repository, such as a cloud environment or dedicated data centre. Once gathered, algorithms train models on this aggregated dataset, producing models that can generalize and make predictions on unseen data. This approach has driven advances in numerous applications, including computer vision [10]–[13], natural language processing [14]–[16], and recommendation systems [17]–[19].

However, the centralized nature of ML presents significant challenges related to data privacy, security, and scalability, particularly as the volume of distributed data generated by devices, sensors, and clients continues to grow. In many real-world applications [20]–[23], data required for training ML models is inherently distributed across organizations, devices, or clients. This distribution introduces considerable obstacles to centralization due to privacy risks, high data transfer costs, and strict compliance requirements imposed by data protection regulations such as the General Data Protection Regulation (GDPR) [24] and the California Consumer Privacy Act (CCPA) [25]. These constraints render centralized ML impractical in privacy-sensitive environments, underscoring the need for decentralized learning approaches.

Federated Learning (FL) [26], [27] offers a natural solution to the above-mentioned limitations. It allows multiple parties to collaboratively train a global model without sharing their private data. First proposed by Google in 2016 [26], FL was introduced as a method to harness the massive amounts of data generated by edge devices while preserving user privacy and reducing communication costs. It has proved its value in a slew of real-world applications, ranging from mobile computing [21], [28], [29] to health and medical applications [20], [30], [31]

Despite its advantages, FL also faces significant security and privacy challenges. Malicious clients can disrupt the training process by launching poisoning attacks, either by degrading the accuracy of the global model [32], [33] or injecting backdoors into it [34], [35]. These attacks compromise the integrity of the model and reduce its reliability. Furthermore, FL remains susceptible to privacy attacks [36], [37], as sensitive client data can still be inferred from local model updates shared during the FL training process.

Although many works have been dedicated to addressing these issues, new challenges continue to emerge, necessitating the development of innovative methods and tools. Among these challenges, this thesis focuses on two critical aspects of FL: improving Byzantine robustness, which ensures reliable model training even in the presence

of malicious clients, and enhancing privacy-preserving capabilities, which mitigates the risks of sensitive data leakage during the training process. These topics are central for several reasons:

First, state-of-the-art Byzantine-robust FL techniques designed to defend against poisoning attacks often exhibit slow convergence rates. This makes them impractical for real-world applications, where rapid adaptation to new information is essential. Furthermore, these methods typically fail to retain information from previous aggregation rounds, allowing adversaries to mount successful attacks by distributing them over time [38]. Second, the most commonly employed aggregation rule in FL remains vulnerable to poisoning attacks [39]. This vulnerability arises from the principle of one person one vote (1p1v), in which all local data points are treated equally during aggregation. This can lead to unfair advantages for malicious clients, who may possess, or falsely claim to possess, large datasets for training. Third, the expected level of privacy protection offered by the privacy-enhancing technique in FL has not been comprehensively characterized. This lack of understanding creates a need for further research to optimize these techniques for privacy while preserving utility. In conclusion, addressing these three issues is imperative before FL can be deployed in real-world scenarios.

This chapter introduces the concept of FL, which forms the foundation of this thesis. Section 1.1 provides an overview of FL. Section 1.2 outlines the key challenges in the current state-of-the-art FL that are addressed in this thesis. The problem statement and research questions are presented in Section 1.3, followed by a summary of the contributions and the structure of the thesis in Section 1.4.

1.1. Federated Learning

Federated Learning (FL) is a specialized machine learning paradigm designed to train machine learning models across numerous local datasets held by individual clients. A key distinguishing feature of FL is its ability to train a global model without sharing the clients' private datasets. The learning process is iterative: the global model is first broadcast to participating clients, who train it locally on their datasets. Instead of transferring raw data, clients send model updates—such as parameters or gradients—back to a central server. The server aggregates these updates to refine the global model, which is then redistributed to all clients for the next training round.

In FL, clients possess local datasets generated from diverse sources, reflecting varying user behaviours, environments, and preferences. As a result, the number and types of labels vary across clients, leading to significant distributional imbalances. For example, some clients might have datasets dominated by specific classes, while others might have a more even label distribution. This inconsistency poses challenges in ensuring fair model updates and achieving convergence across all clients. This heterogeneity mirrors real-world scenarios where datasets can differ substantially in class representation, quantity,

and quality. Unlike centralized data collection environments, where data can be aggregated and curated to ensure balance, FL operates in decentralized settings, making such balancing nearly impossible. Consequently, FL must address issues such as statistical heterogeneity, model divergence, and fairness in aggregation, all of which complicate the learning process.

Moreover, FL clients are often less reliable due to their susceptibility to failures and dropout events. This fragility stems from several practical constraints, including unstable communication infrastructure, limited computational power, and energy constraints. Many FL clients rely on wireless communication methods such as Wi-Fi, mobile networks, or Bluetooth, which are prone to disruptions, signal loss, and latency. Additionally, clients are frequently battery-powered devices like smartphones, smartwatches, and IoT devices, making energy efficiency critical. Devices may go offline unexpectedly due to power depletion, network outages, or system updates, interrupting the learning process and further complicating client-server coordination.

Definition of FL Consider an FL system involving N clients and a central server. During training round t , a subset of clients $\mathcal{S}'$ is selected to participate in the training task. Each client i has the local dataset $\mathcal{D}_i$ with $|\mathcal{D}_i|$ samples, drawn from non-independent and non-identically (Non-IID) distribution $\mathcal{X}_i(\mu_i, \sigma_i^2)$ and the whole dataset $\mathcal{D} = \bigcup_{i=1}^N \mathcal{D}_i$ with the mean μ_i and standard deviation σ_i . The goal of using FL is to learn a global model at the server by aggregating the local model updates collected from participating clients. Given the loss function $\ell(\mathbf{w}; \mathcal{D})$, where $\mathbf{w} \in \mathcal{W}$, the objective function of FL can be described as

$$\mathcal{L}(\mathbf{w}) = \mathbb{E}_{\mathcal{D} \sim \mathcal{X}} [\ell(\mathbf{w}; \mathcal{D})]$$

Therefore, the task becomes:

$$\mathbf{w}^* = \arg \min_{\mathbf{w} \in \mathbb{R}^d} \mathcal{L}(\mathbf{w})$$

To find the optimal $\mathbf{w}^*$, Stochastic Gradient Descent (SGD) is employed to optimise the objective function. Let T be the total number of local SGD updates performed by a single party during the entire training process, E be the local iterations between two communication rounds, and thus $\frac{T}{E}$ is the number of communication rounds.

Types of FL FL can be categorized into three main types based on the data distribution across clients:

- **Horizontal FL:** In this type, all clients share the same feature space but have different samples. For example, for a hospital network, each hospital may have patient records with identical features such as age, gender, and medical history. However, the actual patients represented in each hospital's dataset are different, resulting in distinct samples across clients.

- **Vertical FL:** In vertical FL, clients have different features but share the same samples. An example would be a collaboration between a bank and an e-commerce platform. The bank has customer data such as credit scores, income levels, and loan history, while the e-commerce platform has purchase history, browsing behavior, and product preferences. Both organizations share the same customers (samples) but collect different types of data (features) about them.
- **Federated Transfer Learning (FTL):** In FTL, clients have both different features and different samples. An example would be a collaboration between a university and an online learning platform. The university collects data on students' academic records, course enrolments, and grades, while the online learning platform tracks students' course progress, quiz scores, and engagement metrics. The students in both datasets may not overlap, and the features collected by each institution are different. This type of FL is often applied when there is little overlap between the datasets but knowledge from one domain can be transferred to another.

In this thesis, we focus exclusively on horizontal FL as it embodies the original intent behind the design of FL and remains the most widely adopted scenario within the field, particularly in settings involving numerous users and resource-constrained devices.

Training of FL The horizontal FL model training process involves several rounds of communication between the clients and the server, including broadcasting, local training, and aggregation. The global model is first initialised at a random state by the server and, then, in round t the following steps are taken:

Broadcast: The server first randomly selects a subset of clients $\mathcal{S}^t$ ($|\mathcal{S}^t| = \mathcal{C} \geq 1$) and then distributes the latest global model as the global proposal to all chosen clients.

Local Training: The selected client i performs local computation based on the global proposal and its local dataset $\mathcal{D}_i$, and sends the update $\mathbf{w}_i^t$ back to the server:

$$\mathbf{w}_i^t \leftarrow \mathbf{w}_i^{t-1} - r_{t-1} \frac{\partial \ell_i(\mathbf{w}_i^{t-1}; \mathcal{D}_i)}{\partial \mathbf{w}}$$

where r_{t-1} is the learning rate.

Aggregation: The server receives the model updates from all participating clients and aggregates them to update the global model.

For the aggregation rule, FEDAVG [26] uses the fraction of the local training sample size of each client over the total training samples as the weight of a client:

$$\mathbf{w}^{t+1} = \frac{1}{\sum_{i \in \mathcal{S}^t} |\mathcal{D}_i|} \sum_{i \in \mathcal{S}^t} |\mathcal{D}_i| \cdot \mathbf{w}_i^t$$

The overview of the training process is summarized in Figure 1.1.

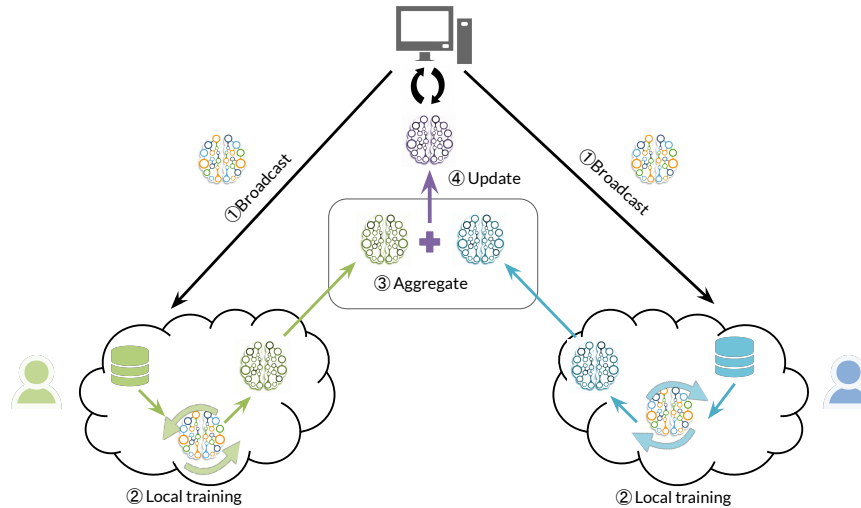


Figure 1.1: Overview of the Federated Learning training process. Step 1: The server broadcasts the global model to the selected clients. Step 2: Each selected client performs local training and sends its local model updates to the server. Step 3: The server aggregates the local model updates. Step 4: The server updates the global model. This process is repeated over multiple training rounds, with the goal of minimizing the model’s loss and improving its fit to the data.

This process repeats for multiple rounds until the global model converges. Throughout this process, no raw data is ever shared between clients or with the server, which ensures that sensitive information remains private.

Advantages of FL The key benefits of FL include:

- **Privacy Preservation:** FL enhances privacy by keeping data localized on clients’ devices, minimizing the need for centralized data collection. Since only model updates or gradients are transmitted to the central server, sensitive personal data remains on the devices. This approach significantly reduces privacy risks compared to traditional machine learning methods that require aggregating raw data into a central repository. Advanced techniques like differential privacy and secure aggregation can further strengthen privacy guarantees in FL systems.
- **Data Accessibility:** In many scenarios, data is siloed across different organizations or jurisdictions (e.g., across hospitals or financial institutions), making it difficult or impossible to centralize. FL allows for collaborative model training without the need to move the data. This accessibility allows training models on richer datasets, which improves generalization and robustness while ensuring compliance with data protection laws such as GDPR.

Applications of FL The versatility of FL has led to its adoption in a wide range of industries:

- Healthcare: FL has been used to develop predictive models for disease diagnosis [20], [40], patient similarity [41], and treatment recommendations [42], all while ensuring patient data remains on-premise in hospitals. For instance, FL has been applied to predict the future oxygen requirements of symptomatic patients with COVID-19 by pooling input data of vital signs, laboratory data and chest X-rays from 20 institutes across the globe without sharing sensitive medical records [43].
- Finance: FL has been used to improve fraud detection [22], [44], credit scoring [45], and personalized financial recommendations [46] in the financial sector. By leveraging data from multiple financial institutions without exposing customer information, FL can help build more robust fraud detection models that are harder to evade.
- Internet of Things (IoT): With the proliferation of IoT devices such as smart home systems [47], autonomous vehicles [23], and wearable technology [48], FL is emerging as a powerful tool for enabling these devices to collaboratively learn and improve their models. For example, FL is being leveraged to create a serverless, real-time seizure detection framework that can be deployed on resource-constrained wearable IoT devices [49].
- Mobile and Edge Computing: FL has become a standard method for training models in mobile applications, particularly for mobile keyboard predictions [21], [50]. Mobile FL allows applications to deliver personalized experiences to users without compromising their privacy.

1.2. Challenges in Federated Learning

While FL offers significant advantages in terms of privacy preservation and data decentralization, it also presents a set of challenges that need to be addressed to advance the current state-of-the-art FL. These challenges arise from the decentralized nature of FL [34], [51], the potential for malicious behaviour from some participants or the server. In the following, we explore the intricate challenges related to enhancing Byzantine robustness, quantifying the privacy implications, and developing corresponding defense mechanisms within FL systems.

1.2.1. Enhancing Byzantine Robustness and Security in FL

The decentralized nature of FL limits the server’s ability to exert direct control over clients, making the system vulnerable to compromised or malicious participants. These so-called Byzantine clients may engage in poisoning attacks [34], [52], where they submit faulty model updates with the intent to degrade the global model’s performance or inject backdoors. In response, several defense mechanisms have been proposed [52]–

[55]. While these state-of-the-art approaches demonstrate effectiveness in certain scenarios, they are not without limitations.

Recent studies [34], [51] have shown that existing Byzantine-robust FL methods are still vulnerable to local model poisoning since they are forgetful by not tracking information from previous aggregation rounds. Thus, an attacker can efficiently mount an attack by spreading it across time [56]. For example, [38] recently showed that even after infinite training epochs, any aggregation that is neglectful of the past cannot converge to an efficient solution. Furthermore, these methods frequently experience slow convergence and inefficiencies in real-world applications due to the lack of historical data tracking. These limitations underscore the critical need for designing more robust aggregation methods that incorporate historical performance data from clients to more accurately evaluate their trustworthiness, thus enabling faster convergence and more effective mitigation of Byzantine threats in practical FL environments. Beyond traditional approaches, some Byzantine-robust methods employ encryption techniques to secure aggregation processes. For example, SEAR [57], ShieldFL [58] and APFed [59] combine cryptographic protocols with robust aggregation mechanisms to defend against adversarial behaviours. Additionally, recent advancements have introduced several novel Byzantine-robust aggregation strategies aimed at enhancing resilience against poisoning attacks in FL. For instance, BALANCE [39] employs local similarity measures in decentralized settings to effectively filter out malicious updates. FLTracer [60] proposes a FL attack provenance framework that enables the detection of a wide range of attacks and the tracing of Byzantine behaviours. Moreover, FoundationFL [61] enhances robustness by incorporating synthetic updates into traditional robust aggregation mechanisms. Other notable contributions include those proposed by Wu et al. [62], Cao et al. [63], and Xue et al. [64], which offer diverse approaches to improving robustness in FL systems. These methods have been analysed and evaluated alongside our proposed method, demonstrating their respective strengths and limitations in addressing Byzantine threats.

Another security challenge arises from vulnerabilities in the widely used FEDAVG [26] aggregation. While numerous alternative aggregation methods have been proposed in the literature since the inception of federated learning (FL)—such as FedProx [65], FedNova [66], SCAFFOLD [67], and FedDyn [68]—to address specific issues like data heterogeneity and system instability, FEDAVG remains the “de facto” aggregation rule used in many FL tasks [21], [50], [69], due to its simplicity and effectiveness. In FEDAVG the global model is produced from a simple weighted averaging of local updates with weights that represent the amount of data that each party has used for its training. Recent work [39] has shown that FEDAVG is vulnerable to poisoning attacks, as even a single attacker can degrade the global model by sharing faulty local updates of sufficiently large weight. Such attacks become possible because FEDAVG treats all local data points equally. In essence, the aggregation rule, when seen at the granularity of individual training data, resembles the *one person one vote (1p1v)* election rule of modern democratic elections. In this context, the server distributes votes (weights) to a party in accordance with the amount of its

training data, which may be regarded as its population. This, however, may confer an unjust advantage to malicious parties who may have, or falsely claim to have, large training datasets. This demonstrates the need to replace FEDAvg with a more robust fundamental aggregation mechanism that retains the simplicity of FEDAvg.

1.2.2. Privacy leakage in FL

Although FL is designed to enhance privacy by keeping data localized on the client devices, it is not immune to privacy risks. The process of sharing model updates (such as gradients) between clients and the server throughout the training process in FL can inadvertently leak sensitive information. Certain gradient leakage attacks, including *Deep leakage from gradients* (DLG) [70], *Improved DLG* (iDLG) [36] and *Inverting Gradients* [37], exploit model weights to reconstruct input data—such as images or text—through gradient inversion attacks initiated by the server, demonstrating how gradient sharing can reveal private training data. Although other types of privacy attacks exist—such as those leveraging generative adversarial networks (GANs) to reconstruct user data or class representatives [71], [72]—these approaches are generally limited to scenarios where client data is largely homogeneous and an auxiliary dataset is available, or they are evaluated under the assumption that all samples within a class are nearly identical. In this work, we focus specifically on gradient leakage attacks, as they are the most widely studied and serve as the primary threat model in the majority of privacy defense literature [73], [74]. While these gradient leakage attacks have demonstrated the privacy vulnerabilities of standard FL, their effectiveness against pruned FL models remains unexplored. It is intuitively expected that by reducing the number of weights of FL models, the pruning FL should, as a side-effect, also improve their resistance to reconstruction attacks. However, prior work [75]–[79] on model pruning focuses solely on the scalability/accuracy trade-off and thus neither considers nor quantifies the privacy impact of pruning on FL. This gap emphasizes the need for further investigation into the privacy implications of pruned FL, to measure privacy leakage and optimize pruning techniques for both privacy resilience and model accuracy.

1.3. Problem Statement

This thesis addresses critical and impactful challenges in FL, focusing on enhancing Byzantine-robust aggregation mechanisms, developing superior fundamental FL aggregation techniques, and improving privacy preservation in pruned FL models. The research questions of this thesis can be categorized into two key areas: (1) improving the Byzantine robustness of FL to ensure secure and reliable aggregation in the presence of malicious clients, and (2) achieving an effective privacy-utility trade-off in FL, particularly in scenarios where model pruning is employed to optimize performance and communication efficiency.

The first problem arises from the limitations of existing Byzantine-robust aggregation techniques. These methods, designed to filter out malicious updates, often exhibit slow convergence and inefficiencies when applied to real-world data, particularly in scenarios with Non-IID (non-independent, non-identically distributed) data. Furthermore, these techniques typically fail to retain information across aggregation rounds, leaving the system vulnerable to time-distributed attacks, where adversaries gradually spread malicious updates over multiple rounds to avoid detection.

Thus, the first research question is:

Q1. *Can we design a robust aggregation mechanism that retains information from previous rounds to build client reputations, withstand poisoning attacks, and achieves fast convergence in Non-IID data settings?*

The second challenge arises from vulnerabilities in the aggregation process used by most FL systems. The commonly employed FedAvg is vulnerable to poisoning attacks, as even a single attacker can degrade the global model by sharing faulty local updates of sufficiently large weight [39]. Such attacks become possible because FedAvg treats all local data points equally. In essence, the aggregation rule, when seen at the granularity of individual training data, resembles the one person one vote (1p1v) election rule of modern democratic elections. In this context, the server distributes votes (weights) to a party in accordance with the amount of its training data, which may be regarded as its population. This, however, may confer an unjust advantage to malicious parties who may have, or falsely claim to have, large training datasets.

Thus, the second research question is:

Q2. *Can we design a superior aggregation mechanism based on a voting system that assigns fairer weights to clients' contributions, and can this mechanism serve as a fundamental aggregation technique that integrates with other Byzantine-robust and privacy-enhancing methods?*

Finally, after addressing the resilience of FL to poisoning attacks, the focus shifts to its privacy aspects. Model pruning has been proposed as an effective technique for reducing the size and complexity of FL models by removing less significant parameters. By making local models coarser, pruning is intuitively expected to enhance protection against privacy attacks by limiting the amount of information shared between clients and the server. However, the extent of this expected privacy protection has not been thoroughly characterized, nor has it been optimized in conjunction with model utility. Therefore, the third and fourth research questions are formulated as follows:

Q3. *Can we quantify privacy in FL model pruning?*

Q4. *How can we further optimize pruning for privacy, and jointly with accuracy?*

1.4. Contributions of the Thesis

The thesis is structured such that each chapter represents a self-contained adaptation of a corresponding research publication. These chapters are deliberately structured to be autonomously comprehensible, thereby allowing each to be read and understood independently of the others. In striving to maintain the technical integrity and fidelity of the original research works, only minimal modifications have been applied. As a result, readers may encounter variations in notation and repeated elements, such as introductions and literature reviews, across different chapters. The thesis is organised as follows:

Chapter 2. Byzantine-Robust Aggregation Mechanism in Federated Learning

In this chapter, we address research question **Q1**, which investigates how to design a robust aggregation mechanism for FL that (i) retains historical information to build reliable client reputations, (ii) withstands poisoning attacks, and (iii) achieves efficient convergence under Non-IID data distributions. To this end, we propose a novel Byzantine-robust FL aggregation technique that integrates subjective logic reputation with residual-based anomaly detection to enhance robustness against adversarial behaviour.

We evaluate our method in a real-world application scenario: sensitive content classification in a federated setting. Our approach enables the construction of a distributed classifier capable of detecting GDPR-sensitive content (e.g., health-related or tracking-laden websites) without centralized data collection, thus preserving user privacy.

Key contributions of this chapter include:

- Design an FL-based classification method capable of analysing arbitrary URLs. The system can be deployed as a browser extension to (i) alert users about sensitive content and trackers in real time, and (ii) collect user-contributed labels to maintain up-to-date detection models.
- By retaining inter-round aggregation history, the method effectively mitigates slow, distributed attacks and demonstrates faster convergence than existing Byzantine-robust algorithms, making it suitable for practical FL deployments.
- Provide formal analysis and extensive experimental evaluation, demonstrating the proposed method’s superior performance in terms of both accuracy and robustness compared to state-of-the-art baselines.
- We develop and evaluate a proof-of-concept system, *EITR* (standing for ‘Elephant In the Room’ of privacy), which incorporates real user feedback and dynamic label collection for classifying COVID-19-related URLs.

This chapter is a comprehensive adaptation of the paper “Securing Federated Sensitive Topic Classification against Poisoning Attacks” by Chu, T., Garcia-Recuero, A., Iordanou, C., Smaragdakis, G., and Laoutaris, N., presented at the 30th Annual Network and Distributed System Security Symposium (NDSS) 2023.

Building upon the need for robust and reliable aggregation mechanisms established in Chapter 2, the next chapter expands our focus to tackle the fundamental vulnerability of most well-used traditional FL aggregation methods.

Chapter 3. Secure Federated Learning Aggregation Leveraging Quadratic Voting

In this chapter, we address research question **Q2**, which explores how to enhance the security and fairness of the aggregation process in FL. We introduce FEDQV, a novel aggregation method that incorporates quadratic voting (QV) principles to mitigate the influence of malicious clients within traditional FL settings, such as those using FEDAVG. FEDQV introduces adaptive reputation-aware voting budgets and layered defenses, enabling robust aggregation that is both secure and compatible with privacy-preserving techniques.

The main contributions are:

- We formulate FEDQV as a truthful voting-based mechanism grounded in QV principles, augmented with multiple defensive layers to enhance robustness.
- FEDQV dynamically adjusts clients' voting power based on historical behaviour, assigning more influence to reputable clients and limiting the impact of adversaries.
- We provide analytical guarantees under Byzantine settings and validate the method empirically, showing improved robustness and accuracy over FEDAVG.
- FEDQV is compatible with and improves upon several state-of-the-art Byzantine-resilient strategies.
- We demonstrate the integration of FEDQV with secure aggregation protocols (e.g., SECAGG), thereby enhancing defense against privacy threats such as inference or reconstruction attacks.

This chapter is a full adaptation of the papers “FedQV: Leveraging Quadratic Voting in Federated Learning” by Chu, T., Laoutaris, N., et al., presented at ACM SIGMETRICS 2024, and “Strengthening Privacy in Robust Federated Learning through Secure Aggregation” by Chu, T., İşler, D., and Laoutaris, N., presented at the Workshop on Artificial Intelligence Systems with Confidential Computing (AISCC 2024), co-located with the NDSS Symposium 2024.

While Chapter 2 and Chapter 3 focus on improving fairness and robustness in aggregation, the final chapter shifts the attention toward privacy risks in FL, especially in scenarios where sensitive data could be inferred from shared updates in pruned FL.

Chapter 4. Quantifying and Preserving Privacy in Pruned Federated Learning

In this chapter, we address research questions **Q3** and **Q4**, which concern the quantification of privacy leakage in pruned FL and the development of defense mechanisms that

preserve both privacy and model performance. We introduce *PriPrune*, a novel privacy-preserving pruning method designed to mitigate gradient inversion attacks in federated settings.

Key contributions include:

- We derive the first information-theoretic upper bounds for privacy leakage in pruned FL models. These bounds quantify the extent to which sensitive information can be inferred from shared updates. Additionally, we evaluate model vulnerability to advanced inversion attacks, including Deep Leakage from Gradients (DLG) and a novel Sparse Gradient Inversion (SGI) attack.
- Our method introduces a personalized defense mask generated through *Pseudo-Pruning*, enabling clients to prune models locally and upload only pruned versions. This approach reduces the exposure of sensitive gradients while maintaining access to full model parameters locally for future rounds, preserving learning accuracy and efficacy.
- *PriPrune* uses Gumbel Softmax Sampling to dynamically balance privacy and accuracy, allowing fine-grained control over the pruning rate during training.

This chapter is based on our work titled “Privacy-Preserving Pruning Mechanism in Federated Learning” by Chu, T., and Laoutaris, N., ACM Transactions on Modelling and Performance Evaluation of Computing Systems, October 2024

2. BYZANTINE-ROBUST AGGREGATION MECHANISM IN FEDERATED LEARNING

In this chapter we present a Federated Learning (FL) based solution for building a distributed classifier capable of detecting URLs containing sensitive content, i.e., content related to categories such as health, political beliefs, sexual orientation, etc. Although such a classifier addresses the limitations of previous offline/centralised classifiers, it is still vulnerable to poisoning attacks from malicious users that may attempt to reduce the accuracy for benign users by disseminating faulty model updates. To guard against this, we develop a robust aggregation scheme based on subjective logic and residual-based attack detection. Employing a combination of theoretical analysis, trace-driven simulation, as well as experimental validation with a prototype and real users, we show that our classifier can detect sensitive content with high accuracy, learn new labels fast, and remain robust in view of poisoning attacks from malicious users, as well as imperfect input from non-malicious ones.

2.1. Introduction

Most people are not aware that tracking services are present even on sensitive web domains. Being tracked on a cancer discussion forum, a dating site, or a news site with non-mainstream political affinity can be considered an “elephant in the room” when it comes to the anxieties that many people have about their online privacy. The General Data Protection Regulation (GDPR) [24] puts specific restrictions on the collection and processing of sensitive personal data “*revealing racial or ethnic origin, political opinions, religious or philosophical beliefs, or trade union membership, also genetic data, biometric data for the purpose of uniquely identifying a natural person, data concerning health or data concerning a natural persons sex life or sexual orientation*”. So do other public bodies around the world, e.g. in California (California Consumer Privacy Act (CCPA) [25]), Canada [80], Israel [81], Japan [82], and Australia [83].

In a recent paper, Matic et al. [84] showed how to train a classifier for detecting whether the content of a URL relates to any of the above-mentioned sensitive categories. The classifier was trained using 156 thousand sensitive URLs obtained from the Curlie [85] crowdsourced web taxonomy project. Despite the demonstrated high accuracy, this method has limitations that stem from being *centralised* and *tied to a fixed training set*. The first limitation means that the method cannot be used “as is” to drive a privacy-preserving distributed classification system. The second limitation implies that it is not straight-forward to cover new labels related to yet unseen sensitive content. For example, in their work the Health category could be classified with accuracy greater than 90%. However, the training labels obtained from Curlie in 2020 did not include any labels related to the COVID-19

pandemic. Therefore, as will be shown later, this classifier classifies COVID-19 related sites with only 53.13% accuracy.

Federated Learning (FL) [26], [27] offers a natural solution to the above two mentioned limitations, namely, centralized training and training for a fixed training set. FL allows different clients to train their classification models locally without revealing new or existing sensitive URLs that they label, while collaborating by sharing model updates that can be combined to build a superior global classification model. FL has proved its value in a slew of real-world applications, ranging from mobile computing [21], [28], [29] to health and medical applications [20], [30], [31]. However, due to its very nature, FL is vulnerable to so-called *poisoning attacks* [34], [51] mounted by malicious clients that may intentionally train their local models with faulty labels or backdoor patterns, and then disseminate the resulting updates with the intention of reducing the classification accuracy for other benign clients. State-of-the-art approaches for defending against such attacks depend on robust aggregation [39], [52]–[55], [86] which, as we will demonstrate later, are slow to converge, thereby making them impractical for the sensitive-content classification problem that we tackle in this chapter.

Our Contributions: In this chapter, we employ FL for sensitive content classification. We show how to develop a robust FL method for classifying arbitrary URLs that may contain GDPR sensitive content. Such a FL-based solution allows building a distributed classifier that can be offered to end-users in the form of a web browser extension in order to: (i) warn them before and while they navigate into such websites, especially when they are populated with trackers, and (ii) allow them to contribute new labels, e.g., health-related websites about COVID-19, and thus keeping the classifier always up-to-date. To the best of our knowledge this method represents the first use of FL for such task.

Our second major contribution is the development of a reputation score for protecting our FL-based solution from poisoning attacks [34], [51]. Our approach is based on a novel combination of subjective logic [87] with residual-based attack detection. Our third contribution is the development of an extensive theoretical and experimental performance evaluation framework for studying the accuracy, convergence, and resilience to attacks of our proposed mechanism. Our final contribution is the implementation of our methods in a prototype system called *EITR* (standing for “Elephant In the Room” of privacy) and our preliminary experimental validation with real users tasked to provide fresh labels for the accurate classification of COVID-19 related URLs.

Our findings: Using a combination of theoretical analysis, simulation, and experimentation with real users, we:

- Demonstrate experimentally that our FL-based classifier achieves comparable accuracy with the centralised one presented in [84].
- Prove analytically that under data poisoning attacks, our reputation-based robust aggregation built around subjective logic, converges to a near-optimal solution of

the corresponding Byzantine fault tolerance problem under standard assumptions. The resulting performance gap is determined by the percentage of malicious users.

- Evaluate experimentally our solution against state-of-the-art algorithms such as FedAvg [26], Coordinate-wise median [53], Trimmed-mean [53], FoolsGold [54], [55], Residual-based re-weighting [52] and FLTrust [86], and show that our algorithm is robust under Byzantine attacks by using different real-world datasets. We demonstrate that our solution outperforms these popular solutions in terms of convergence speed by a factor ranging from 1.6× to 2.4× while achieving the same or better accuracy. Furthermore, our method yields the most consistent and lowest Attack Success Rate (ASR), with at least 72.3% average improvement against all other methods.
- Validate using our *EITR* browser extension that our FL-based solution can quickly learn to classify health-related sites about COVID-19, even in view of noisy or inconsistent input provided by real users.

2.2. Background

2.2.1. A Centralised Offline Classifier for Sensitive Content

Matic et al. [84] have shown how to develop a text classifier able to detect URLs that contain sensitive content. This classifier is centralised and was developed in order to conduct a one-off offline study aimed at estimating the percentage of the web that includes such content. Despite achieving an accuracy of at least 88%, utilising a high-quality training set meticulously collected by filtering the Curlie web-taxonomy project [85], this classifier cannot be used “as is” to protect real users visiting sensitive URLs populated by tracking services.

2.2.2. Challenges in Developing a Practical Classifier for Users

From offline to online: The classifier in [84] was trained using a dataset of 156 thousand sensitive URLs. Despite being the largest dataset of its type in recent literature, this dataset is static and thus represents sensitive topics up to the time of its collection. This does not mean, of course, that a new classifier trained with this data would never be able to accurately classify new URLs pertaining to those sensitive categories. This owes to the fact that categories such as Health, involve content and terms that do not change radically with time. Of course, new types of sensitive content may appear that, for whatever reason, may not be so accurately classified using features extracted from past content of the same sensitive category. For instance, consider a new Health-related topic such as COVID-19, which introduced terms like "social distancing" and "pandemic protocols" that were absent from pre-2020 content. A classifier trained solely on pre-2020 data may misclassify

or fail to detect the sensitivity of URLs containing such novel terms. This simple example highlights the broader challenge: to maintain high classification accuracy over time, a classifier must be updated as new sensitive content and vocabulary emerge. Indeed, as we will see later in Figure 2.17, the classifier of [84] classifies accurately as Health only 53.13% of the COVID-19 URLs with which we tested it. This observation further confirms our point, since the dataset of [84] corresponds to content generated before the first months of 2020, during which COVID-19 was not yet a popular topic. Therefore, we need to find a way to update an existing classifier so that it remains accurate as new sensitive content appears.

From centralised to distributed: A natural way to keep a classifier up-to-date is to ask end-users to label new sensitive URLs as they encounter them. End-users can report back to a centralised server such URLs which can then be used to retrain the classification model. This, however, entails obvious privacy challenges of “Catch-22” nature, since to protect users by warning them about the presence of trackers on sensitive URLs, they would first be required to report to a potentially untrusted centralised server that they visit such URLs. Even by employing some methods for data scarcity, e.g., semi-supervised learning, the manual labelling from users remains sensitive and may be harmed by the untrusted server. Federated Learning, as already mentioned, is a promising solution for avoiding the above Catch22 by conducting a distributed, albeit, privacy-preserving, model training. In an FL approach to our problem, users would label new URLs locally, e.g., a COVID-19 URL as Health, retrain the classifier model locally, and then send model updates, not labelled data, to a centralised server that collects such updates from all users, compiles and redistributes the new version of the model back to them. In Section 2.3.3 we show how to develop a distributed version of the sensitive topic classifier of [84] using FL. The trade-off of using FL, is that the distributed learning group becomes vulnerable to attacks, such as “label-flipping” poisoning attacks discussed in Section 2.4. This paper develops a reputation scheme for mitigating such attacks. Other types of attacks and measures for preserving the privacy of users that participate in a FL-based classification system for sensitive content are discussed in Section 2.6.

2.2.3. Related Work

Privacy preserving crowdsourcing: Similar challenges to the ones discussed in the previous paragraph have been faced in services like the *Price \$ heriff* [88] and *eyeWnder* [89] that have used crowdsourcing to detect online price discrimination and targeted advertising, respectively. Secure Multi-Party Computation (SMPC) techniques such as private k -means [90] are used to allow end-users to send data in a centralised server in a privacy-preserving manner. The centralised computation performed by *Price \$ heriff* and *eyeWnder* is not of ML nature, thus leaving data anonymisation as the main challenge, for which SMPC is a good fit. Classifying content as sensitive or not is a more complex ML-based algorithm for which FL is a more natural solution than SMPC.

General works on FL: FL [26], [27] is a compelling technique for training large-scale distributed machine learning models while maintaining security and privacy. The motivation for FL is that local training data is always kept by the clients and the server has no access to the data. Due to this benefit that alleviates privacy concerns, several corporations have utilised FL in real world services. In mobile devices, FL is used to predict keyboard input [21], human mobility [28] and behaviour for the Internet of Things [29]. FL is also applied in healthcare to predict diseases [20], [30], detect patient similarity [31] while overcoming any privacy constraints. For the classification, FL is not only implemented for image classification [91] but also text classification [92].

Resilience to poisoning attacks: Owing to its nature [34], [51], FL is vulnerable to poisoning attacks, such as label flipping [52] and backdoor attacks [34]. Therefore, several defence methods have been developed [52]–[55]. While these state-of-the-art approaches perform excellently in some scenarios, they are not without limitations. First, they are unsuitable for our sensitive content classification, which necessitates that a classifier responds very fast to “fresh” sensitive information appearing on the Internet. In existing methods, the primary objective is to achieve a high classification accuracy. This is achieved via statistical analysis of client-supplied model updates and discarding of questionable outliers before the aggregation stage. However, since the server distrusts everyone by default, even if an honest client discovers some fresh sensitive labels, its corresponding updates may be discarded or assigned low weights, up until more clients start discovering these labels. This leads to a slower learning rate for new labels.

Second, recent studies [34], [51] have shown that existing Byzantine-robust FL methods are still vulnerable to local model poisoning since they are forgetful by not tracking information from previous aggregation rounds. Thus, an attacker can efficiently mount an attack by spreading it across time [56]. For example, [38] recently showed that even after infinite training epochs, any aggregation which is neglectful of the past cannot converge to an efficient solution.

The preceding studies demonstrate the importance of incorporating clients’ previous long-term performance in evaluating their trustworthiness. Few recent studies have considered this approach [38], [86]. In [38], the authors propose leveraging historical information for optimisation, but not for assessing trustworthiness. In [86], a trust score is assigned to each client model update according to the cosine similarity between the client’s and server’s model updates, which is trained on the server’s root dataset (details in Section 2.4.1). However, it is impractical for a server to obtain additional data, such as a root dataset, in order to train a server-side model. In addition, because the server collects root data only once and does not update it throughout the training process, when new types of content emerge over time, the root data may become stale thereby harming the classifier’s performance. Other recent studies employ spectral analysis [33], differential privacy [93], and deep model inspection [94] to guard against poisoning attacks, but, again, they do not use historical information to assess the reliability of clients. To measure client trustworthiness without collecting additional data at the server, in the next sections

Table 2.1: Notations

Abbreviation	Description
M	the total number of clients
N	the number of parameters of global model
Q	the number of samples of each client
T	the total number of iterations
$w_{i,n}^t$	the n -th parameter from client i in t iteration
$x_{i,n}^t$	the ranking of $w_{i,n}^t$ in w_n^t
A_n, B_n	the slope and intercept of repeated median linear regression
$e_{i,n}^t$	the normalised residual of the n -th parameter from client i in t iteration

we show how to design a robust aggregation method to generate reputation automatically based on the historical behaviours of clients, which is a more realistic approach for a real FL-based decentralised system implemented as a browser extension for clients.

2.3. A Robust FL Method for Classifying Sensitive Content on the Web

In this section, we first show how to build an FL-based classifier for sensitive content. Then we design a reputation score for protecting against poisoning attacks. We analyse theoretically the combined FL/reputation-based solution and establish convergence and accuracy guarantees under common operating assumptions.

2.3.1. FL Framework for Classifying Sensitive Content

Table 2.1 presents the notation that we use in the remainder of the paper. In FL, clients provide the server updated parameters from their local model, which the server aggregates to build the global model. Suppose we have M clients participating in our classification training task and the dataset $\mathcal{D} = \bigcup_{i=1}^M \mathcal{D}_i$. In our task, the clients' data is the textual content of URLs stripped of HTML tags. The objective function and the optimization method used are the same as those defined in Section 1.1, where w^* is the optimal weight.

During the broadcast phase, the server broadcasts the classification task and training instructions to clients. Then, the clients apply the following standard pre-processing steps on the webpage content, that is, transformation of all letters in lowercase and the removal of stop words. Next, the clients extract the top one thousand features utilising the Term Frequency-Inverse Document Frequency (TF-IDF) [95] as in [84]. At iteration t , the client i receives the current global model M_{global} and then following the training instructions from the server, trains the local model on its training data $\mathcal{D}_i$ and optimises $w_i^t = \arg \min_w \mathcal{L}_i(w_i^t)$, as defined in **Local Training** in Section 1.1. Here $\mathcal{L}_i(w) := \mathbb{E}_{\mathcal{D}_i \sim \mathcal{X}}[l(w; \mathcal{D}_i)] = \frac{1}{Q_i} \sum_{j=1}^{Q_i} l(w; \mathcal{D}_i^j)$, $\mathcal{D}_i^j$ and Q_i means the j -th sample and the number of samples of the client i respectively, and r is the learning rate.

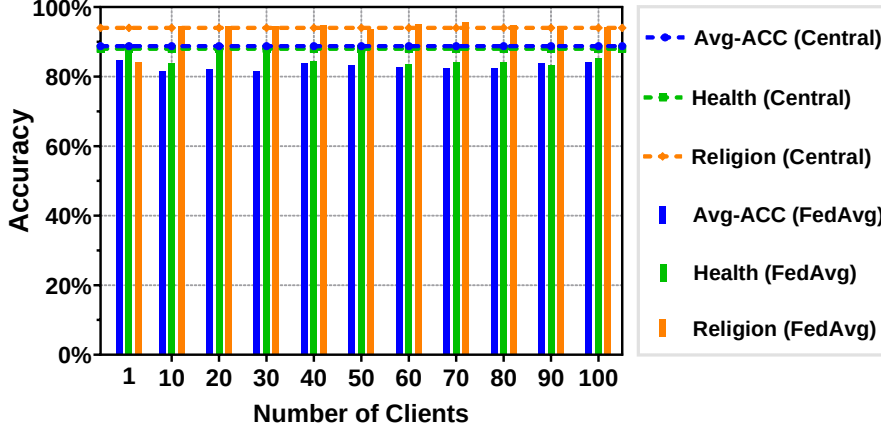


Figure 2.1: Accuracy of FL classifiers and centralised classifiers in Health, Religion and all categories.

In every iteration, after finishing the training process the clients send back their local updates to the server. Then, the server computes a new global model update by combining the local model updates via an aggregation method AGG as follows:

$$w^t = \text{AGG}\left(\{w_i^t\}_{i=1}^M\right)$$

Here we utilise FEDAVG which is introduced in Section 1.1: $w^t = \sum_{i=1}^M \frac{Q_i}{Q} w_i^t$. We introduce other robust aggregation methods in the next subsection. Subsequently, the server uses the global model update to renew the global model M_{global} .

Using the above FL-based framework we first evaluate how the number of users in the system affects the average accuracy of the classifier. The results for the sensitive categories, Health and Religion, as well as the overall average accuracy (Avg-ACC) are depicted in Figure 2.1. A first observation is that when a fixed size dataset is divided into multiple segments and distributed to more clients, the model’s accuracy decreases since each client has less data for training. Compared to the centralised classifier, using the same data, the accuracy of the FL classifier is slightly lower, which is expected when the training is distributed to a larger number of clients. Overall, we observe that the average accuracy difference between the FL and the centralised classifier is 5.76%, and this remains steady as the number of clients grows. In addition, Looking at the different sensitive categories (Health and Religion), we see the FL-based classifier achieves an accuracy very close to the corresponding one of the centralised classifier for these categories (on average 0.8533 vs. 0.88 and 0.9366 vs. 0.94, respectively).

2.3.2. A Reputation score for Thwarting Poisoning Attacks

Figure 2.2 shows an overview of our reputation-based aggregation algorithm consisting of three components: the *attack detection scheme*, the *reputation model*, and the *aggregation module*. The attack detection scheme re-scales and rectifies damaging updates received

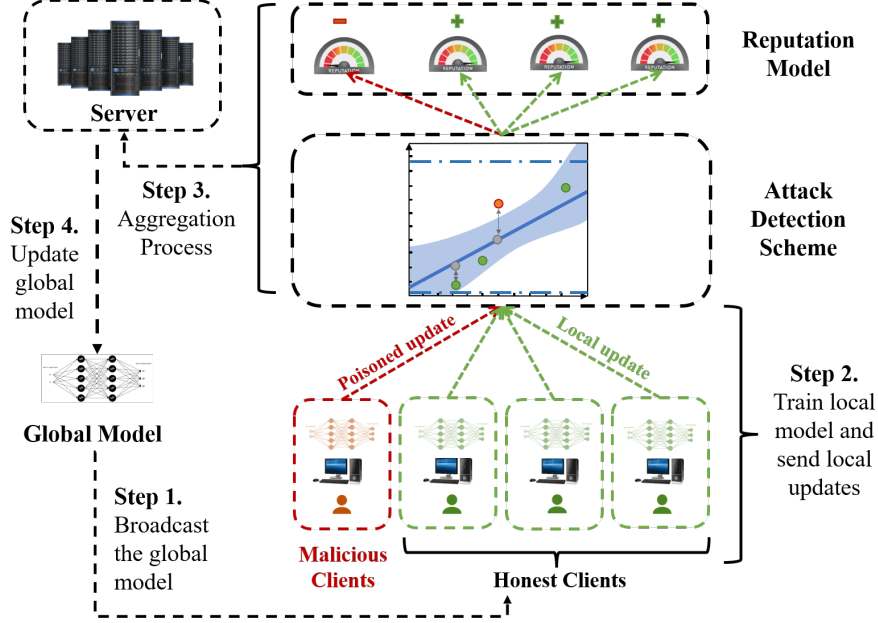


Figure 2.2: Overview of reputation-based aggregation algorithm.

from clients. Then, the reputation model calculates each client’s reputation based on their past detection results. Finally, the aggregation module computes the global model by averaging the updates of the clients using their reputation scores as weights. We detail each component in the following subsections.

Attack Detection Scheme

Our attack detection scheme aims to reduce the impact of suspicious updates by identifying them and applying a rescaling algorithm. At every iteration, when model updates from clients arrive at the server, we apply Algorithm 1 there to rescale the range of values for those parameters in the updates.

This restriction on the value range aims not only to minimise the impact of abnormal updates from attackers but also to limit the slope for the repeat median regression. Considering the n -th parameter in round t from all the participants, we calculate the standard deviation $\sigma(w_{i,n}^t)$ of this series. Then we sort them in ascending order and determine the range by subtracting the lowest value from the highest one. If the result is above the threshold ϖ , we rescale the highest and lowest value by deducting and adding its standard deviation respectively to further bound their range.

Then, a robust regression [96] is carried out to identify outliers among the updates in the current round. Outlier detection is a well-established topic in statistics. Robust regression methods often handle outliers by using the median estimators. Median-based aggregation methods have a rich and longstanding history in the area of robust statistics [97]. However, the methods developed by the traditional robust statistics can only withstand a small fraction of Byzantine clients, resulting in a low “breakdown point” [98]. Different

Algorithm 1: Rescale(w)

Input : $\{w_{i,n}^t\} \leftarrow$ Local Model parameters in round t

Output: $\{w_{i,n}^t\}$ with range of value less than ϖ

```
1 for  $n \leftarrow 1$  to  $N$  do
2   // Determine the maximum range
3    $Rm = \max w_{i,n}^t - \min w_{i,n}^t = w_{i,n}^{t,(Max)} - w_{i,n}^{t,(Min)}$ 
4   while  $Rm > \varpi$  do
5     // Rescale range based on standard deviation.
6      $w_{i,n}^{t,(Max)} := w_{i,n}^{t,(Max)} - \sigma(w_{i,n}^t);$ 
7      $w_{i,n}^{t,(Min)} := w_{i,n}^{t,(Min)} + \sigma(w_{i,n}^t);$ 
8     // Updated  $Rm$ .
9      $Rm = \max w_{i,n}^t - \min w_{i,n}^t = w_{i,n}^{t,(Max)} - w_{i,n}^{t,(Min)}$ 
10  end while
11 end for
```

from many other variations of the univariate median, the repeated median [96] is impervious to atypical points even when their percentage is nearly 50%. The repeated median is defined as a modified U-statistic and the concept behind it is to utilise a succession of partial medians for computing approximation $\hat{\tau}$ of the parameter τ : For $k \in \mathbb{N}$, the value of parameter $\tau(z_1, \dots, z_k)$ is determines by subset of k data points $z_1, \dots, z_k$.

$$\hat{\tau} = \text{median}_{z_1} \left\{ \text{median}_{z_2 \notin \{z_1\}} \left\{ \text{median}_{z_k \notin \{z_1, \dots, z_{k-1}\}} \tau(z_1, \dots, z_k) \right\} \right\} \quad (2.1)$$

In our case, the intercept $\hat{A}$ and slop $\hat{B}$ are estimated by repeated median as bellow:

$$\hat{B}_n = \text{median}_i \left\{ \text{median}_{i \neq j} \{B_n(i, j)\} \right\} \quad (2.2)$$

$$\hat{A}_n = \text{median}_i \{w_{i,n} - \hat{B}_n x_{i,n}\} \quad (2.3)$$

where $B_n(i, j) = \frac{w_{j,n} - w_{i,n}}{x_{j,n} - x_{i,n}}$, $x_{i,n}$ represents the index of $w_{i,n}$ in w_n which is sorted in ascending order.

Next, we employ the IRLS scheme [99] to generate each parameter's confidence score $s_{i,n}^t$ based on the normalised residual from repeated median regression, which is also utilised in a residual-based aggregation method [52]:

$$s_{i,n}^t = \frac{\sqrt{1 - \text{diag}(H_n^t)}}{e_{i,n}^t} \Psi \left(\frac{e_{i,n}^t}{\sqrt{1 - \text{diag}(H_n^t)}} \right) \quad (2.4)$$

where confidence interval $\Psi(x)$:

$$\Psi(x) = \max\{-\lambda \sqrt{2/M}, \min(\lambda \sqrt{2/M}, x)\}$$

and the hat matrix H_n^t :

$$H_n^t = x_n^t (x_n^{tT} x_n^t)^{-1} x_n^{tT}$$

$$\text{with } e_{i,n}^t = \frac{25(M-1)(w_{i,n}^t - \hat{B}_n x_{i,n}^t - \hat{A}_n)}{37(M+4)\text{median}_i(|w_{i,n}^t - \hat{B}_n x_{i,n}^t - \hat{A}_n|)}.$$

The distance between the point and the robust line is described by the confidence score derived from the normalised residual, which can be used to evaluate if the point is anomalous. Following the computation of the parameter's confidence score, and in light of the fact that some attackers want to generate updates with abnormal magnitudes in order to boost the damage, a useful protection is to identify low confidence values based on a threshold δ . Once the server recognises an update $\mathbf{w}_{i,n}^t$ of the client i with confidence values less than δ , rather than altering this update to the repeat median estimation, our technique replaces it with the median of $\mathbf{w}_n^t$, as follows:

$$\mathbf{w}_{i,n}^t = \begin{cases} \mathbf{w}_{i,n}^t & \text{if } s_{i,n}^t > \delta \\ \text{median}_i \{ \mathbf{w}_{i,n}^t \} & \text{if } s_{i,n}^t \leq \delta \end{cases} \quad (2.5)$$

With the above, not only we bound the range of updates, but also improve the aggregation by introducing a robustness estimator.

Reputation Model

During the aggregation phase in FL, we use a subjective logic model to produce client reputation scores. The subjective logic model is a subset of probabilistic logic that depicts probability values of belief and disbelief as degrees of uncertainty [87]. In the subjective logic model, reputation score R_i^t for client i in t iteration correlates to a subjective belief in the dependability of the client's behaviour [100], as measured by the belief metric opinion τ_i^t [101]. An opinion is comprised of three elements: belief b_i^t , disbelief d_i^t and uncertainty u_i^t , with restrictions that $b_i^t + d_i^t + u_i^t = 1$ and $b_i^t, d_i^t, u_i^t \in [0, 1]$. The reputation score may be calculated as the expected value of an opinion $E(\tau_i^t)$ which can be regarded as the degree of trustworthiness in client i . As a result, the value of the client's reputation is defined as follows:

$$R_i^t = E(\tau_i^t) = b_i^t + au_i^t \quad (2.6)$$

where $a \in [0, 1]$ denotes the prior probability in the absence of belief, which reflects the fraction of uncertainty that may be converted to belief. On the other side, distinct observations determined by the rectification phase in our Algorithm 2 are used to count belief, disbelief, and uncertainty opinions. The positive observation denoted by P_i^t indicates that the update $\mathbf{w}_{i,n}^t$ is accepted ($s_{i,n}^t > \delta$), whereas a negative observation denoted by N_i^t indicates that the update is rejected ($s_{i,n}^t \leq \delta$). As a consequence, the positive observations boost the client's reputation, and vice versa. To penalise the negative observations from the unreliable updates, a higher weight η is assigned to negative observations than the weight κ to positive observations with constrain $\eta + \kappa = 1$. Therefore, in Beta distribution below:

$$Beta(p|\alpha, \beta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1} (1-x)^{\beta-1} \quad (2.7)$$

with the constraints $0 \leq x \leq 1$, parameters $\alpha > 0, \beta > 0$, and $x \neq 0$ if $\alpha < 1$ and $x \neq 1$ if $\beta < 1$. The parameters α and β that represent positive and negative observations respectively, can be expressed as below

$$\begin{cases} \alpha = \kappa P_i^t + Wa \\ \beta = \eta N_i^t + W(1-a) \end{cases} \quad (2.8)$$

where W is the non-information prior weight and the default value is 2 [87].

As a consequence, the expected value of Beta distribution, which also stands for reputation value, can be calculated as follows:

$$E(Beta(p|\alpha, \beta)) = \frac{\alpha}{\alpha + \beta} = \frac{\kappa P_i^t + Wa}{\kappa P_i^t + \eta N_i^t + W} = R_i^t \quad (2.9)$$

Based on (2.6) and (2.9), we can derive

$$\begin{cases} b_i^t = \frac{\kappa P_i^t}{\kappa P_i^t + \eta N_i^t + W} \\ d_i^t = \frac{\eta N_i^t}{\kappa P_i^t + \eta N_i^t + W} \\ u_i^t = \frac{W}{\kappa P_i^t + \eta N_i^t + W} \end{cases} \quad (2.10)$$

In addition, in order to take the client's historical reputation values in previous rounds into consideration, a time decay mechanism is included to lower the relevance of past performances without disregarding their influence. In other words, the reputation value from the most recent iteration contributes the most to the reputation model. We use exponential time decay in our model, as shown below:

$$\theta_{j,t} = \exp(-c(t-j)) \quad (2.11)$$

where $\exists c > 0, j \in [\tilde{s}, t], \tilde{s} = \max(t-s, 0)$. We include a sliding window with a window length s that allows us to get a reputation for a certain time interval rather than the entire training procedure. We remove expired tuples with timestamps outside the window period during computation since they cannot provide meaningful information for the reputation. Hence, the final reputation score $\tilde{R}_i^t$ can be expressed as:

$$\tilde{R}_i^t = \frac{\sum_{j=\tilde{s}}^t \theta_{j,t} R_i^j}{\sum_{j=\tilde{s}}^t \theta_{j,t}} \quad (2.12)$$

To demonstrate how the reputation model evolves, we consider four scenarios where each client: (i) only attacks once at the same iteration, (ii) attacks continuously after launching an attack at the same iteration, (iii) only attacks once at different iteration, (iv) attacks continuously after launching an attack at different iteration. Here, clients conduct attacks as described in Section 2.4.1 by utilising polluted data while training the local model, whereas the server uses our attack detection mechanism to identify these attacks.

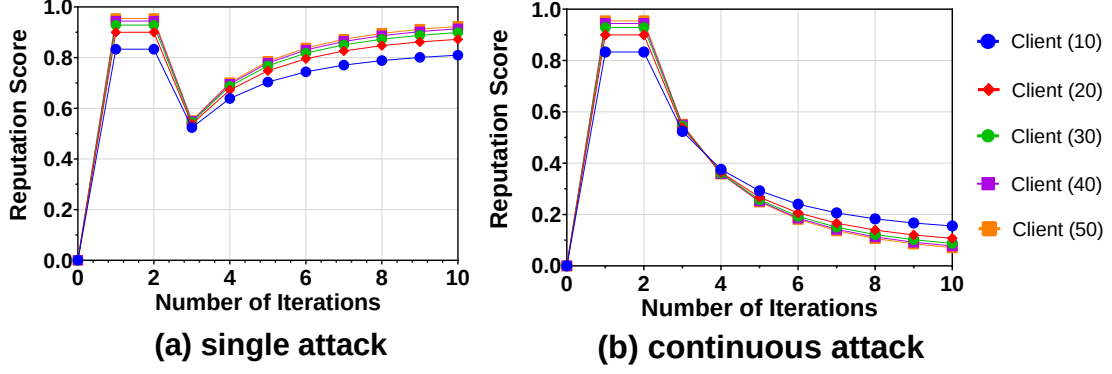


Figure 2.3: The decay of reputation score in Client (X) with X model parameters when they (a) attack once at 3rd iteration and (b) attack continuously at and after the 3rd iteration.

Figure 2.3 displays the first two scenarios (i)-Figure 2.3a and (ii)-Figure 2.3b, respectively, with Client X , who has X parameters in their local models, under single and continuous attack. In Figure 2.3a, all of the clients only attack once at the third iteration. When they start attacking, their reputation score plummets dramatically. In both scenarios, we observe that the client who has more parameters has a larger relative decline in reputation score. This is also compatible with Corollary 2.3.8 in Section 2.3.3, that is, increasing the number of parameters N in the global model results in a lower error rate.

Figure 2.4 shows the last two scenarios (iii) and (iv) respectively with clients, who have 20 parameters in their local models, under single and continuous attack. In Figure 2.4a, Client X only launches an attack at X iteration. We observe that only one attack would lead to at least a 25.11% relative decrease in reputation score. In Figure 2.4b, Client X launches an attack at X iteration and keeps attacking in the following iterations. We observe in the end that 80% of their reputation scores are below 0.5, which is approximately half of the reputation score of honest clients, implying that the damage that they can inflict throughout the aggregation process is considerably decreased.

In addition, we consider a scenario in which an attacker spreads out the poisoning over a longer time duration while using a higher number of model parameters. Figure 2.5 (left) depicts the decay of reputation scores of four malicious clients over 40 iterations, each with a different model size—1k, 10k, 100k, and 1 million parameters—launching attacks at iterations 10, 50, and 90. Figure 2.5 (right) depicts the decay of reputation score of four malicious clients, each with 1 million parameters, performing the first attack at iteration 10 and a second attack at iteration $10 + X$, where X varies from 40 to 70. These figures show that even if attackers spread out poisoning over multiple iterations and then try to recover their reputation score by acting benignly, our detection scheme can still identify them. This is because our attack detection and reputation schemes work in sequence. The attack detection scheme detects malicious updates without considering any reputation scores and rectifies them to mitigate damage. Then, the reputation scheme

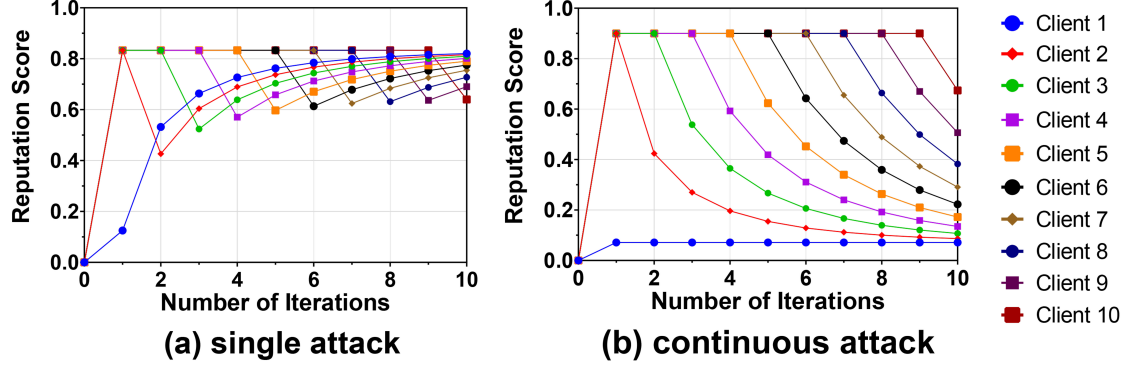


Figure 2.4: The decay of reputation score in Client X with the same model parameters when they (a) attack once at X iteration and (b) attack continuously after starting to attack at X iteration.

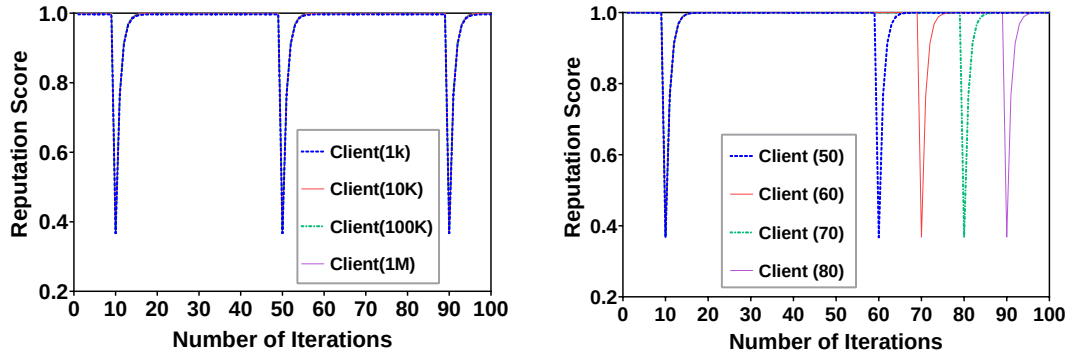


Figure 2.5: The decay of reputation scores of malicious clients under multi-iteration poisoning attacks. (Left) Four malicious clients with varying model sizes (1k, 10k, 100k, and 1M parameters) perform attacks at iterations 10, 50, and 90. (Right) Four malicious clients, each with 1M parameters, perform two attacks: the first at iteration 10 and the second at iteration $10 + X$, where $X \in \{40, 50, 60, 70\}$.

modifies the reputation scores based on the detection results. Also, attackers that employ a higher number of model parameters suffer a slightly higher reduction of reputation, which is consistent with Corollary 2.3.8.

Aggregation Algorithm

Algorithm 2 explains our aggregation method based on the attack detection scheme and subjective logic reputation model. First, the server sends all clients the pre-trained global model with initial parameters. Then, using their own data samples, clients train the global model locally and send the trained parameters back to the server. At this point, the server executes the attack detection scheme. In round t , if the n -th update parameter $\mathbf{w}_{i,n}^t$ from the client i has been rectified by the attack detection scheme in Section 2.3.2 to the median value, the server regards it as a negative observation, whereas no rectification represents a

positive observation. Then, the server punishes the negative observation by reducing the corresponding client's reputation. Both types of observations are accumulated through all the N parameters of client i to obtain the reputation value $\tilde{R}_i^t$ in t round for client i so as to all the other clients. The server would conduct Min-Max normalisation to obtain $\bar{R}^t$ after receiving the reputation values $\tilde{R}^t$ of all the clients in t round.

After the server gets correction updates and the normalised reputation of each client, it aggregates the updates using average weighted reputation as the weights to get our global model updates for the current iteration. In this way, even over many training rounds, the attackers are still incapable of shifting parameters notably from the target direction and this ensures the quality of the resulting global model as will be demonstrated experimentally and analytically next.

2.3.3. Theoretical Guarantees

We prove the convergence of our reputation-based aggregation method. Our major results are Theorem 2.3.7 and Corollary 2.3.8, which state that convergence is guaranteed in bounded time. Regarding the performance of our algorithm in terms of metric average accuracy and convergence, we show that it is consistent with our theoretical analysis. We start by stating our assumptions, which are standard and common for such types of results, and per recent works such as [53], [102].

Assumption 2.3.1 (Smoothness). The loss functions are L -smooth, which means they are continuously differentiable and their gradients are Lipschitz-continuous with Lipschitz constant $L > 0$, whereas:

$$\begin{aligned} \forall \mathbf{w}_1, \mathbf{w}_2 \in \mathbf{R}^d \\ \|\nabla \mathcal{L}(\mathbf{w}_1) - \nabla \mathcal{L}(\mathbf{w}_2)\|_2 &\leq L \|\mathbf{w}_1 - \mathbf{w}_2\|_2 \\ \|\nabla \ell(\mathbf{w}_1; \mathcal{D}) - \nabla \ell(\mathbf{w}_2; \mathcal{D})\|_2 &\leq L \|\mathbf{w}_1 - \mathbf{w}_2\|_2 \end{aligned}$$

Assumption 2.3.2 (Bounded Gradient). The expected square norm of gradients $\mathbf{w}$ is bounded:

$$\forall \mathbf{w} \in \mathbf{R}^d, \exists \mathcal{G}_{\mathbf{w}} < \infty, \mathbb{E} \|\nabla \ell(\mathbf{w}; \mathcal{D})\|_2^2 \leq \mathcal{G}_{\mathbf{w}}$$

Assumption 2.3.3 (Bounded Variance). The variance of gradients $\mathbf{w}$ is bounded:

$$\forall \mathbf{w} \in \mathbf{R}^d, \exists \mathcal{V}_{\mathbf{w}} < \infty, \mathbb{E} \|\nabla \ell(\mathbf{w}; \mathcal{D}) - \mathbb{E}(\nabla \ell(\mathbf{w}; \mathcal{D}))\|_2^2 \leq \mathcal{V}_{\mathbf{w}}$$

Assumption 2.3.4 (Convexity). The loss function $\mathcal{L}(\mathbf{w})$ are μ -strongly convex:

$$\begin{aligned} \exists \mu > 0, \forall \mathbf{w}_1, \mathbf{w}_2 \in \mathbf{R}^d, \nabla \mathcal{L}(\mathbf{w}^*) = 0 \\ \mathcal{L}(\mathbf{w}_1) - \mathcal{L}(\mathbf{w}_2) &\geq \langle \nabla \mathcal{L}(\mathbf{w}_2), \mathbf{w}_1 - \mathbf{w}_2 \rangle + \frac{\mu}{2} \|\mathbf{w}_1 - \mathbf{w}_2\|_2^2 \end{aligned}$$

Algorithm 2: Aggregation Algorithm

Server :

Input : $w^0 \leftarrow$ Pretrained Model

$\kappa, \eta, a, W, c, s \leftarrow$ Reputation parameters

Output: Global model M_{global} with w^T

```
1 for Iteration  $t \leftarrow 1$  to  $T$  do
2   // Broadcast global model to clients
3   send( $w^{t-1}$ );
4   // Wait until all updates arrive
5   receive( $w^t$ );
6   // Rescale parameters by Algorithm 1
7    $\bar{w}^t \leftarrow \text{Rescale}(w^t)$ ;
8   for  $n \leftarrow 1$  to  $N$  do
9     for  $i \leftarrow 1$  to  $M$  do
10      // Compute parameter confidence
11       $s_{i,n}^t = \text{Eq 2.4}(\bar{w}_{i,n}^t)$ ;
12      // Rectify abnormal parameters
13       $w_{i,n}^t := \text{Eq 2.5}(s_{i,n}^t, \delta)$ ;
14      record ( $P_i^t, N_i^t$ );
15    end for
16  end for
17  for  $i \leftarrow 1$  to  $M$  do
18    // Calculate reputation score
19     $\tilde{R}_i^t = \text{Eq 2.12}(P_i^t, N_i^t, \kappa, \eta, a, W, c, s)$ ;
20  end for
21  // Normalisation
22   $\bar{R}^t \leftarrow \text{Norm}(\tilde{R}^t)$ ;
23  for  $n \leftarrow 1$  to  $N$  do
24    // Update the parameters
25     $w_n^t := \sum_{i=1}^M \frac{\bar{R}_i^t}{\sum_{i=1}^M \bar{R}_i^t} w_{i,n}^t$ ;
26  end for
27  // Obtain parameters for global model
28   $w^t := [w_1^t, \dots, w_n^t]$ ;
29 end for
```

Client :

```
1 for Client  $i \leftarrow 1$  to  $M$  do in parallel
2   receive( $w^{t-1}$ );
3   // Train local model
4    $w_i^t \leftarrow w_i^{t-1} - r \frac{\partial \ell_i(w_i^{t-1})}{\partial w}$ ;
5   send( $w_i^t$ );
6 end forpar
```

Suppose the percentage of attackers in the whole system is p , and all the clients in the system participate in every training iteration. r is the learning rate ($r < \frac{1}{L}$) and $\hat{Q} = \max \{Q_i\}_{i=1}^M$. $\forall \mathbf{w} \in \mathcal{W}$, we denote

$$\mathbf{m}_i(\mathbf{w}^t) = \begin{cases} * & \text{if } i \in \text{malicious clients} \\ \nabla l_i(\mathbf{w}^t; \mathcal{D}) & \text{if } i \in \text{honest clients} \end{cases}$$

where $*$ stands for an arbitrary value from the malicious clients.

$$\begin{aligned} \mathbf{m}(\mathbf{w}^t) &= \sum_{i=1}^M \bar{R}_i \mathbf{m}_i(\mathbf{w}^t) \\ \text{s.t. } \bar{R}_i &= \frac{\tilde{R}_i^t}{\sum_{i=1}^M \tilde{R}_i^t}, \quad \sum_{i=1}^M \bar{R}_i = 1, \quad \bar{R}_i \in (0, 1) \end{aligned}$$

The following are the lemmas we utilise in the proof of Theorem 2.3.7.

Lemma 2.3.5. *From Assumption 2.3.1 and 2.3.4, $\mathcal{L}(\mathbf{w})$ is L -smooth and μ -strongly convex. Then $\forall \mathbf{w}_1, \mathbf{w}_2 \in \mathcal{W}$, one has*

$$\langle \nabla \mathcal{L}(\mathbf{w}_1) - \nabla \mathcal{L}(\mathbf{w}_2), \mathbf{w}_1 - \mathbf{w}_2 \rangle \geq \frac{L\mu}{L+\mu} \|\mathbf{w}_1 - \mathbf{w}_2\|_2^2 + \frac{1}{L+\mu} \|\nabla \mathcal{L}(\mathbf{w}_1) - \nabla \mathcal{L}(\mathbf{w}_2)\|_2^2 \quad (2.13)$$

Lemma 2.3.6. *The difference between $\mathbf{m}(\mathbf{w})$ and $\nabla \mathcal{L}(\mathbf{w})$ is bounded in every iteration:*

$$\|\mathbf{m}(\mathbf{w}) - \nabla \mathcal{L}(\mathbf{w})\|_2 \leq \|\mathbf{m}_0(\mathbf{w}) - \nabla \mathcal{L}(\mathbf{w})\|_2 + \sqrt{N} \Delta_1 \quad (2.14)$$

where:

$$\Delta_1 = \frac{M(\varpi(M-1) + \frac{2E}{\sqrt{M}\delta})}{\frac{Wa(M-1)(\kappa N+W)}{(\eta N+W)(\kappa N+Wa)} + 1}$$

$$E = \sup \left\{ \frac{37\sqrt{2}\lambda(M+4)}{25(M-1)} \text{median}_i \left\{ |w_{i,n}^t - \hat{B}_n x_{i,n}^t - \hat{A}_n| \right\} \right\}$$

and

$$\mathbf{m}_0(\mathbf{w}) := \text{median}_i \{\mathbf{m}_i(\mathbf{w})\}$$

Proof of Lemma 2.3.5 Let $g(\mathbf{w}) = \mathcal{L}(\mathbf{w}) - \frac{\varsigma}{2} \|\mathbf{w}\|_2^2$. Based on the Assumption 2.3.4, $g(\mathbf{w})$ is $(L - \varsigma)$ -strongly convex. From Equation 3.6 of [103], we have

$$\langle \nabla \mathcal{L}(\mathbf{w}_1) - \nabla \mathcal{L}(\mathbf{w}_2), \mathbf{w}_1 - \mathbf{w}_2 \rangle \geq \frac{1}{L} \|\nabla \mathcal{L}(\mathbf{w}_1) - \nabla \mathcal{L}(\mathbf{w}_2)\|_2^2 \quad (2.15)$$

Hence,

$$\langle \nabla g(\mathbf{w}_1) - \nabla g(\mathbf{w}_2), \mathbf{w}_1 - \mathbf{w}_2 \rangle \geq \frac{1}{L - \varsigma} \|\nabla g(\mathbf{w}_1) - \nabla g(\mathbf{w}_2)\|_2^2 \quad (2.16)$$

Now we have

$$\begin{aligned} & \langle \nabla \left(\mathcal{L}(\mathbf{w}_1) - \frac{\varsigma}{2} \|\mathbf{w}_1\|_2^2 \right) - \nabla \left(\mathcal{L}(\mathbf{w}_2) - \frac{\varsigma}{2} \|\mathbf{w}_2\|_2^2 \right), \mathbf{w}_1 - \mathbf{w}_2 \rangle \\ & \geq \frac{1}{L + \mu} \left\| \nabla \left(\mathcal{L}(\mathbf{w}_1) - \frac{\varsigma}{2} \|\mathbf{w}_1\|_2^2 \right) - \nabla \left(\mathcal{L}(\mathbf{w}_2) - \frac{\varsigma}{2} \|\mathbf{w}_2\|_2^2 \right) \right\|_2^2 \end{aligned} \quad (2.17)$$

Therefore,

$$\begin{aligned} & \langle \nabla \mathcal{L}(\mathbf{w}_1) - \nabla \mathcal{L}(\mathbf{w}_2), \mathbf{w}_1 - \mathbf{w}_2 \rangle - \langle \varsigma \mathbf{w}_1 - \varsigma \mathbf{w}_2, \mathbf{w}_1 - \mathbf{w}_2 \rangle \\ & \geq \frac{1}{L - \varsigma} \|(\nabla \mathcal{L}(\mathbf{w}_1) - \nabla \mathcal{L}(\mathbf{w}_2)) - (\varsigma \mathbf{w}_1 - \varsigma \mathbf{w}_2)\|_2^2 \end{aligned} \quad (2.18)$$

Referring to Assumption 2.3.1, we obtain

$$\begin{aligned} \langle \nabla \mathcal{L}(\mathbf{w}_1) - \nabla \mathcal{L}(\mathbf{w}_2), \mathbf{w}_1 - \mathbf{w}_2 \rangle & \geq \frac{L\varsigma}{L - \varsigma} \|\mathbf{w}_1 - \mathbf{w}_2\|_2^2 - \frac{2\varsigma}{L - \varsigma} \langle \nabla \mathcal{L}(\mathbf{w}_1) - \nabla \mathcal{L}(\mathbf{w}_2), \mathbf{w}_1 - \mathbf{w}_2 \rangle \\ & \quad + \frac{1}{L - \varsigma} \|\nabla \mathcal{L}(\mathbf{w}_1) - \nabla \mathcal{L}(\mathbf{w}_2)\|_2^2 \\ & \geq -\frac{L\varsigma}{L - \varsigma} \|\mathbf{w}_1 - \mathbf{w}_2\|_2^2 + \frac{1}{L - \varsigma} \|\nabla \mathcal{L}(\mathbf{w}_1) - \nabla \mathcal{L}(\mathbf{w}_2)\|_2^2 \end{aligned} \quad (2.19)$$

Let $\varsigma = -\mu$. This concludes the proof of Lemma 2.3.5.

Proof of Lemma 2.3.6 We have the following equation:

$$\|\mathbf{m}(\mathbf{w}) - \nabla \mathcal{L}(\mathbf{w})\|_2 \leq \|\mathbf{m}(\mathbf{w}) - \mathbf{m}_0(\mathbf{w})\|_2 + \|\mathbf{m}_0(\mathbf{w}) - \nabla \mathcal{L}(\mathbf{w})\|_2 \quad (2.20)$$

From [52] inequality 18, we know $\forall i, n, \exists E > 0$

$$\sup |e_{i,n}| \leq \frac{E}{\sqrt{M}\delta}$$

Where

$$E = \sup \left\{ \frac{37 \sqrt{2} \lambda (M + 4)}{25(M - 1)} \text{median}_i \left\{ |w_{i,n}^t - \hat{B}_n x_{i,n}^t - \hat{A}_n| \right\} \right\}$$

and the dimension of $\mathbf{w}$ is N . Hence the distance between the two aggregation functions satisfies

$$\|\mathbf{m}(\mathbf{w}) - \mathbf{m}_0(\mathbf{w})\|_2 \leq \sqrt{N} \left\| \sum_{i=1}^M \bar{R}_i \left(\hat{B}_i (M - 1) + \frac{2E}{\sqrt{M}\delta} \right) \right\|_2 \quad (2.21)$$

Based on Equation 2.12

$$\frac{s \exp(-cs)}{\sum_{j=0}^s \exp(-cj)} \cdot \frac{Wa}{\eta N + W} \leq \tilde{R}_i^t \quad (2.22)$$

$$\tilde{R}_i^t \leq \frac{s}{\sum_{j=0}^s \exp(-cj)} \cdot \frac{\kappa N + Wa}{\kappa N + W} \quad (2.23)$$

so we have

$$\bar{R}_i = \frac{\tilde{R}_i^t}{\sum_{i=1}^M \tilde{R}_i^t} \leq \frac{1}{\frac{Wa(M-1)(\kappa N + W)}{(\eta N + W)(\kappa N + Wa)} + 1} \quad (2.24)$$

Due to our Aggregation Algorithm

$$\hat{B}_n = \text{median}_i \left(\text{median}_{i \neq j} \frac{w_{j,n} - w_{i,n}}{x_{j,n} - x_{i,n}} \right) \leq \varpi \quad (2.25)$$

Therefore, we have

$$\left\| \sum_{i=1}^M \bar{R}_i \left(\hat{B}_i (M-1) + \frac{2E}{\sqrt{M}\delta} \right) \right\|_2 \leq \frac{M(\varpi(M-1) + \frac{2E}{\sqrt{M}\delta})}{\frac{Wa(M-1)(\kappa N + W)}{(\eta N + W)(\kappa N + Wa)} + 1} = \Delta_1 \quad (2.26)$$

Hence, we proof Lemma 2.3.6.

Considering the assumptions lemmas above, we have

Theorem 2.3.7. *Under Assumptions 2.3.1, 2.3.2, 2.3.3 and 2.3.4, $\exists \epsilon > 0$ that:*

$$\sqrt{\frac{d \log(1 + \hat{Q}MLD_\epsilon)}{M(1-p)}} + C \frac{\mathcal{G}_w}{\sqrt{\hat{Q}}} + p \leq \frac{1}{2} - \epsilon \quad (2.27)$$

After t rounds, Algorithm 2 converges with probability at least $1 - \xi \in \left[1 - \frac{4d}{(1 + \hat{Q}MLv)^d}, 1 \right)$ as

$$\|\mathbf{w}^t - \mathbf{w}^*\|_2 \leq (1 - Lr)^t \|\mathbf{w}^0 - \mathbf{w}^*\|_2 + \frac{\sqrt{N}}{L} \Delta_1 + \frac{1}{L} \Delta_2 \quad (2.28)$$

where

$$\begin{aligned} \Delta_1 &= \frac{M \left(\varpi(M-1) + \frac{2E}{\sqrt{M}\delta} \right)}{\frac{Wa(M-1)(\kappa N + W)}{(\eta N + W)(\kappa N + Wa)} + 1} \\ \Delta_2 &= 2\sqrt{2} \frac{1}{M\hat{Q}} + \sqrt{\frac{2}{\hat{Q}}} D_\epsilon V_w \left(\sqrt{\frac{d \log(1 + \hat{Q}MLv)}{M(1-p)}} + C \frac{\mathcal{G}_w}{\sqrt{\hat{Q}}} + p \right) \\ D_\epsilon &:= \sqrt{2\pi} \exp \left(\frac{1}{2} (\Phi(1 - \epsilon))^2 \right) \end{aligned}$$

with $\Phi(\cdot)$ being the cumulative distribution function of Wald distribution.

Proof of Theorem 2.3.7 We first consider a general problem of robust estimation of a one dimensional random variable. Suppose that there are M clients, and p percentage of them are malicious and own adversarial data. In t iteration, we have:

$$\begin{aligned} \|\mathbf{w}^t - \mathbf{w}^*\|_2 &= \|(\mathbf{w}^{t-1} - r\mathbf{m}(\mathbf{w}^{t-1}) - \mathbf{w}^*)\|_2 \leq \underbrace{\|\mathbf{w}^{t-1} - r\nabla\mathcal{L}(\mathbf{w}^{t-1}) - \mathbf{w}^*\|_2}_A \\ &\quad + \underbrace{r\|\mathbf{m}(\mathbf{w}^{t-1}) - \nabla\mathcal{L}(\mathbf{w}^{t-1})\|_2}_B \end{aligned} \quad (2.29)$$

We bound part A first. We have

$$\|\mathbf{w}^{t-1} - r\nabla\mathcal{L}(\mathbf{w}^{t-1}) - \mathbf{w}^*\|_2^2 = \|\mathbf{w}^{t-1} - \mathbf{w}^*\|_2^2 + r^2 \|\nabla\mathcal{L}(\mathbf{w}^{t-1})\|_2^2 - 2r \langle \nabla\mathcal{L}(\mathbf{w}^{t-1}), \mathbf{w}^{t-1} - \mathbf{w}^* \rangle \quad (2.30)$$

Under the Assumption 2.3.4 and Lemma 2.3.5, we have

$$\langle \nabla\mathcal{L}(\mathbf{w}^{t-1}), \mathbf{w}^{t-1} - \mathbf{w}^* \rangle \geq \frac{L\mu}{L+\mu} \|\mathbf{w}^{t-1} - \mathbf{w}^*\|_2^2 + \frac{1}{L+\mu} \|\nabla\mathcal{L}(\mathbf{w}^{t-1})\|_2^2 \quad (2.31)$$

Then we combine inequalities 2.31 to equation 2.30

$$\|\mathbf{w}^{t-1} - r\nabla\mathcal{L}(\mathbf{w}^{t-1}) - \mathbf{w}^*\|_2^2 \leq (1 - 2r\frac{L\mu}{L+\mu}) \|\mathbf{w}^{t-1} - \mathbf{w}^*\|_2^2 + (r^2 - \frac{2r}{L+\mu}) \|\nabla\mathcal{L}(\mathbf{w}^{t-1})\|_2^2 \quad (2.32)$$

From Assumption 2.3.1, we can derive:

$$\|\nabla\mathcal{L}(\mathbf{w}^{t-1}) - \nabla\mathcal{L}(\mathbf{w}^*)\|_2^2 \leq L^2 \|\mathbf{w}^{t-1} - \mathbf{w}^*\|_2^2 \quad (2.33)$$

Combining inequalities 2.32 and 2.33, we have:

$$\|\mathbf{w}^{t-1} - r\nabla\mathcal{L}(\mathbf{w}^{t-1}) - \mathbf{w}^*\|_2^2 \leq (1 - Lr)^2 \|\mathbf{w}^{t-1} - \mathbf{w}^*\|_2^2 \quad (2.34)$$

Let $r < \frac{1}{L}$, we have

$$\|\mathbf{w}^{t-1} - r\nabla\mathcal{L}(\mathbf{w}^{t-1}) - \mathbf{w}^*\|_2 \leq (1 - Lr) \|\mathbf{w}^{t-1} - \mathbf{w}^*\|_2 \quad (2.35)$$

Then we turn to bound part B . Based on Lemma 2.3.6, we know:

$$\|\mathbf{m}(\mathbf{w}) - \nabla\mathcal{L}(\mathbf{w})\|_2 \leq \|\mathbf{m}_0(\mathbf{w}) - \nabla\mathcal{L}(\mathbf{w})\|_2 + \sqrt{N}\Delta_1 \quad (2.36)$$

Assume Assumption 2.3.1, 2.3.2, 2.3.3 and 2.3.4 holds, and $\exists \epsilon$ fulfills inequality 2.27. Based on Lemma 1 in [53], with the probability $1 - \xi \geq 1 - \frac{4d}{(1+\hat{Q}MLv)^d}$, we have

$$\|\mathbf{m}_0(\mathbf{w}) - \nabla\mathcal{L}(\mathbf{w})\|_2 \leq \sqrt{\frac{2}{\hat{Q}}} D_\epsilon V_w \left(\sqrt{\frac{d \log(1 + \hat{Q}MLv)}{M(1-p)}} + C \frac{\mathcal{G}_w}{\sqrt{\hat{Q}}} + p \right) + 2\sqrt{2} \frac{1}{M\hat{Q}} = \Delta_2 \quad (2.37)$$

where $C = 0.4748$. After obtaining the bound of part A and B, now we have

$$\|\mathbf{w}^t - \mathbf{w}^*\|_2 \leq \underbrace{(1 - Lr) \|\mathbf{w}^{t-1} - \mathbf{w}^*\|_2}_{\text{Bound A}} + \underbrace{r(\sqrt{N}\Delta_1 + \Delta_2)}_{\text{Bound B}} \quad (2.38)$$

Hence, we can prove Theorem 2.3.7 through iterations using the formula of a finite geometric series,

$$\begin{aligned} \|\mathbf{w}^t - \mathbf{w}^*\|_2 &\leq (1 - Lr)^t \|\mathbf{w}^0 - \mathbf{w}^*\|_2 + \frac{1 - (1 - Lr)^t}{Lr} r(\sqrt{N}\Delta_1 + \Delta_2) \\ &\leq (1 - Lr)^t \|\mathbf{w}^0 - \mathbf{w}^*\|_2 + \frac{1}{L} (\sqrt{N}\Delta_1 + \Delta_2) \end{aligned} \quad (2.39)$$

Corollary 2.3.8. *Continuing with Theorem 2.3.7, when t satisfy $t \geq \frac{1}{Lr} \log \left(\frac{L}{\sqrt{N}\Delta_1 + \Delta_2} \|\mathbf{w}^0 - \mathbf{w}^*\|_2 \right)$, $\exists \xi \in \left(0, \frac{4d}{(1 + \hat{Q}MLv)^d} \right]$, we have:*

$$\mathbb{P} \left(\|\mathbf{w}^t - \mathbf{w}^*\|_2 \leq \frac{2\sqrt{N}}{L} \Delta_1 + \frac{2}{L} \Delta_2 \right) \geq 1 - \xi$$

Remark 2.3.9. Due to

$$\Delta_1 := \mathcal{O} \left(\frac{\varpi}{a\kappa WN} + \frac{1}{\kappa N} + \frac{1}{\sqrt{MN}\delta} \right)$$

and

$$\Delta_2 := \mathcal{O} \left(\frac{1}{\hat{Q}} + \frac{p}{\sqrt{\hat{Q}}} + \frac{1}{\sqrt{\hat{Q}M}} \right)$$

Based on Corollary 2.3.8, we achieve an error rate:

$$\mathcal{O} \left(\frac{\varpi}{a\kappa W \sqrt{N}} + \frac{1}{\kappa \sqrt{N}} + \frac{1}{\sqrt{M}\delta} + \frac{1}{\hat{Q}} + \frac{p}{\sqrt{\hat{Q}}} + \frac{1}{\sqrt{\hat{Q}M}} \right)$$

This theoretical error rate has implications for designing our experiments. Specifically, our theoretical analysis informs several practical considerations:

- **Impact of Model Complexity (N):** Increasing the number of model parameters N theoretically reduces the error rate, implying a trade-off between model complexity and performance. As empirically demonstrated in Figure 2.3 and Figure 2.5, increasing the number of parameters indeed enhances robustness, as reflected by more pronounced changes in reputation scores when clients behave maliciously.
- **Impact of Attacker Percentage (p):** Increasing the percentage of attackers p theoretically raises the error rate. Our experiments consistently validate this prediction across various datasets and scenarios. As illustrated empirically in Figures 2.7, 2.9, 2.12, and 2.13, increasing the proportion of attackers consistently leads to a measurable drop in accuracy.

- **Impact of Number of Participating Clients (M):** Increasing the number of participating clients theoretically reduces the error rate. This is empirically supported by results in Figure 2.10, highlighting the improved robustness achieved with greater client participation.
- **Impact of Hyper-parameters (κ, a):** The hyper-parameters κ (reward weight) and a (prior probability), appearing in our theoretical bound, inversely correlate with the error rate. Empirically, systematic experiments (see Figure 2.14) confirm this theoretical prediction, demonstrating that careful tuning of these parameters affects the performance of our reputation system.

Thus, these theoretical insights have directly shaped our experimental design and hyper-parameter selections. Moreover, all empirical observations align consistently with the predictions of our theoretical analysis.

Remark 2.3.10. Derived from the Corollary 2.3.8 and Remark 2.3.9, there is a trade-off problem between convergence speed and error rate according to the level of reward κ and punishment η from the reputation model. This trade-off problem is mainly based on the fact that if the model penalises the bad behaviours of clients heavily, it would decrease their reputation dramatically, so the model would take a longer time to converge. On the other hand, mitigating the punishment to increase the reward, would lead to an increase in the error rate. This provides guidance for adaptively tuning κ and η based on observed client behaviours, in order to maintain an optimal balance between convergence speed and robustness.

2.4. Performance Evaluation

The objectives of our experimental evaluation are the following: (a) evaluate the performance of our aggregation method against other state-of-art robust aggregation methods, (b) benchmark it in three different scenarios, namely, no attack, label flipping attack, and backdoor attack, (c) do so using a text-based real-world dataset of sensitive categories from [84] to which we will henceforth refer to as SURL, and finally (d) show that our experimental result are consistent with our previous theoretical analysis.

2.4.1. Experimental Setup

Datasets and Models

The SURL dataset comes from a crowdsourcing taxonomy in the Curlie project [85], containing six categories of URLs: five sensitive categories (Health, Politics, Religion, Sexual Orientation, Ethnicity) and one for non-sensitive URLs, with a total of 442,190

webpages. The number of URLs in sensitive and non-sensitive categories are equally balanced. Each sample contains content, metadata and a class label of the webpage.

For the SURL text classification task, we train a neural network model with three fully connected layers and a final softmax output layer. This architecture is chosen to ensure consistency with the evaluated methods [52], [53].

Furthermore, in order to fulfil the fundamental setting of a heterogeneous and unbalanced dataset for FL, we sample u_k from a Dirichlet distribution [104] with the concentration parameter $\iota = 0.9$ as in [34], which controls the imbalance level of the dataset, then assigns a $u_{k,i}$ fraction of samples in class k to client i , with the intention of generating non-IID and unbalanced data partitions. As a sanity check, we also tested our reputation scheme on a different classification task involving images CIFAR-10 dataset and got consistent results as those we got for sensitive content.

Threat Model

We consider the following threat model.

Attack capability: In the FL setting, the malicious clients have complete control over their local training data, training process and training hyper-parameters, e.g., the learning rate, iterations and batch size. They can pollute the training data as well as the parameters of the trained model before submitting it to the server but cannot impact the training process of other clients. We follow the common practice in the computer security field of overrating the attacker’s capability rather than underrating it, so we limit our analysis to worst-case scenarios. There, an attacker has perfect knowledge about the learning algorithm, the loss function, the training data and is able to inspect the global model parameters. However, attackers would still have to train with the model published by the server, thus complying with the prescribed training scheme by FL to their local data. Furthermore, the percentage of byzantine clients p is an important factor that determines the level of success for the attack. We assume that the number of attackers is less than the number of honest clients, which is a common setting in similar methods [52], [53] to the ones we evaluate and compare our method with.

Attack strategy: We focus on two common attack strategies for sensitive context classification, namely, (i) label flipping attack [105] and (ii) backdoor attack [34]. Comparing to other attacks, for example model poisoning attack [32], [33], these two data poisoning attacks are more likely to be carried out by real users in the real world via our browser extension described in Section 2.5, since polluting data is easier than manipulating model updates using the browser extension. Note that privacy attacks including membership inference attack [93] and property inference attack [106], are out of the scope of this paper, but form part of our ongoing and future work.

In a label flipping attack, the attacker flips the labels of training samples to a targeted label and trains the model accordingly. In our case, the attacker changes the label of

“Health” to “Non-sensitive”. In a backdoor attack, attackers inject a designed pattern into their local data and train these manipulated data with clean data, in order to develop a local model that learns to recognise such pattern. We realise backdoor attacks inserting the top 10 frequent words with their frequencies for the “Health” category. Therein the backdoor targets are the labels “non-sensitive”. A successful backdoor attack would acquire a global model that predicts the backdoor target label for data along with specific patterns.

For both attacks, instead of a single-shot attack where an attacker only attacks in one round during the training, we enhance the attacker by a repeated attack schedule in which an attacker submits the malicious updates in every round of the training process. Also, we evaluate a looping attack where the attackers spreads out poisoning every 30 epochs for the label flipping attack based on Figure 2.5. It is important to note that even if the attackers have full knowledge of our method, they would still be unable to mount smarter attacks that would try to maximise the damage caused while minimising their reputation drop. This is because the attackers are unaware and cannot compute their reputation score since the latter is computed at the server and requires input from all clients. Moreover, we allow extra training epochs for an attacker, namely, being able to train the local models with 5 more epochs as in [34].

Evaluated Aggregation Methods

We compare the performance of our aggregation method against the existing state-of-the-art robust aggregation methods. Recent surveys [107], [108] underline the growing diversity of aggregation strategies in FL. These works categorize methods ranging from the classical FedAvg to various robust and secure FL aggregations. Table 2.2 provides a comparative summary of the aggregation methods evaluated in this work alongside notable alternatives discussed in recent surveys. Each method is contrasted in terms of robustness to attacks, efficiency, and scalability.

Aggregation Method	Robustness to Attacks	Efficiency	Scalability
FedAvg (baseline) [26]	None — No defense against malicious updates.	High — Simple weighted averaging.	High — Lightweight, scalable to many clients.
Median [53]	High — Set coordinate-wise median of all updates as the global update	Moderate — Requires sorting per parameter.	Moderate — Suitable for tens of clients.
Trimmed-Mean [53]	High — Trims extreme values from the updates.	Moderate — Sorting and filtering overhead.	Moderate — Handles moderate client numbers.
FoolsGold [54]	Moderate — Targets sybil/colluding clients using similarity metrics.	Moderate — Computes pairwise similarities.	Moderate — Scalability challenged by large client counts.
Residual-based Re-weighting [52]	Moderate — Down-weights suspicious updates based on residuals.	Moderate — Adds re-weighting computation overhead.	Moderate — Feasible for tens of clients.
FLTrust [86]	High — Scores updates based on a trusted server-side dataset.	Moderate — Trust scoring adds inference cost.	Moderate — Practical for moderate-sized systems.
Krum [39]	High — Selects the closest update; excludes outliers.	Low — $O(n^2)$ distance computations required.	Low — Poor scalability for large client populations.
Multi-Krum [39]	High — Aggregates multiple close updates for improved robustness.	Low — The extensive analysis of model parameters.	Low — Not suitable for many clients.
Bulyan [109]	High — Robust even against strong collusion.	Low — Iteratively executes aggregation rule.	Low — Impractical for large-scale FL.
Secure Aggregation [110]–[112]	None — Focuses on privacy, not robustness.	Low/Moderate — Cryptographic overhead.	Moderate — Scales, but increases communication cost.
Asynchronous Aggregation [113]–[115]	None/Low — Focuses on latency, not robustness.	High — Incremental updates reduce idle time.	High — Good scalability for heterogeneous networks.

Table 2.2: Comparison of FL aggregation methods in terms of robustness to attacks, efficiency, and scalability

In light of these comparisons, the methods selected for evaluation in this chapter (FedAvg [26], Median [53], Trimmed-mean [53], FoolsGold [54], [55], Residual-based Re-weighting [52], and FLTrust [86]) were prioritized due to their popularity, proven robustness, diversity of approach, and practical applicability. In contrast, some alternatives, such as Krum [39], Multi-Krum [39], and Bulyan [109], while theoretically highly robust,

involve high computational costs that limit their scalability to large federated networks. Secure Aggregation [110]–[112] focuses primarily on privacy rather than robustness, and asynchronous aggregation methods [113]–[115] address efficiency issues without necessarily improving resilience against attacks. Consequently, our choice covers a broad set of aggregation philosophies while remaining suitable for practical FL deployments.

We provide a detailed description of each evaluated method below.

FedAvg is a FL aggregation method that demonstrates impressive empirical performance in non-adversarial settings [26]. Nevertheless, even a single adversarial client could control the global model in FedAvg easily [39]. This method averages local model updates of clients as a global model update weighted by the fraction of the training sample size of each client compared to the total training sample size. We use it as a baseline evaluation to assess the performance of our method.

Median is using coordinate-wise median for aggregation. After receiving the updates $\{\mathbf{w}_i^t\}_{i=1}^M$, in round t the global update is set equal to the coordinate-wise median of the updates, $\mathbf{w}^t := \text{median}_i \{\mathbf{w}_i^t\}$ with its n -th coordinate $\mathbf{w}_n^t := \text{median}_i \{\mathbf{w}_{i,n}^t\}$ for each $n \in N$ where the median is the 1-dimensional median.

Trimmed-mean is another coordinate-wise mean aggregation technique that requires prior knowledge of the attacker fraction β , which should be less than half of the number of model parameters. For each model parameter, the server eliminates the highest and lowest β values from the updates before computing the aggregated mean with the remaining values.

FoolsGold presents a strong defence against attacks in FL, based on a similarity metric. Such an approach identifies attackers based on the similarity of the client updates and decreases the aggregate weights of participating parties that provide indistinguishable gradient updates frequently, while keeping the weights of parties that offer distinct gradient updates. It is an effective defence for sybil attacks, but it requires more iterations to converge to an acceptable accuracy.

Residual-based Re-weighting weights each local model by accumulating the outcome of its residual-based parameter confidence by multiplying the standard deviation of the parameter based on the robust regression through all the parameters of this local model. In our reputation-based aggregation method, we implement the same re-weighting scheme IRLS [99] as residual-based aggregation, but choose the collection of reputation as the weights of clients' local models.

FLTrust establishes trust in the system by bootstrapping it via the central server, instead of depending entirely on updates from clients, like the other methods do. The server obtains an initial server model trained on clean root data. Then, depending on the cosine similarity of the server model and each local model, it assigns a trust score to each client in each iteration.

Performance Metrics

We use the average test accuracy (Avg-ACC) of the global model to evaluate the result of the aggregation defence for the poisoning attack in which attackers aim to mislead the global model during the testing phase. The accuracy is the percentage of testing examples with the correct predictions by the global model in the whole testing dataset, which is defined as:

$$\text{Avg-ACC} = \frac{\# \text{ correct predictions}}{\# \text{ testing samples}}$$

In addition, there is existence of targeted attacks that aim to attack a specific label while keeping the accuracy of classification on other labels unaltered. Therefore, instead of Avg-ACC, we choose the attack success rate (ASR) to measure how many of the samples that are attacked, are classified as the target label chosen by a malicious client, namely:

$$\text{ASR} = \frac{\# \text{ successfully attacked samples}}{\# \text{ attacked samples}}$$

A robust federated aggregation method would obtain higher Avg-ACC as well as a lower ASR under poisoning attacks. An ideal aggregation method can achieve 100% Avg-ACC and has the ASR as low as the fraction of attacked samples from the target label.

Evaluation Setup

For the malicious attack, we assume that 30% of the clients are malicious as in [39], which is also a common byzantine consensus threshold for resistance to failures in a typical distributed system [116]. For the server-side setting, in order to evaluate the reliability of the local model updates sent by the client to the server, we assume that the server has the ability to look into and verify the critical properties of the updates from the clients before aggregating. Also, we only consider FL to be executed in a synchronous manner, as most existing FL defences require [32], [39], [53], [102]. For all the above aggregation methods under attack, we perform 100 iterations using the SURL dataset with a batch size of 64 and 10 clients. Furthermore, we evaluate our method for increasing numbers of clients. These settings are inline with existing state-of-the-art methods for security in FL [34], [52], [53].

Our simulation experiments are implemented with the Pytorch framework [117] on the cloud computing platform Google Colaboratory Pro (Colab Pro) with access to Nvidia K80s, T4s, P4s and P100s with 25 GB of Random Access Memory. Table 2.3 shows the default setting in our experiments. For the training setting, the local models are trained independently and sequentially in each round of training before being aggregated into the global model. Also, we presume that clients do not leave the system.

Table 2.3: Default Experimental Settings

Explanation	Notation	Default Setting
Prior probability	a	0.5
Non-information prior weight	W	2
Weight for positive observation	κ	0.3
Time decay parameter	c	0.5
Window length	s	10
Confidence threshold	δ	0.1
Value range	ϖ	2
Objective function	$\mathcal{L}(\cdot)$	Negative Log-likelihood Loss
Learning rate	r	0.01
Batch size per client		64
The number of local iterations		10
The number of total iterations		100

2.4.2. Convergence and Accuracy

In Figure 2.6 (left), we analyse the performance of our method in the no attack scenario and compare the convergence and accuracy of our method with others during training. We show the training loss (left axis) and average accuracy (right axis) during 100 training iterations for 7 methods. Our aggregation starts with the lowest training loss and maintains

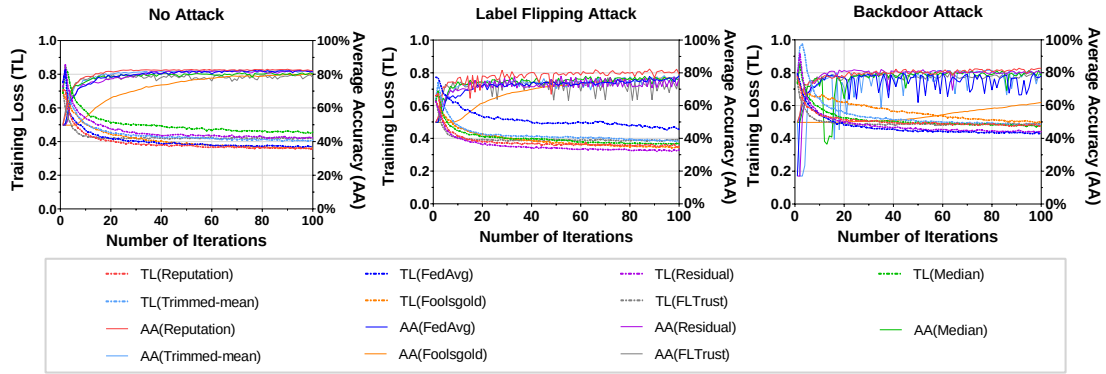


Figure 2.6: Training Loss (TL) and Average Accuracy (AA) for 100 epochs of Reputation-based, FedAvg, Residual-based, Median, Trimmed-mean, FoolsGold and FLTrust methods in SURL Dataset under no attack (left) scenario, under label flipping attack (middle) and backdoor attack (right) scenarios with 30% malicious clients.

it throughout the training process. It only takes 24 iterations to achieve 82% accuracy and then converge to 82.13%, which represents a $2.7\times$ faster converge rate than FedAvg. In comparison, Residual-based and Trimmed-mean have almost identical training loss and take 52 and 47 iterations to reach 82% accuracy and practically converge to 81.79% and 80.76% respectively, which is $2.2\times$ and $2\times$ slower than our reputation method. Median reaches 81% at 83 rounds and after that converges to 79.94%, which amounts to a $3.6\times$ slower converge rate than our method. Especially, FoolsGold and FLTrust are slow to converge and do not converge within 100 iterations, so our convergence rate is at least $4.2\times$

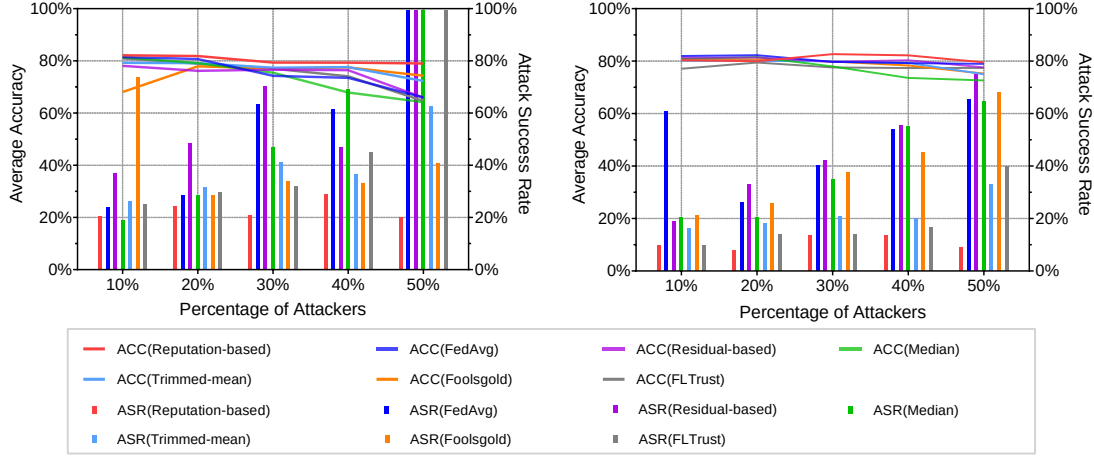


Figure 2.7: Average accuracy (ACC) and attack success rate (ASR) for varying percentage of attackers from 10% to 50% under label flipping (left) and backdoor (right) attack for Reputation-based, FedAvg, Residual-based, Median, Trimmed-mean, FoolsGold and FLTrust methods in SURL Dataset.

better than FoolsGold and FLTrust. This demonstrates that our reputation model benefits from convergence speed and accuracy performance. This is because our reputation scheme assigns higher weight to more reliable clients when there is no ongoing attack, which generates more consistent updates thereby accelerating the convergence.

2.4.3. Resilience to Attacks

We begin by analysing the performance with a static percentage (30%) of attackers, and then move on to the performance with a varying percentage of attackers under label flipping and backdoor attacks.

Label Flipping Attack

Static percentage of attackers: Figure 2.6 (middle) shows the convergence of mentioned methods under label flipping attack. Our method converges $1.8\times$ to $2\times$ faster than all competing state-of-the-art methods under attack, enlarging its performance benefits compared to the no attack scenario. In addition, our method outperforms competing methods by at least 1.4% in terms of accuracy.

Varying the percentage of attackers: Here we analyse the impact on our aggregation method as the proportion of attackers increases. Figure 2.7 (left) shows the change of performance metrics for varying percentage of attackers for seven evaluated methods. When the percentage of attackers p ranges from 10% to 50%, our method is resistant against label flipping attacks with a small loss in accuracy and a consistent attack success rate of all the methods. As p approaches 50%, FedAvg, Residual-based, Median and FLTrust

defences become ineffective in mitigating the attack, and correspondingly their Avg-ACC decreases linearly. Moreover, under label flipping attack during the whole process, our reputation-based method has the highest accuracy outperforming other methods by 1% to 23.1%. At the same time it has the lowest ASR. The average ASR of other methods are at least 82.8% higher than ours.

Backdoor Attack

Static percentage of attackers: Figure 2.6 (right) shows the convergence of the mentioned methods under backdoor attack. Same as in no attack and label flipping attack scenario, our method converges $1.6\times$ to $2.4\times$ faster than all competing state-of-the-art methods. In addition, our method outperforms competing methods by 3.5% to 33.6% in terms of classification accuracy.

Varied percentage of attackers: we examine the scenario in which the percentage of attackers increases. Figure 2.7 (right) shows the performance for the evaluated methods under backdoor attack when varying the percentage of attackers p from 10% to 50%. Figure 2.7 (right) demonstrates that under backdoor attack, our reputation-based method has a consistent accuracy throughout the process with the lowest attack success rate, whereas the average ASR of other methods is at least 72.3% higher than ours. As p changes, the ASR of the Residual-based, Median, and Foolsgold methods increase linearly. Although FLTrust has a stable ASR, it increases by a factor of 1.39 when p reaches 50%.

Analysis of Attacks

First, we evaluate a varying compromise rate for the label flipping and backdoor attacks using our reputation-based method. For the label-flipping attack, we vary the percentage of the flipped label poisoned by attackers from 10% to 90%. Also, for the backdoor attack, we vary the number of top frequent words inserted as the trigger pattern, from 5 to 25. The remaining settings are the same as in previous experiments. Figure 2.8 presents the ACC and ASR results as the compromise rate is varied for both attacks. As shown in Figure 2.8 (left), increasing the percentage of poisoned samples leads to a decrease in model accuracy and a slight increase in ASR. Figure 2.8 (right) demonstrates that increasing the number of frequent words from 5 to 20 has little effect on ASR. However, when the number of frequent words exceeds 25, the attack becomes less stealthy and easier to detect, resulting in a lower ASR.

Next, we conduct an analysis of different attack timing strategies. Specifically, instead of repeating the attack at every epoch, the attacker distributes the poisoning effort across 30 epochs in our study of the looping attack. This setup simulates a more subtle and prolonged adversarial behaviour compared to the repeated attack scenario. The performance of the looping attack is presented in Figure 2.9. As anticipated, the looping attack proves to be less effective than the repeated attack evaluated previously. All methods

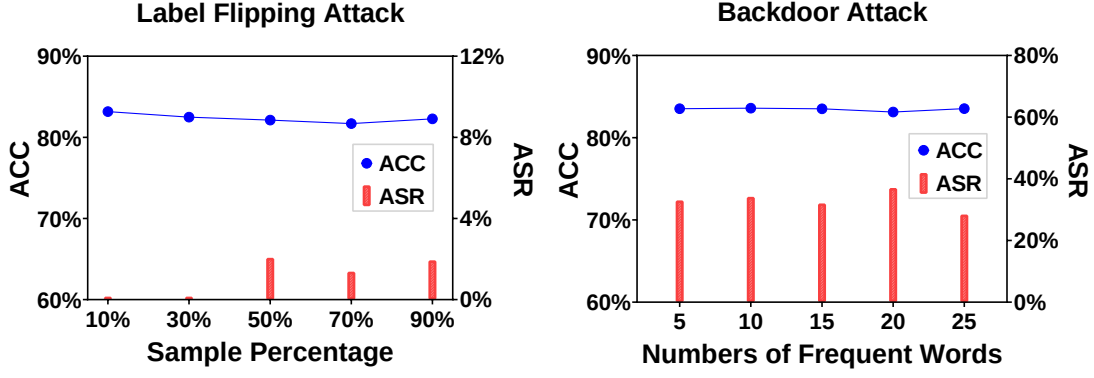


Figure 2.8: ACC and ASR as we vary the percentage of flipped labels from 10% to 90% (left) for label flipping attack, and the number of the frequent words as the trigger pattern from 5 to 25 for backdoor attack (right).

under consideration are able to defend against it effectively, achieving consistently low Attack Success Rates (ASR). Notably, our method maintains the highest classification accuracy among all defenses, further demonstrating its robustness and resilience against diverse and prolonged poisoning strategies.

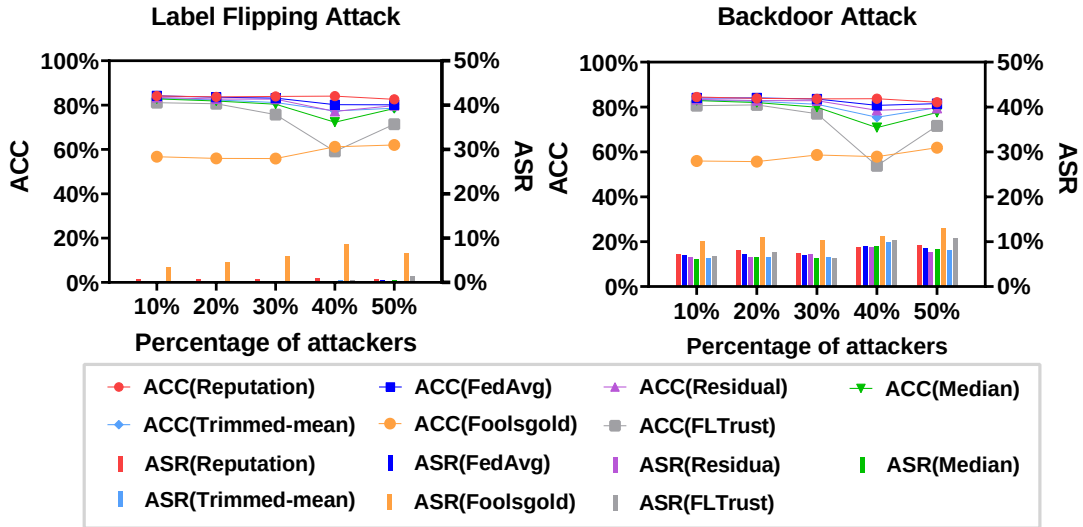


Figure 2.9: Average accuracy (ACC) and attack success rate (ASR) for varying percentage of attackers from 10% to 50% under label flipping (left) and backdoor (right) attack with a looping attack in which an attacker attacks every 30 epochs.

Evaluation Results

In the no-attack scenario, we observe that our method

- converges $2\times$ to $4.2\times$ faster than all competing state-of-the-art methods.

- achieves classification accuracy that is comparable to or surpasses that of competing methods.

The above validates that our reputation scheme is helpful even in the no attack scenario. This is due to the fact that in our algorithm we give higher weights to the clients with high-quality updates, as illustrated in Figure 2.15, causing the model to converge rapidly and retain consistent accuracy. In addition, even under the two different attacks, our method:

- converges $1.6\times$ to $2.4\times$ faster than all competing state-of-the-art methods.
- provides the same or better accuracy than competing methods.
- yields the lowest ASR compared to all other methods, with the average ASR of them being at least 72.3% higher than ours.

2.4.4. Evaluation with an Increased Number of Participating Clients

We evaluate the performance of our method with respect to the number of participating clients. Under a 30% compromise rate, we scale the number of clients from 10 to 200. As shown in Figure 2.10, our method maintains stable performance, with both average accuracy (ACC) and attack success rate (ASR) remaining consistent as the number of clients increases. In addition, we analyse the performance of the aforementioned methods

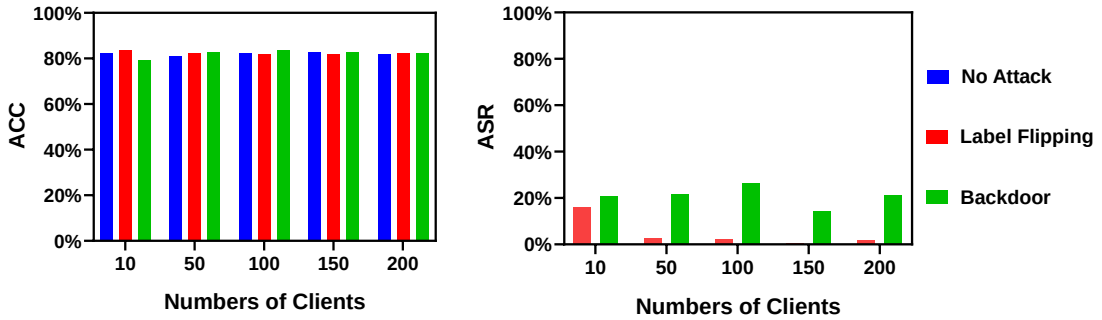


Figure 2.10: Average accuracy (ACC) and attack success rate (ASR) for varying the number of clients from 10 to 200 under label flipping and backdoor attack with 30% malicious clients.

for 100 participating clients. The results are presented in Figure 2.11 and Figure 2.12 that correspond to the average accuracy (ACC) and the attack success rate (ASR), respectively. In the no attack scenario, we observe that our method converges between $1.7\times$ to $7.1\times$ faster than all competing state-of-the-art methods with at least as good performance (or outperforms) compared with competing methods in terms of classification accuracy, see Figure 2.11 (left). In addition, even under the two different attacks, our method: (i) converges between $1.6\times$ to $3.6\times$ faster than all competing state-of-the-art methods, (ii)

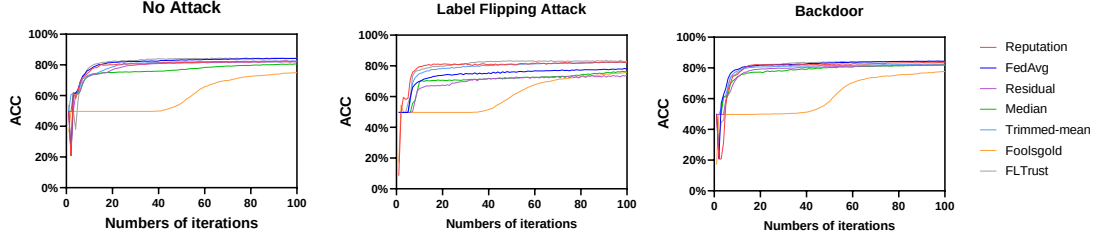


Figure 2.11: Average accuracy (ACC) with no attack (left) and 30% percentage of attackers under label flipping (middle) and backdoor (right) attack for Reputation, FedAvg, Residual-based, Median, Trimmed-mean, Foolsgold, FLTrust in SURL dataset with 100 clients.

provides the same or better accuracy than competing methods, and (iii) yields the lowest ASR compared to all other methods.

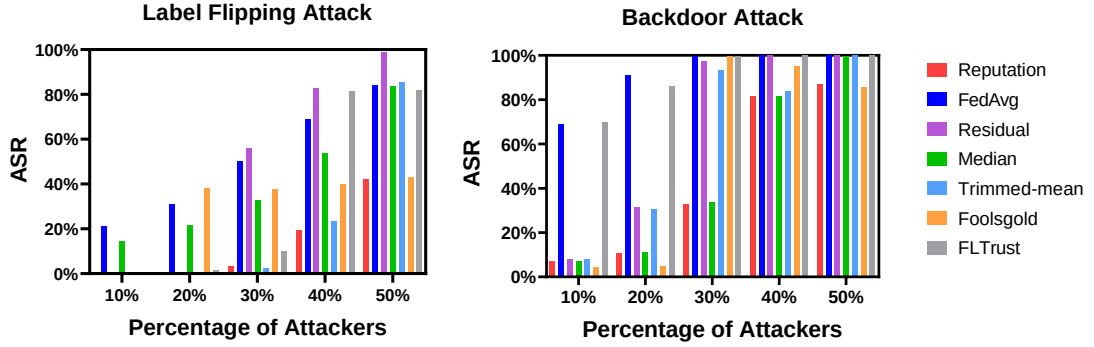


Figure 2.12: Attack success rate (ASR) for varying percentage of attackers from 10% to 50% under label flipping (left) and backdoor (right) attack for Reputation, FedAvg, Residual-based, Median, Trimmed-mean, Foolsgold, FLTrust in SURL dataset with 100 clients.

2.4.5. Supplementary Dataset For Validation

CIFAR-10 is a supplementary dataset assessing the robustness of our reputation scheme in image datasets to poisoning attacks. The CIFAR-10 dataset is a 32×32 colour image dataset that includes ten classes with a total number of 50 thousand images for training and 10 thousand images for testing. Here we use ResNet-18 [118] model pertaining to ImageNet [119] with 20 iterations. In the CIFAR-10 dataset, attackers switch the label of “cat” images to “dog”. In Figure 2.13, the average accuracy (ACC, left y-axis) is shown with lines, while the attack success rate (ASR, right y-axis) is represented by bars. It shows that under label flipping and backdoor attack during the whole process, our reputation-based method has the highest accuracy outperforming other methods and

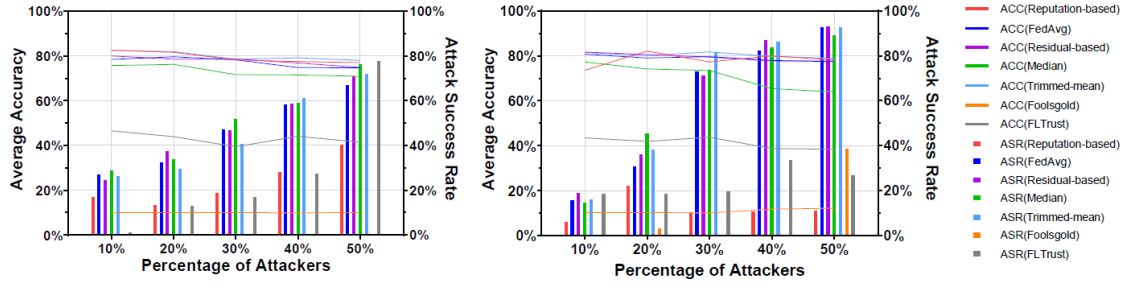


Figure 2.13: Average accuracy (ACC) and attack success rate (ASR) for varying percentage of attackers from 10% to 50% under label flipping (left) and backdoor (right) attack for Reputation, FedAvg, Residual-based, Median, Trimmed-Median, FoolsGold and FLTrusts in CIFAR-10 dataset.

the lowest ASR, excluding FoolsGold, yielding a result that is consistent with the one in SURL.

2.4.6. Stability of Hyper-parameters

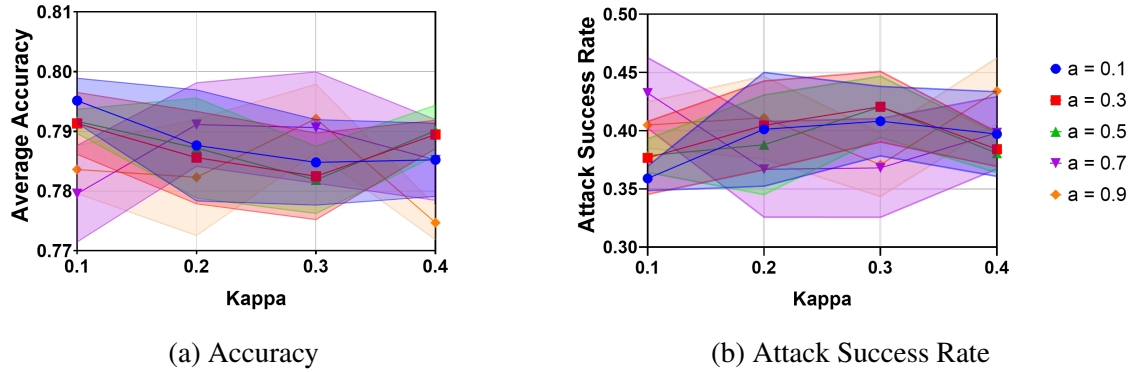


Figure 2.14: Average accuracy and attack success rate as we vary rewarding weight κ and prior probability a .

We employ four hyper-parameters in our reputation model: rewarding weight κ , prior probability a , time decay parameter c and window length s . As shown in Remark 2.3.9, c and s do not affect the performance of our model; we only consider hyper-parameters κ and a , where κ controls the reward weight to positive observations and a controls the fraction of uncertainty converted to belief. To demonstrate the impact of these two hyper-parameters of our reputation model, we grid search κ in $[0.1, 0.2, 0.3, 0.4]$ and a in $[0.1, 0.3, 0.5, 0.7, 0.9]$. The setup is the same as on SURL dataset under the label flipping attack. The ultimate accuracy of stability of reputation-based aggregation are shown in Figure 2.14. Note that these results are tested for the label flipping attack, and they hold according to theory also for the backdoor.

The result in Figure 2.14 demonstrates that our approach is very stable and efficient in terms of hyper-parameter selection, and it achieves a high degree of precision. Furthermore, the result is compatible with the theoretical analysis in Section 2.3.3.

2.4.7. Comparison Against the Residual-based Method

To demonstrate how our method improves the residual-base method by assigning the aggregation weights based on reputation, we consider a scenario with 10 clients in the FL system, 8 of which are malicious. The training lasts 10 communication rounds during which attackers carry out the backdoor attack. The remaining settings are the same as the default. Results are shown in Figure 2.15, in which the first two clients are benign, and

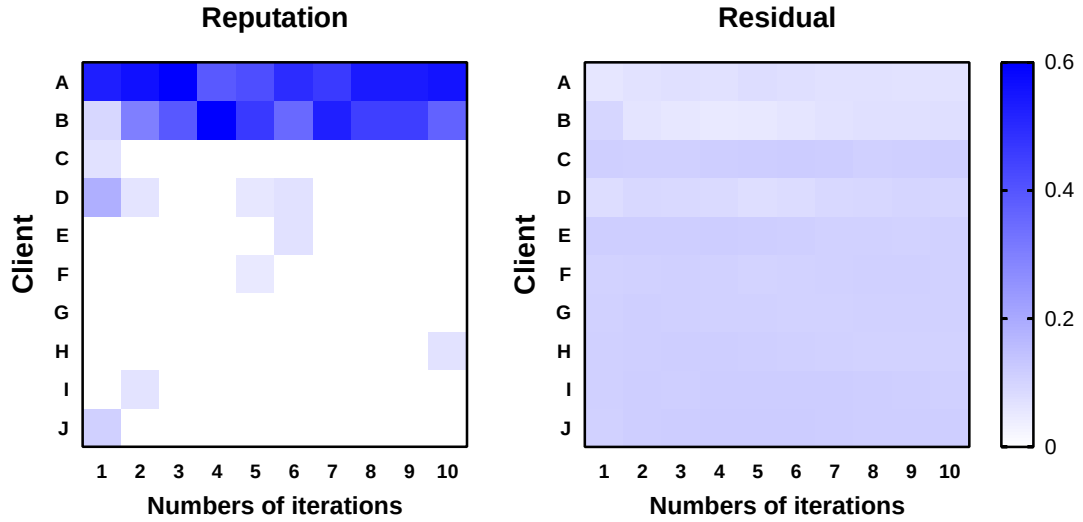


Figure 2.15: The aggregation weights of clients from our reputation-based (left) and residual-based method (right) for 10 communication rounds under label flipping attack with 80% attackers.

the rest are malicious. We observe that for our reputation method the aggregation weights of malicious clients, which are their reputations, are rectified to 0 since the second round, demonstrating that our method is successful in eradicating their influence. On the other hand, the aggregate weights of malicious clients in residual-based methods, which are calculated by multiplying the parameter confidence by its standard deviation, are nearly similar and non-zero. This is because repeated median regression seldom yields 0 for the parameter confidence, which causes practically non-zero weights to be assigned to malicious clients by residual-based methods. To address this issue, the reputation model uses positive and negative observations that introduce rewards and punishments to assign divergent weights to clients. As a result, benign clients are given higher weights whereas malicious clients are eliminated from the aggregation.

2.5. The *EITR* system

In this section, we provide a high level description of our *EITR* [120] system (standing for “Elephant In the Room” of privacy). We then present some preliminary results with real users demonstrating the ability of the system to quickly learn how to classify yet unseen sensitive content, in our case COVID-19 URLs pertaining to the category Health, even in view of inaccurate user input. The system is currently being used as a research prototype to evaluate the robustness of our algorithm in a simple real-world setting. A full in depth description of the system and its performance with more users and more intricate settings, including adoption, incentives, and HCI issues, over a longer time period is the topic of our ongoing efforts and will be covered by our future work.

2.5.1. System Architecture and Implementation

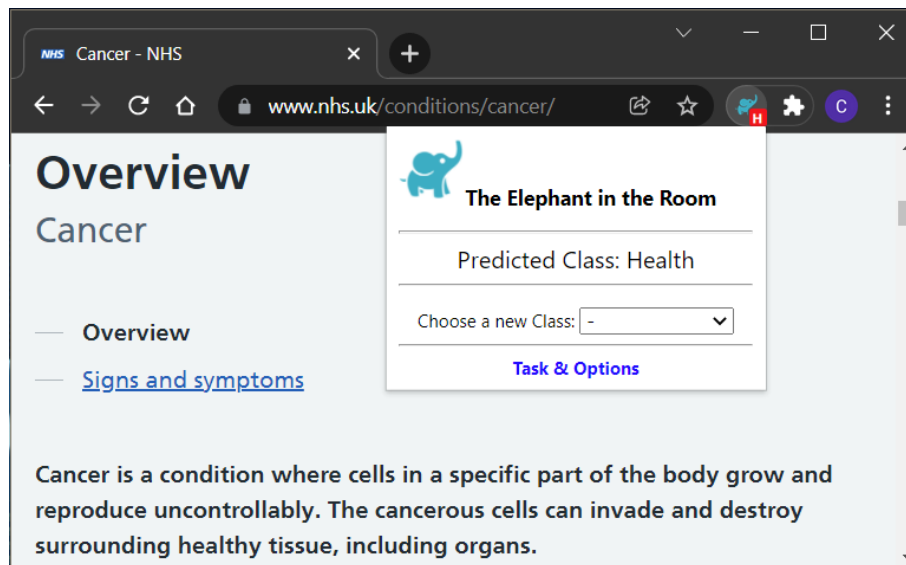


Figure 2.16: *EITR* extension in action. The letter “H” inside the red frame at the bottom right of the extension’s icon indicates that a health-related page has been detected.

The *EITR* system is based on the client-server model. The back-end server is responsible to distribute the initial classification model and the consequent updated model(s) to the clients and receive new annotations from the different clients of the system. The client is in the form of a web browser extension that is responsible to fetch and load the most recent global classification model to the users’ browser from the back-end server. The loaded model can then be used to label website in real time into the 5 different sensitive topics as defined by GDPR, i.e., Religion, Health, Politics, Ethnicity and Sexual Orientation. Next, we provide more details for each part of the system.

Back-end server: The back-end server is written in JavaScript using the node.js Express [121] framework. To build the initial classification model we use the dataset provided by Matic et al. [84], and the TensorFlow [122] and Keras [123] machine learning

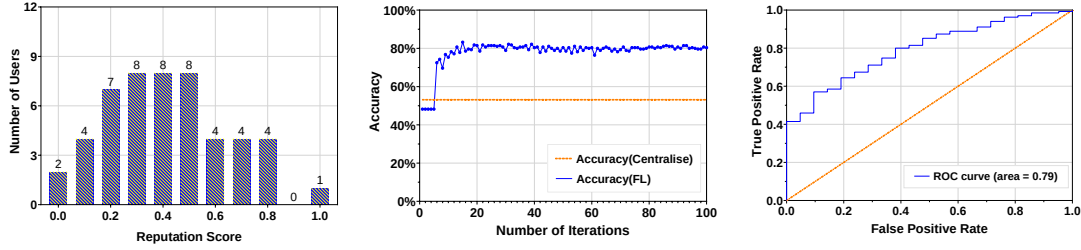


Figure 2.17: Results of real-user experiment for COVID-19 related URLs with 50 users over 100 iterations. (left): the reputation score of the real users at the end of experiment, (middle): the accuracy of the centralised classifier and the global model of our reputation-based FL approach for COVID-19 URLs, and (right): the ROC curve of real-user experiment with 0.79 area under the ROC curve (AUC).

libraries. The final trained model is then exported using the TensorFlow.js [124] library in order to be able to distribute it to the system’s clients (browser extension). The back-end server also includes additional functionalities such as the creation and distribution of users’ tasks, i.e., a short list of URLs that the users need to visit and annotate, and an entry point that collects the resulting users annotations during the execution of the task.

Web browser extension: Currently the browser extension only supports the Google Chrome browser and is implemented in JavaScript using the Google Chrome Extension APIs [125]. To handle the classification model the extension utilises the TensorFlow.js [124] library to load, use, and update the model. The main functionality of the extension is to classify the visited website in real time and provide information to the user related to the predicted class as depicted in Figure 2.16. The website classification is based on the metadata (included in the website <head> HTML tag) and the visible text of the website. The extension also allows users to provide their input related to their agreement or disagreement with the predicted class using a drop down list as depicted in Figure 2.16 with the label “Choose a new Class”.

2.5.2. Real Users Experimental Setup

The goal of the real-user experiment is to evaluate our federated reputation-based method on real user activity (instead of systematic tests), and demonstrate that even with real users with different comprehensions of the definition of sensitive information, our method can learn new content fast and achieve higher accuracy than centralised classifiers, which is compatible with our simulation experiment.

For the setup, the participant is directed to visit the experiment website that provides the necessary information and instructions on the goal of our study, the definition of sensitive information provided by the current GDPR and how to participate in our study. In order to have access to the browser extension and the installation instructions the user must in advance give explicit consent and accept the data privacy policy. Upon success-

ful installation of the extension, new users are asked to provide a valid email address (to contact them for the reward) and then receive their task, a list of 20 URLs, that they need to visit and provide their labels in order to successfully complete their task. The list of 20 URLs is sampled by the Dirichlet distribution with $\iota = 0.9$ for each participant from a database, which includes 300 URLs with sensitive and non-sensitive content related to COVID-19.

Ethical Considerations: We have ensured compliance with the GDPR pertaining to collecting, handling, and storing data generated by real users. To that end, we have acquired all the proper approvals from our institutions. Furthermore, the participants are directed to visit a pre-selected set of URLs selected by us to avoid collecting the actual visiting patterns of our users. In addition, the user input is only collected *if and only if* the user explicitly provide input to the drop-down list labelled “*Choose a new Class*” to avoid collecting the visiting patterns of the user accidentally while they are executing their tasks. Finally, we only use the users’ email address to contact them for the reward. The mapping between the user input and their email address is based on a random identifier that is generated during the installation time of the extension.

2.5.3. Validation with Real Users

Data collection: We had 50 users participating in our experiment. In order to evaluate our reputation-based FL method using real-user data, we define a methodology to label ground truth on COVID-19 related sites.

Ground truth methodology: To set the ground truth for COVID-19 sites related to our sensitive or non-sensitive labels, we create a database of 100 websites, which we collect by searching on Google with the query “sensitive websites about COVID-19” and choose the top 100 sites returned from the query. Then, four experts in the privacy field, independently annotate them based on their professional expertise in order to achieve an agreement on whether each of those sites included sensitive or non-sensitive content.

Ground truth annotation: In order to evaluate the annotation of the 100 websites from human experts, we calculate the inter-rater agreement among them using Fleiss Kappa [126]. We obtain 0.56 of Fleiss Kappa, which is an acceptable agreement because the values of Fleiss Kappa. above 0.5 are regarded as good. Furthermore, given that COVID-19 is a controversial issue, it is difficult for humans to agree on what constitutes sensitive content relating to it. Even though, we still attain a valid ground truth of 85 items belonging to the health sensitive category with agreement ratings of at least 0.5. Note that we also classify the above 100 websites using the centralised classifier proposed in [84] and get only 53.13% accuracy.

Result with real users: Figure 2.17 shows the results of accuracy and reputation score with 50 real users in the experiment. Figure 2.17 (left) shows that the majority of users have reputation scores falling in the intermediate range, with some having a very high reputation and a few having a very low reputation. This indicates the divergence of the

user’s interpretation of the sensitive information as we expect. In Figure 2.17 (middle) we compare the accuracy of the centralised classifier and the global model of our reputation-based FL approach for COVID-19 URLs. Despite the diversity of reputation scores of real users, our method converges as rapidly as in simulation and achieves an average accuracy of 80.36%, thereby verifying the quick convergence and high accuracy results presented in the previous sections. Figure 2.17 (right) shows that the ROC curve in real-user experiment yielded 0.79 AUC. Our result is acceptable in this scenario because most existing FL techniques are designed to minimise the conventional cost function and are not optimal for optimising more appropriate metrics for imbalanced data, such as AUC [127].

As we observe, with real users holding our method achieve a good performance. This means that, as new sensitive content appears and/or is defined by GDPR or new upcoming legislation, we will be able to continue training our FL model for this type of task with quick convergence and good accuracy. The empirical results in Figure 2.17 (middle) also shows that there is a quick convergence to the accuracy’s stable value within a small number of iterations (around 30), in line with the theoretical results in Section 2.3.3.

2.6. Conclusions

In this chapter we have shown how to use federated learning to implement a robust to poisoning attacks distributed classifier for sensitive web content. Having demonstrated the benefits of our approach in terms of convergence rate and accuracy against state-of-the-art approaches, we implemented and validated it with real users using our *EITR* browser extension. Collectively, our performance evaluation has showed that our reputation-based approach to thwarting poisoning attacks consistently converges faster than the state-of-the-art while maintaining or improving the classification accuracy.

We are currently working towards disseminating *EITR* to a larger user-base and using it to classify additional sensitive and non-sensitive types of content. This includes but it is not limited to categories defined by the users themselves for different purposes, not necessarily related to sensitive content, as well as evaluating additional attacks and threat models under our subjective-logic reputation scheme for FL. In turn, our approach can support other FL models going beyond sensitive content classification in future work.

3. SECURE FEDERATED LEARNING AGGREGATION LEVERAGING QUADRATIC VOTING

A crucial step of Federated Learning as demonstrated in the previous chapter, that of aggregating local models to produce the global one, shares many similarities with public decision-making, and elections in particular. In that context, a major weakness of FL, namely its vulnerability to poisoning attacks, can be interpreted as a consequence of the *one person one vote* (henceforth *1p1v*) principle that underpins most contemporary aggregation rules.

In this chapter, we introduce FEDQV, a novel aggregation algorithm built upon the *quadratic voting* scheme, recently proposed as a better alternative to *1p1v*-based elections. Our theoretical analysis establishes that FEDQV is a truthful mechanism in which bidding according to one’s true valuation is a dominant strategy that achieves a convergence rate matching that of state-of-the-art methods. Furthermore, our empirical analysis using multiple real-world datasets validates the superior performance of FEDQV against poisoning attacks. It also shows that combining FEDQV with unequal voting “budgets” according to a reputation score increases its performance benefits even further. Finally, we show that FEDQV can be easily combined with Byzantine-robust privacy-preserving mechanisms to enhance its robustness against both poisoning and privacy attacks.

3.1. Introduction

Parties participating in FL collaborate to train a global model by submitting their local models to a server, which applies a specific aggregation rule to generate a new version of the global model to be sent back to parties. The process of agreeing on a common global model in FL shares many similarities with public decision-making and elections in particular. Indeed, the weights of local model updates from each party can be seen as votes of preference that affect the global model resulting from an aggregation rule applied at the centralised server of an FL group.

FEDAVG [26] has been the “de facto” aggregation rule used in FL tasks such as Google’s emoji and next-word prediction for mobile device keyboards [21], [50]. In FEDAVG the global model is produced from a simple weighted averaging of local updates with weights that represent the amount of data that each party has used for its training.

The problem. Recent work [39] has shown that FEDAVG is vulnerable to poisoning attacks, as even a single attacker can degrade the global model by sharing faulty local updates of sufficiently large weight. Such attacks become possible because FEDAVG treats all local data points equally. In essence, the aggregation rule, when seen at the granularity of individual training data, resembles the *one person one vote* (*1p1v*) election rule of modern

democratic elections. In this context, the server distributes votes (weights) to a party in accordance with the amount of its training data, which may be regarded as its population. This, however, may confer an unjust advantage to malicious parties who may have, or falsely claim to have, large training datasets.

Our approach. To address this issue, we propose a robust aggregation rule inspired by elections based on *Quadratic Voting* [128] (henceforth QV). In QV, each party is given a voting budget that can be spent on different rounds of voting for proposals. Within a particular vote, an individual has to decide the number of "credit voices" to commit, whose square root is what impacts the corresponding outcome of the vote, hence the name quadratic voting. The number of "credit voices" is determined by the individual's voting preference for the proposal. QV has been proposed as a means to break out from the tyranny-of-the-majority vs. subsidising-the-minority dilemma of election systems [129]. Its formal analysis [130] under a game theoretic price-taking model, has shown that QV outperforms *lplv* in terms of efficiency and robustness. Importantly, it has the unique capacity to deter collusion attacks by effectively taxing extreme behaviours.

Our contributions. In this chapter, we propose **FEDQV**, a novel FL aggregation scheme that draws inspiration from QV. Our objective is to mitigate the ability of malicious peers to impose disproportional damage on the global model – a vulnerability inherent in **FEDAVG** that applies the *lplv* principle at the granularity of individual votes. By addressing this issue, **FEDQV** serves as a superior alternative to **FEDAVG**, offering increased robustness against poisoning attacks while retaining compatibility with privacy-guaranteed mechanisms to defend against privacy attacks, as will be demonstrated later.

Our primary contribution involves integrating QV principles into FL, augmented by the implementation of multiple defensive layers, to establish the truthful mechanism, **FEDQV**. We begin with the incorporation of quadratic computation from QV into the FL setting. This incorporation restricts the ability of malicious peers to inflict high damages by taxing their aggregation weights more than linear. However, this direct incorporation alone falls short of capturing each party's voting preference, a crucial aspect of QV. To capture this preference, we require parties to submit the similarity of their local model with the previous round's global model, which serves as a measure of the parties' preference. This modification enables the integration of QV principles into the FL system. Then in response to potential malicious attempts from peers, we introduce **FEDQV**, a truthfulness mechanism alongside our application of QV to FL. This mechanism employs a masked voting rule on the server side to conceal the voting calculation process from parties. Additionally, it incorporates outlier detection and a limited voting budget, information that is exclusive to the server. These defence measures collectively act as a deterrent against potential poisoning attacks and untruthful strategic behaviour, ensuring the overall robustness of the **FEDQV** system in the face of adversarial threats.

To further enhance resilience against poisoning attacks, we extend **FEDQV** to use adaptive voting budgets. In election-related applications, QV allocates equal budgets to all

voters, reflecting the democratic principle of equal rights. However, in our adaptation of QV for FL, it makes sense to allocate more votes to benign peers and limit the influence of malicious ones by assigning them smaller voting budgets. We achieve this by employing unequal budgets, which are tied to a reputation score for each peer, as discussed in Section 3.5.7.

Our next contribution is the implementation of various state-of-the-art Byzantine fault tolerance techniques on top of FEDQV and demonstrating that FEDQV serves as a complementary defence that further boosts the robustness of existing defence techniques. Put differently, FEDQV is an enhancer of existing byzantine fault tolerance techniques, *not a competitor* – implementing these defences atop FEDQV consistently yields superior results compared to implementing them atop FEDAVG.

To also defend against privacy attacks, we design FEDQV such that it can be easily combined with existing privacy-guaranteeing mechanisms to thwart *inference and reconstruction attacks* [37], [70], [106]. Specifically, we showcase the compatibility of FEDQV by implementing SECAGG[110] on top of the FEDQV framework. We then demonstrate experimentally the effectiveness of SECAGG within the FEDQV framework, focusing on its ability to withstand powerful privacy attacks.

Our final contribution comprises an extensive theoretical analysis in order to: 1) establish convergence guarantees, and 2) prove the truthfulness of our method.

Our findings. Using a combined theoretical and experimental evaluation, we show that:

- FEDQV is a truthful mechanism and is theoretically and empirically compatible with FEDAVG in terms of accuracy and convergence under attack and no-attack scenarios.
- FEDQV consistently outperforms FEDAVG under various state-of-the-art poisoning attacks, especially for local model poisoning attacks improving the robustness to such attacks by a factor of at least 4×.
- The combination of FEDQV with a reputation model to assign unequal credit voice budgets to parties according to their respective reputations, improves robustness against poisoning attacks by at least 26% compared to the baseline FEDQV that uses equal budgets.
- We show that integrating FEDQV with established Byzantine-robust FL defences, including Multi-Krum [39], Trimmed-Mean [53], and Reputation [1], results in substantial enhancements in accuracy and reductions in the attack success rate (ASR) under state-of-the-art attacks when compared to the original defence methods. Specifically, integrating FEDQV into Multi-Krum results in a twofold improvement in accuracy under two untargeted attack scenarios, along with an average enhancement of 26.72% in accuracy and a notable average reduction of 70% in ASR under two targeted attack scenarios.

- We demonstrate the compatibility of FEDQV with SECAGG, and our empirical evaluation confirms that this integration enhances resistance against two privacy attacks.

3.2. Background

3.2.1. Election Mechanisms in FL

Election mechanisms are widely used in distributed systems for choosing a coordinator from a collection of processes [131], [132]. Likewise, there exist works that explore the value of the election mechanism for the aggregation step of FL. Plurality voting is employed in *FedVote* [133] for weighting the local updates, and *FedVoting* [134] for treating the validation results as votes to decide the optimal model. Also in [135], the authors propose two forms of election coding, random Bernoulli codes and deterministic algebraic codes, for discovering majority opinions for the aggregation step. *DETOX* [136] proposes a hierarchical aggregation step based on majority votes upon groups of updates. Finally, *DRACO* [137] and *ByzShield* [138] also employ majority voting to fend off attacks against the aggregation step. All the aforementioned election mechanisms suffer from the tyranny of the majority problem in election systems [139]. In FL, this means that if attackers manage to control the majority of votes, then via poisoning their tyranny will manifest itself as a degradation of the accuracy of the FL model used by the minority.

To address these limitations, QV is proposed as a solution that combines simplicity, practicality, and efficiency under relatively broad conditions. QV considers a quadratic vote pricing rule, inspired by economic theory, under which voters can purchase votes at ever-increasing prices within a predetermined voting budget. The advantages of QV over *1p1v* have a rigorous theoretical basis, which of course applies also to the use of QV in FL. For any type of symmetric Bayes-Nash equilibrium, the price-taking assumption approximately holds for all voters, as a result, the expected inefficiency of QV is bounded by constant [140]. This theoretical analysis [141], [142] combined with strong empirical validation, both at the laboratory [143] and on the field [144], suggest that QV is near-perfectly efficient and more robust than *1p1v* which, as already explained, forms the basis of contemporary FL aggregation mechanisms. The advantages of QV can also be observed from the viewpoint of collusion, which is generally deterred either by unilateral deviation incentives or by the reactions of non-participants [130].

3.2.2. FL Aggregation against Poisoning and Privacy Attacks

There exist several Byzantine-robust FL aggregation methods for mitigating poisoning attacks either by leveraging statistic-based outlier detection techniques [1], [39], [53], [102] or by utilising auxiliary labelled data collected by the aggregation server in order to verify the correctness of the received gradients [86], [145]. Both approaches, though, require examining the properties of the updates of individual parties, which can jeopardise

their privacy due to inference [106] and reconstruction attacks [37], [70] mounted by an honest but curious aggregation server.

To fight against privacy attacks, secure aggregation (SA) protocols have been proposed as potential countermeasures [146]. Secure aggregation can be achieved using four main privacy-enhancing technologies: differential privacy [147], trusted-execution environment (TEE) [148], secure shuffling under anonymity assumptions [149], and cryptography. Among those, secure aggregation based on cryptography is the most widely studied [110]–[112]. Although crypto-based secure aggregation provides strong security guarantees compared to alternatives, i.e., differential privacy, it also suffers from high computational and communication overhead. Consequently, these crypto-based secure aggregations are primarily established in the FEDAVG framework due to the impracticality of integrating them into Byzantine-robust FL aggregation methods, owing to the computational complexity in these aggregations.

In response to this limitation, FEDQV emerges as a promising solution, offering enhanced resilience against poisoning attacks while inheriting the simplicity of FEDAVG. Notably, this simplicity facilitates FEDQV’s integration with crypto-based secure aggregations, thereby bolstering defences against privacy attacks while also exhibiting superior resilience against poisoning attacks compared to FEDAVG. Although a few FL aggregation approaches [150]–[152] can be adapted to incorporate secure aggregation, they still rely on majority voting as the aggregation scheme, which can be integrated with FEDQV to enhance its robustness against poisoning attack.

3.3. FEDQV: Quadratic Voting in FL

3.3.1. Federated Learning Setting

Consider an FL system involving N parties and a central server. During the the training round t , a subset of parties $\mathcal{S}^t$ is selected to participate in the training task. Each party i has the local dataset $\mathcal{D}_i$ with $|\mathcal{D}_i|$ samples (voters), drawn from Non-IID distribution $\mathcal{X}_i(\mu_i, \sigma_i^2)$. The goal of using FL is to learn a global model for the server. Given the loss function $\ell(\mathbf{w}; \mathcal{D})$, the objective function of FL can be described as

$$\mathcal{L}(\mathbf{w}) = \mathbb{E}_{\mathcal{D} \sim \mathcal{X}} [\ell(\mathbf{w}; \mathcal{D})]$$

Therefore, the task becomes:

$$\mathbf{w}^* = \arg \min_{\mathbf{w} \in \mathbb{R}^d} \mathcal{L}(\mathbf{w})$$

To find the optimal $\mathbf{w}^*$, SGD is employed to optimise the objective function. Let T be the total number of every part’s SGD, E be the local iterations between two communication rounds, and thus $\frac{T}{E}$ is the number of communication rounds.

The FL model training process entails several rounds of communication between the parties and the server, including broadcasting, local training, and aggregation, as demon-

strated in Algorithm 3. The global model is first initialised at a random state by the server and, then, in round t the following steps are taken:

Broadcast: The server first randomly selects a subset of parties $\mathcal{S}'$ ($|\mathcal{S}'| = \mathcal{C} \geq 1$) and then distributes the latest global model as the global proposal to all chosen parties.

Local Training: The selected party i performs local computation based on the global proposal and its local dataset $\mathcal{D}_i$, and sends the update $\mathbf{w}_i^t$ back to the server:

$$\mathbf{w}_i^t \leftarrow \mathbf{w}_i^{t-1} - r_{t-1} \frac{\partial \ell_i(\mathbf{w}_i^{t-1}; \mathcal{D}_i)}{\partial \mathbf{w}}$$

where r_{t-1} is the learning rate.

Aggregation: The server receives the model updates from all participating parties and aggregates them to update the global model.

For aggregation rule, FEDAVG uses the fraction of the local training sample size of each party over the total training samples as the weight of a party:

$$\mathbf{w}^{t+1} = \frac{1}{\sum_{i \in \mathcal{S}'} |\mathcal{D}_i|} \sum_{i \in \mathcal{S}'} |\mathcal{D}_i| \cdot \mathbf{w}_i^t$$

Similar to *IpIv*, each sample here represents a single voter, and since party i possesses $|\mathcal{D}_i|$ samples, it is able to cast $|\mathcal{D}_i|$ votes for its local model during the aggregation. Hence, the global proposal is a combination of all parties' local proposals weighted by their votes.

We assume that the participating parties may behave maliciously. In this setting, malicious parties can deliberately submit manipulated training sample sizes to the central server. Their objectives may include influencing the global model's performance by artificially inflating their contribution through overweighing. By reporting exaggerated sample sizes, malicious parties can amplify the impact of their local updates during aggregation, thereby skewing the global model towards their desired outcomes. This behaviour can degrade overall model quality, introduce targeted biases, or even embed backdoors.

3.3.2. Motivations

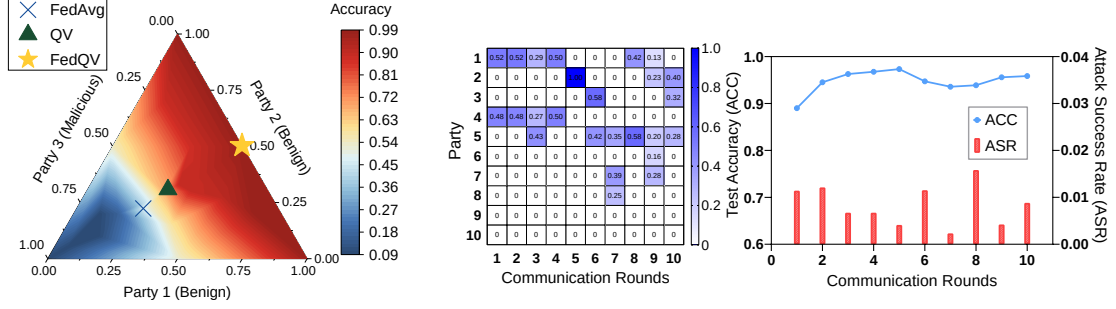
In FEDAVG, the aggregation weight assigned to each party is proportional to its claimed dataset size. This grants malicious parties a disproportionately large influence on the global model. To address this issue, we propose a robust aggregation algorithm inspired by elections based on Quadratic Voting (QV). To show the rationale behind the algorithm design, we consider a toy use case involving two benign parties and one malicious party in an FL system. Through this case, we illustrate the shortcomings of FEDAVG and motivate the need for our proposed approach. The two benign parties and one malicious party claim possession of training datasets with the dataset sizes set to $\{1, 1, 2\}$, respectively. Importantly, the malicious party asserts a larger training dataset than the benign ones. These claims are reported to the central server, which lacks access to the raw data of the participating parties for verification purposes. Under FEDAVG, the aggregation weights are

directly proportional to the claimed dataset sizes, resulting in $\{1, 1, 2\}$. This grants the malicious party a disproportionately large influence. As shown in Figure 3.1a, the global model suffers from very low accuracy due to the malicious party’s attack.

To limit the influence of parties claiming large datasets, we first apply *quadratic computation* from QV into the FL. The quadratic computation involves taxing the aggregation weights with the square root. Consequently, the corresponding aggregation weights in QV are adjusted to $\{1, 1, \sqrt{2}\}$. We conduct a 10-round training of a multi-layer CNN on the MNIST dataset, during which the malicious party executes a backdoor attack under the same settings as discussed in Section 3.5. The test accuracy is colour-coded and presented in Figure 3.1a, where the vertices of the triangle correspond to different parties, and their positions inside the triangle reflect their respective aggregation weights. As expected, the compromised accuracy in FedAvg arises from the presence of a malicious party that possesses or falsely claims to possess a larger dataset. Compared to FedAvg, QV, with the adjusted weights, exhibits superior accuracy. This suggests that the quadratic computation in QV can enhance performance by restraining the influence of attackers within FedAvg by taxing their aggregation weights more than linear.

However, the direct incorporation of quadratic computation alone proves insufficient in capturing the voting preferences of each party, a critical aspect of QV. In QV, parties with pronounced disagreements with a proposal express their dissent by casting more votes to reject it. To capture each party’s voting preference into the FL system, we refine the integration by requiring parties to submit the similarity of their local model with the previous round’s global model. The similarity score serves as a measure of agreement, where higher scores indicate stronger alignment between the local model (proposal) and the global model (proposal). Consequently, parties with higher similarity scores will cast fewer votes in this round, reflecting their reduced need for adjustment to the existing global model. Conversely, parties showing greater dissimilarity ($1 - s_i^t$) between their local proposal and the global one cast more votes. This strategic allocation empowers parties with substantial disagreements to wield more significant influence over the impending global proposal w^t , thereby making it a closer alignment with their local proposal w_i^t .

Based on these similarity scores, the voting calculation is conducted to determine the votes, serving as the aggregation weights. Rather than having parties directly calculate their votes, which could lead to malicious attempts, we introduce the FedQV mechanism. This approach adopts a more secure approach by delegating the vote allocation task to the server. It employs a masked voting rule to obscure the vote calculation process from the parties, thereby preventing them from knowing the exact votes they have cast for aggregation. This confidentiality measure is crucial to maintaining the integrity and impartiality of the voting process. Built upon the lack of awareness regarding the voting calculation process among parties, the FedQV system incorporates two additional defensive mechanisms: a limited voting budget and outlier detection, which are known only by the server. Given that parties are unaware of their budget and the possibility of their updates being detected as abnormal, attempts by malicious parties to manipulate their updates and



(a) Aggregation weights (position within the triangle) and corresponding test accuracy (colour-coded) with three parties (two benign and one malicious). FEDAvg is located at the bottom left ; QV is positioned around the centre ; FEDQV is situated along the right triangle side. (b) FEDQV aggregation weights for each communication round, involving 10 parties (on the left). The first 6 parties are benign, while the subsequent 7 are malicious. The presentation is complemented by the corresponding metrics (on the right), including test accuracy (ACC) and attack success rate (ASR) of the global model. This scenario is observed during 10 communication rounds in the MNIST dataset under a Backdoor attack.

Figure 3.1: Visualization of Aggregation Weights and Robustness Performance of FEDQV

overcast their votes entail a dual risk. There is a potential for exclusion from the aggregation process through outlier detection or unknowingly reaching their voting budget limit, resulting in the assignment of no votes. These protections serve as a deterrent against potential poisoning attacks, ensuring the overall robustness of the FEDQV system in the face of adversarial threats. Details of the design of FEDQV are presented in the next subsection.

Returning to our previous toy example, the implementation of FEDQV leads to the allocation of weights $\{1, 1, 0\}$, as depicted in Figure 3.1a. This weight distribution effectively excludes the malicious party from the aggregation process. Consequently, this exclusion contributes to the improved accuracy of the resulting global model. To further illustrate how FEDQV assigns aggregation weights to different parties during training, consider another toy case with 10 parties in FL, where 4 of them are attackers. The settings remain the same as the first toy case, and the results are presented in Figure 3.1b. In the left of Figure 3.1b, the first six parties are benign, and the rest are malicious. Notably, during the 10 communication rounds, only two attackers (parties 7 and 8) successfully participated in the aggregation, with party 7 participating twice and party 8 once, while most others did not contribute, resulting in an aggregation weight of 0 for them. This outcome demonstrates FEDQV’s effectiveness in expelling malicious parties. Even when a round includes two malicious parties, the decrease in test accuracy is limited, and the Attack Success Rate (ASR) remains low ($<2\%$) as shown in the right of Figure 3.1b. This demonstration underscores FEDQV’s capability to mitigate the influence of malicious parties, thus preventing significant damage to the global model.

Why Quadratic. The uniqueness of Quadratic Voting, compared to other types of functions x^a , is proven in Theorem 1 [128]: "A vote pricing rule is robustly optimal if and only if it is quadratic". The central idea of the proof is that quadratic functions are the only ones with linear derivatives. Hence they are the only functions where a voter buying votes equates her marginal benefit and cost at a number of votes proportional to her value. We also empirically compare our methods using other nonlinear functions in Section 3.5.9, demonstrating its consistency with the aforementioned theoretical results.

3.3.3. FEDQV Design

We use QV in FL to overcome the drawback of *IpIv*, which improves the robustness of aggregation in comparison to FEDAVG without compromising any efficiency. Figure 3.2 provides an overview of the FEDQV algorithm, which comprises two key components: 1. *similarity computation*: Executed on the party side, this component captures the voting preference of each party, which is a critical aspect of QV. Here, the similarity score serves as a metric of consensus, with higher scores indicating closer alignment between the local model and the global model. Consequently, parties with higher similarity scores cast fewer votes, reflecting their reduced need for adjustment to the existing global model. Conversely, parties showing greater dissimilarity between their local model and the global one cast more votes. This dynamic reflects the essence of QV, allowing voters to express the degree of their voting will, by empowering parties with substantial disagreements (strong voting will) to exert more significant influence over the impending global model. 2. *voting scheme*: Managed on the server side, this component is designed to mitigate potential poisoning attacks and untruthful strategic behaviour by parties. It allocates votes to parties based on their similarity scores, but through defence layers, ensuring a robust and trustworthy weight allocation and aggregation process. We detail each component in the following subsections.

Similarity Computation: In round t , based on the server instructions, party i ($i \in \mathcal{S}'$) trains its local model $\mathbf{w}_i^t$, which can be regarded as its local proposal. Following the local training phase, party i computes a similarity score s_i^t utilising cosine similarity, quantifying the alignment between its locally trained model $\mathbf{w}_i^t$ and the previous global model $\mathbf{w}^{t-1}$.

$$s_i^t = S_{\cos}(\mathbf{w}_i^t, \mathbf{w}^{t-1}) = \frac{\langle \mathbf{w}_i^t, \mathbf{w}^{t-1} \rangle}{\|\mathbf{w}_i^t\| \cdot \|\mathbf{w}^{t-1}\|}$$

Notably, the cosine similarity function can be adapted to different similarity metrics, such as L2 distance, to better suit specific tasks. In this context, a higher s_i^t value indicates a stronger agreement with the previous global model (proposal). Once selected parties finish training, they send their updates $\mathbf{w}_i^t$ to the server, with the message $\langle s_i^t \rangle$.

It's noteworthy that similarity calculation can also be launched on the server side. However, this approach presents certain risks: (i) it exposes the system to privacy attacks initiated by the server, and (ii) it may produce distorted similarity scores due to the util-

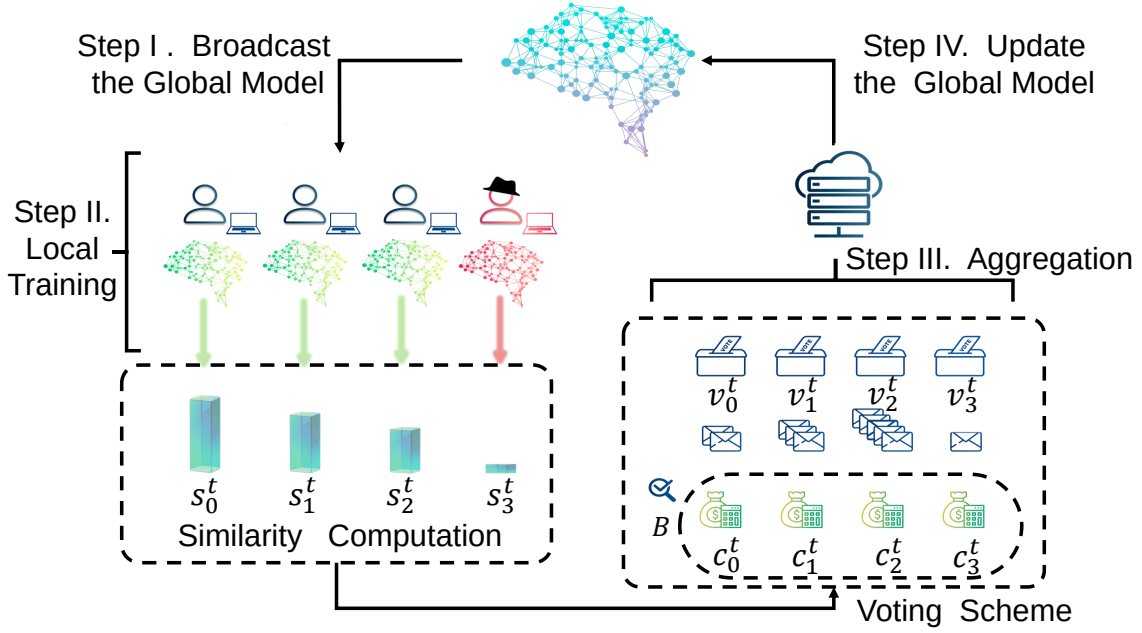


Figure 3.2: Overview of FEDQV algorithm.

isation of regularisation and privacy-preservation methods [153] at the party side. Given our primary objective of comparing FEDQV with FEDAvg, where weights are also computed at the party side, we choose to conduct the calculation on the party side within FEDQV. Additionally, to counteract potential malicious attempts launched from the party side, we have the option to introduce defence mechanisms on the server side, such as other Byzantine-robust aggregations, as detailed in Section 3.5.8. This approach makes it more challenging for attackers to mount successful attacks compared to FEDAvg.

Voting Scheme (Server Side): Upon receiving the updates and messages from selected parties, the server proceeds with the following steps:

- The server normalises the similarity scores using Min-Max scaling to obtain $\bar{s}_i^t$ as:

$$\bar{s}_i^t = \text{norm}(s_i^t, \{s_{i \in [S']}\}) = \frac{s_i^t - \min(s_{i \in [S']})}{\max(s_{i \in [S']}) - \min(s_{i \in [S']})} \quad (3.1)$$

Following normalisation, parties no longer being aware of their exact similarity scores $\bar{s}_i^t$. Consequently, inferring the following information becomes challenging for them.

- The server employs a penalty mechanism for abnormal similarity scores, specifically when $\bar{s}_i^t$ falls below θ or exceeds $1 - \theta$. Here, θ represents the similarity threshold and is established to capture scores that are excessively small or large, indicating potential anomalies. In response to such abnormality, the server enforces a penalty by reducing the budgets B_i^t allocated to the respective party i at round t as:

$$B_i^t = \max(0, B_i^t + \ln \bar{s}_i^t - 1) \quad (3.2)$$

- The server calculates the voice credit c_i^t for party i utilising the masked voting rule $\mathcal{H}$ as:

$$c_i^t = \mathbb{H}(\bar{s}_i^t) = (-\ln \bar{s}_i^t + 1) \mathbb{1}_{\theta < \bar{s}_i^t < 1-\theta} \quad (3.3)$$

Here, the voice credits signify the price party i is required to pay in round t for its local proposal. Parties with higher similarity scores, indicating stronger agreement with the global proposal, are allocated fewer credit votes from the server. Conversely, parties showing greater dissimilarity ($1 - \bar{s}_i^t$) between their local proposal and the global one receive an increased allocation of c_i^t . This mechanism empowers parties with substantial disagreements to exert greater influence over the forthcoming global proposal $\mathbf{w}^t$. Notably, parties with abnormal similarity scores—either too low or too high—are excluded from participating in the aggregation and receive zero voice credits. For similarity scores surpassing the threshold θ , such instances may have two possibilities: firstly, these parties closely resemble the previous global model, thereby contributing negligible new information; secondly, they may potentially result from forgery owing to their excessively high similarity. In both scenarios, these parties are assigned small weights according to the voting rule. Therefore, excluding them from a security perspective is advisable, as their exclusion does not significantly impact the model's performance.

- The server checks the budget B_i for each party and computes their final votes v_i^t as:

$$v_i^t = \sqrt{\min(c_i^t, \max(0, B_i^t))} \quad (3.4)$$

based on Equation 3.4, voting corresponds to the square root of the minimum value between the credit voice and the remaining budget. This limitation arises from the fact that the credit voice spent by a party cannot exceed its remaining budget.

- The server updates the budget as:

$$B_i^t = \max(0, B_i^t - (v_i^t)^2) \quad (3.5)$$

Thus, the server determines the weight (v_i^t) of party i for aggregation and generates the updated global model $\mathbf{w}^{t-1}$, reflecting the collective opinion of selected parties. The Algorithm 3 summarises all these steps of FEDQV.

Malicious Party. In cases where malicious parties attempt to manipulate the similarity scores, their capabilities are restricted by:

(a) *No knowledge of the voting process.* Only the server possesses knowledge of each party's remaining budget and the number of actual votes cast in the current round. The lack of knowledge about the voting calculation process that contains outlier detection, coupled with the parties' unawareness of their limited voting budget, exposes malicious parties to the risk of exclusion from the aggregation process.

(b) *Punitive Measures:* FEDQV, with its masked voting rule that contains outlier detection and limited budget, empower the system to penalise and potentially remove malicious parties who attempt to conduct poisoning attacks;

Algorithm 3: FEDQV

Input : $\mathbf{w}^0 \leftarrow$ random initialisation; $B, \theta \leftarrow$ FEDQV parameters

Server :

```
1 for Iteration  $t \leftarrow 1$  to  $\frac{T}{E}$  do
2   Broadcast  $\mathbf{w}^{t-1}$  to randomly selected set of parties  $\mathcal{S}^t$  ( $|\mathcal{S}^t| = \mathcal{C} \geq 1$ );
3   Receive the local updates  $(\mathbf{w}_i^t, s_i^t)$  from selected parties ( $i \in \mathcal{S}^t$ );
4   Normalisation:  $\bar{s}_i^t \leftarrow$  Eq.(3.1),  $i \in \mathcal{S}^t$ ;
5   for  $i \leftarrow 1$  to  $N$  do in parallel
6     if  $\bar{s}_i^t \leq \theta$  or  $\bar{s}_i^t \geq 1 - \theta$  then
7       Update  $B_i^t \leftarrow \max(0, B_i + \ln \bar{s}_i^t - 1)$ 
8       Credit voice  $c_i^t \leftarrow$  Equation 3.3, Vote  $v_i^t \leftarrow$  Equation 3.4;
9       Budget  $B_i^t \leftarrow \max(0, B_i^t - (v_i^t)^2)$  // Update the budget
10    end forpar
11    return  $\mathbf{w}_n^t \leftarrow \sum_{i=1}^N \frac{v_i^t}{\sum_{i=1}^N v_i^t} \mathbf{w}_{i,n}^t$ 
12 end for
```

Party :

```
1 for Party  $i \in \mathcal{S}^t$  do in parallel
2   Receive the global model  $\mathbf{w}^{t-1}$  ;
3   for local epoch  $e \leftarrow 1$  to  $E$  do
4      $\mathbf{w}_i^t \leftarrow \mathbf{w}_i^{t-1} - r_{t-1} \frac{\partial \ell_i(\mathbf{w}_i^{t-1}; \mathcal{D}_i)}{\partial \mathbf{w}}$  // Local training
5   end for
6   Calculate the similarity score:  $s_i^t \leftarrow \frac{\langle \mathbf{w}_i^t, \mathbf{w}^{t-1} \rangle}{\|\mathbf{w}_i^t\| \cdot \|\mathbf{w}^{t-1}\|}$ ;
7   Send  $(\mathbf{w}_i^t, s_i^t)$ 
8 end forpar
```

(c) *Limited Influence*: Even if a manipulated similarity score is accepted by the server, the influence the malicious party can exert is inherently constrained due to the nature of QV and the limited budgets, minimising the potential damage.

Benefits of FEDQV. The benefits of FEDQV center around two key aspects:

- **Truthful Mechanism.** FEDQV is a *truthful* mechanism [154] as we prove in Theorem 3.4.19. This means that this mechanism compels the parties, even malicious ones, to tell the truth about their votes (weights) for aggregation, rather than any possible lie. This truthfulness is reinforced by the aforementioned several defence layers.

- **Ease of Integration and Compatibility.** FEDQV is highly adaptable and can be seamlessly integrated into Byzantine-robust FL defence schemes with minimal adjustments, specifically by modifying the aggregation weight calculation while leaving other algorithm components unchanged. This integration is demonstrated in Section 3.5.8. Fur-

thermore, similar to FEDAVG, FEDQV boasts efficient communication and simplicity, rendering it compatible with various mechanisms employed in FL. It can effortlessly incorporate the regularisation, sparsification, and privacy modules, encompassing techniques such as clipping [153], gradient compression [155], differential privacy [156], and secure aggregation [110].

Complexity Analysis of FEDQV. Let d be the number of the model parameters. The term $\text{cost}(f)$ denotes the computational cost of one forward-backwards pass *per sample* for the chosen model. In practice, $\text{cost}(f)$ is a function of both the number of model parameters d , and the model architecture—e.g., fully connected layers, convolutional layers, attention mechanisms, and other architectural components.

- **Computation cost.** In standard FL (FEDAVG), each party i trains its local model on its private dataset $\mathcal{D}_i$ for E local epochs. The local computational cost per communication round at party i is therefore given by $O(E \cdot |\mathcal{D}_i| \cdot \text{cost}(f))$. In the QV mechanism, the computation at each party includes both local training and the calculation of similarity scores. The cosine similarity between two d -dimensional vectors incurs a computational cost of $O(d)$. On the server side, the primary task is aggregating model updates received from N parties. For each model update with d parameters, the aggregation cost among N parties is $O(dN)$. Under the QV mechanism, an additional overhead arises from the voting scheme (as described in Lines 4–10 of Algorithm 3), which incurs a cost of $O(N)$. Therefore, the additional computational complexity introduced by the QV mechanism, compared to standard FL, is $O(d + N)$.

- **Communication cost.** In each communication round of standard FL (e.g., FEDAVG), the server broadcasts the global model, consisting of d parameters, to N participating parties, incurring a cost of dN . Each party, after local training, subsequently uploads its locally updated model back to the server, adding a cost of dN . Hence, the total per-round communication cost is $2dN$. Over $\frac{T}{E}$ total global communication rounds, the overall communication is $O\left(\frac{dNT}{E}\right)$. In FEDQV, each party additionally transmits a scalar similarity score alongside its local model update. This increases the upload cost per party to $d + 1$. As a result, the per-round cost increases to $dN + (d + 1)N = 2dN + N$, and the total communication cost over all rounds becomes $O\left(\frac{(d+1)NT}{E}\right)$. Thus, compared to standard FL, FEDQV introduces an additional communication overhead of N units per round (one scalar per party), which accumulates to $O\left(\frac{NT}{E}\right)$ over the entire FL process.

3.3.4. FEDQV with Adaptive Budgets

In democratic elections, all individuals are typically granted equal voting rights, entailing an equal voting budget. In FL, however, it often makes sense to give malicious parties fewer votes than honest ones. Thus to improve the robustness of standard FEDQV, we combine it with the reputation model in [1] to assign an unequal budget based on the reputation score of parties in each round t . Specifically, if a party's reputation score R^t surpasses a predefined threshold λ , we increase their budget, and vice versa. We present a

Algorithm 4: FEDQV with Adaptive Budget

Input : $w_i^t, c_i^t, B_i^t \leftarrow \text{FEDQV}; \kappa, a, W, M, \lambda, \delta \leftarrow \text{Reputation model parameters}$

```
1 for  $i \in \mathcal{S}^t$  do
2   for  $j \leftarrow 1$  to  $M$  do
3     Subjective Observations  $(P_i^t, Q_i^t) := \text{IRLS}(w_{i,j}^t, \delta)$ ;
4   end for
5   Reputation Score  $R_i^t := \text{Rep}(P_i^t, Q_i^t, \kappa, a, W)$ 
6   Budget  $B_i^t \leftarrow R_i^t \mathbb{1}_{\lambda \leq R_i^t} + B_i^t$ , Credit voice  $c_i^t \leftarrow (R_i^t + c_i^t) \mathbb{1}_{\lambda \leq R_i^t}$ 
7 end for
```

summary of this combination in Algorithm 4, with a detailed explanation, expanding on the well-established components from the original paper. We provide empirical evidence in Section 3.5.7 showcasing the substantial performance improvements achieved by the enhanced version of FEDQV featuring an adaptive budget.

Here we present a concise elucidation of key components of the Algorithm 4 as follows:

- **IRLS** (Iteratively Reweighted Least Squares): IRLS serves as an optimisation technique employed to solve specific regression problems. Within [1], IRLS is utilised to compute the Subjective Observations of participating parties based on their parameter's confidence score, which is calculated using the repeated-median regression technique.
- **Subjective Observations**: Positive observations denoted by P_i^t signify acceptance of an update, while negative observations denoted by Q_i^t indicate rejection. Consequently, positive observations enhance a party's reputation, and negative ones have the opposite effect.
- **Reputation Score Calculation**: The reputation score of a party is determined using a subjective logic model, formulated as follows:

$$R_i^t = \frac{\kappa P_i^t + Wa}{\kappa P_i^t + \eta Q_i^t + W}$$

Regarding the integration of the reputation model, our objective is to demonstrate how combining FEDQV with the reputation model enables the allocation of unequal budgets, thereby enhancing the robustness of standard FEDQV. This integration's adaptability extends beyond a single reputation model, allowing customisation to suit various needs. The example presented in the paper serves to showcase the concept's viability.

3.4. Theoretical Analysis

In this section, we establish the convergence guarantees and truthfulness properties of FEDQV. Our first result is Theorem 3.4.9 that states FEDQV converges to the global optimal solution at a rate of $\mathcal{O}(\frac{1}{T})$, comparable to the convergence rate exhibited by FEDAvg. Theorem 3.4.12 extends this statement, affirming that FEDQV converges to a near-optimal solution in the presence of malicious parties, with the resulting performance gap determined by the percentage of malicious parties. The empirical convergence performance of our algorithm, as gauged by metric test accuracy and training loss, is consistent with our theoretical analysis, as elaborated in the following section. Our final result, Theorem 3.4.19, establishes that FEDQV is a truthful mechanism, wherein honesty emerges as the dominant strategy within this framework.

3.4.1. Convergence

We start by stating our assumptions, which are standard and common for such types of analysis and per recent works such as [1], [53], [66], [86], [102], [157], [158].

Assumption 3.4.1. The loss functions are L -smooth, which means they are continuously differentiable and their gradients are Lipschitz-continuous with Lipschitz constant $L > 0$, whereas:

$$\begin{aligned} \forall \mathbf{w}_1, \mathbf{w}_2 \in \mathbb{R}^d, \|\nabla \mathcal{L}(\mathbf{w}_1) - \nabla \mathcal{L}(\mathbf{w}_2)\| &\leq L \|\mathbf{w}_1 - \mathbf{w}_2\| \\ \|\nabla \ell(\mathbf{w}_1; \mathcal{D}) - \nabla \ell(\mathbf{w}_2; \mathcal{D})\| &\leq L \|\mathbf{w}_1 - \mathbf{w}_2\| \end{aligned}$$

Assumption 3.4.2. The loss function $\ell(\mathbf{w}_i, D)$ are μ -strongly convex:

$$\begin{aligned} \exists \mu > 0, \forall \mathbf{w}_1, \mathbf{w}_2 \in \mathbb{R}^d, \nabla \ell(\mathbf{w}^*; \mathcal{D}) = 0, \nabla \mathcal{L}(\mathbf{w}^*) = 0 \\ 2(\mathcal{L}(\mathbf{w}_1) - \mathcal{L}(\mathbf{w}_2)) &\geq 2\langle \nabla \mathcal{L}(\mathbf{w}_2), \mathbf{w}_1 - \mathbf{w}_2 \rangle + \mu \|\mathbf{w}_1 - \mathbf{w}_2\|^2 \\ 2(\ell(\mathbf{w}_1; \mathcal{D}) - \ell(\mathbf{w}_2; \mathcal{D})) &\geq 2\langle \nabla \ell(\mathbf{w}_2; \mathcal{D}), \mathbf{w}_1 - \mathbf{w}_2 \rangle + \mu \|\mathbf{w}_1 - \mathbf{w}_2\|^2 \end{aligned}$$

Assumption 3.4.3. The dissimilarity of gradients of $\mathbf{w}$ is bounded:

$$\begin{aligned} \forall \{u_i \geq 0\}_{i=1}^N, \sum_{i=1}^N u_i = 1, \exists \rho^2 \geq 0, \mathcal{G}^2 \geq 0 \\ \sum_{i=1}^N u_i \|\nabla \ell(\mathbf{w}; \mathcal{D})\|^2 \leq \rho^2 \left\| \sum_{i=1}^N u_i \nabla \ell(\mathbf{w}; \mathcal{D}) \right\|^2 + \mathcal{G}^2 \end{aligned}$$

If local functions are identical to each other, then $\rho^2 = 1, \mathcal{G}^2 = 0$

Assumption 3.4.4. The variance of gradients of $\mathbf{w}$ is bounded:

$$\forall \mathbf{w} \in \mathbb{R}^d, \exists \mathcal{V}_{\mathbf{w}} < \infty, \mathbb{E} \|\nabla \ell(\mathbf{w}; \mathcal{D}) - \mathbb{E}(\nabla \ell(\mathbf{w}; \mathcal{D}))\|^2 \leq \mathcal{V}_{\mathbf{w}}$$

The lemmas we utilise in the proof of Theorem 3.4.9 and Theorem 3.4.12, are presented below.

Lemma 3.4.5. *From Assumption 3.4.1 and 3.4.2, $\mathcal{L}(\mathbf{w})$ is L -smooth and μ -strongly convex. Then $\forall \mathbf{w}_1, \mathbf{w}_2 \in \mathbb{R}^d$, one has*

$$\langle \nabla \ell(\mathbf{w}_1) - \nabla \ell(\mathbf{w}_2), \mathbf{w}_1 - \mathbf{w}_2 \rangle \geq \frac{L\mu}{L+\mu} \|\mathbf{w}_1 - \mathbf{w}_2\|^2 + \frac{1}{L+\mu} \|\nabla \ell(\mathbf{w}_1) - \nabla \ell(\mathbf{w}_2)\|^2$$

Lemma 3.4.6. *Assume Assumption 3.4.1, Assumption 3.4.2 and Lemma 3.4.5 hold, Let $p_i^t = \frac{1}{C} \mathbb{1}_{i \in S^t}$, we have*

$$\begin{aligned} \|\mathbf{w}^{t-1} - r \nabla \mathcal{L}(\mathbf{w}^{t-1}) - \mathbf{w}^*\|^2 &\leq \sum_{i=1}^N p_i^{t-1} \|\mathbf{w}^{t-1} - \mathbf{w}_i^{t-1}\|^2 \\ &\quad + \left(r^2 (1 + L^2) - \frac{2rL\mu + 1}{L + \mu} \right) \|\mathbf{w}^{t-1} - \mathbf{w}^*\|^2 \end{aligned}$$

Lemma 3.4.7. *Assume Assumption 2.3.3 holds, it follows that*

$$\mathbb{E} \sum_{i=1}^N p_i^{t-1} \|\mathbf{w}^{t-1} - \mathbf{w}_i^{t-1}\|^2 \leq (E-1)^2 r_{t-1}^2 \left(\rho^2 L^2 \|\mathbf{w}^{t-1} - \mathbf{w}^*\|^2 + Q^2 \right)$$

Lemma 3.4.8. *Assume Assumption 2.3.4 holds, according to our Algorithm 2, it follows that*

$$\mathbb{E} \|\mathcal{F}(\mathbf{w}^{t-1}) - \nabla \mathcal{L}(\mathbf{w}^{t-1})\|^2 \leq (1 - 2\theta) C \mathcal{V}_{\mathbf{w}} \sqrt{B}$$

Where

$$\mathcal{F}(\mathbf{w}^{t-1}) = \sum_{i \in S^{t-1}} p_i^{t-1} \nabla \ell(\mathbf{w}_i^{t-1}; \mathcal{D}_i^{t-1})$$

Proof of Lemmas

Lemmas 3.4.5, Lemmas 3.4.6, Lemmas 3.4.7 and Lemmas 3.4.8 are all the lemmas we utilise during the proof of Theorem 3.4.9, and we prove them in that order. Notice, Lemmas 3.4.5 are used in the proof Lemmas 3.4.6, and Theorem 3.4.9 is proved using Lemmas 3.4.6, Lemmas 3.4.7 and Lemmas 3.4.8 in order.

Proof of Lemma 3.4.5

Proof. Let $g(\mathbf{w}) = \ell(\mathbf{w}) - \frac{\varsigma}{2} \|\mathbf{w}\|^2$. Base on the Assumption 3.4.2, we have $g(\mathbf{w})$ is $(L - \varsigma)$ -strongly convex. From [159] Equation 3.6, we have

$$\langle \nabla \ell(\mathbf{w}_1) - \nabla \ell(\mathbf{w}_2), \mathbf{w}_1 - \mathbf{w}_2 \rangle \geq \frac{1}{L} \|\nabla \ell(\mathbf{w}_1) - \nabla \ell(\mathbf{w}_2)\|^2 \quad (3.6)$$

Hence,

$$\langle \nabla g(\mathbf{w}_1) - \nabla g(\mathbf{w}_2), \mathbf{w}_1 - \mathbf{w}_2 \rangle \geq \frac{1}{L - \varsigma} \|\nabla g(\mathbf{w}_1) - \nabla g(\mathbf{w}_2)\|^2 \quad (3.7)$$

Now We have

$$\begin{aligned} & \langle \nabla \left(\ell(\mathbf{w}_1) - \frac{\varsigma}{2} \|\mathbf{w}_1\|^2 \right) - \nabla \left(\ell(\mathbf{w}_2) - \frac{\varsigma}{2} \|\mathbf{w}_2\|^2 \right), \mathbf{w}_1 - \mathbf{w}_2 \rangle \\ & \geq \frac{1}{L + \mu} \left\| \nabla \left(\ell(\mathbf{w}_1) - \frac{\varsigma}{2} \|\mathbf{w}_1\|^2 \right) - \nabla \left(\ell(\mathbf{w}_2) - \frac{\varsigma}{2} \|\mathbf{w}_2\|^2 \right) \right\|^2 \end{aligned} \quad (3.8)$$

And therefore

$$\begin{aligned} & \langle \nabla \ell(\mathbf{w}_1) - \nabla \ell(\mathbf{w}_2), \mathbf{w}_1 - \mathbf{w}_2 \rangle - \langle \varsigma \mathbf{w}_1 - \varsigma \mathbf{w}_2, \mathbf{w}_1 - \mathbf{w}_2 \rangle \\ & \geq \frac{1}{L - \varsigma} \|(\nabla \ell(\mathbf{w}_1) - \nabla \ell(\mathbf{w}_2)) - (\varsigma \mathbf{w}_1 - \varsigma \mathbf{w}_2)\|^2 \end{aligned} \quad (3.9)$$

Refer to Assumption 3.4.1, we obtain

$$\begin{aligned} \langle \nabla \ell(\mathbf{w}_1) - \nabla \ell(\mathbf{w}_2), \mathbf{w}_1 - \mathbf{w}_2 \rangle & \geq \frac{L\varsigma}{L - \varsigma} \|\mathbf{w}_1 - \mathbf{w}_2\|^2 - \frac{2\varsigma}{L - \varsigma} \langle \nabla \ell(\mathbf{w}_1) - \nabla \ell(\mathbf{w}_2), \mathbf{w}_1 - \mathbf{w}_2 \rangle \\ & + \frac{1}{L - \varsigma} \|\nabla \ell(\mathbf{w}_1) - \nabla \ell(\mathbf{w}_2)\|^2 \\ & \geq -\frac{L\varsigma}{L - \varsigma} \|\mathbf{w}_1 - \mathbf{w}_2\|^2 + \frac{1}{L - \varsigma} \|\nabla \ell(\mathbf{w}_1) - \nabla \ell(\mathbf{w}_2)\|^2 \end{aligned} \quad (3.10)$$

Let $\varsigma = -\mu$, then we conclude the proof of Lemma 3.4.5.

□

Proof of Lemma 3.4.6

Proof. We have

$$\begin{aligned} \|\mathbf{w}^{t-1} - r_{t-1} \nabla \mathcal{L}(\mathbf{w}^{t-1}) - \mathbf{w}^*\|^2 & = \underbrace{\|\mathbf{w}^{t-1} - \mathbf{w}^*\|^2}_{\text{A1}} - 2r_{t-1} \langle \nabla \mathcal{L}(\mathbf{w}^{t-1}), \mathbf{w}^{t-1} - \mathbf{w}^* \rangle + \underbrace{r_{t-1}^2 \|\nabla \mathcal{L}(\mathbf{w}^{t-1})\|^2}_{\text{A2}} \end{aligned} \quad (3.11)$$

For part **A1** under the Assumption 3.4.2, Lemma 3.4.5 and Maclaurin inequality, we have

$$\begin{aligned}
\mathbf{A1} &= -2r_{t-1} \sum_{i=1}^N p_i^{t-1} \langle \nabla \ell(\mathbf{w}_i^{t-1}), \mathbf{w}^{t-1} - \mathbf{w}^* \rangle \\
&= -2r_{t-1} \sum_{i=1}^N p_i^{t-1} \left(\langle \nabla \ell(\mathbf{w}_i^{t-1}), \mathbf{w}^{t-1} - \mathbf{w}_i^{t-1} \rangle \right) \\
&\quad - 2r_{t-1} \sum_{i=1}^N p_i^{t-1} \left(\langle \nabla \ell(\mathbf{w}_i^{t-1}), \mathbf{w}_i^{t-1} - \mathbf{w}^* \rangle \right) \\
&\leq \sum_{i=1}^N p_i^{t-1} \left(r_{t-1}^2 \|\nabla \ell(\mathbf{w}_i^{t-1})\|^2 + \|\mathbf{w}^{t-1} - \mathbf{w}_i^{t-1}\|^2 \right) \\
&\quad - 2r_{t-1} \sum_{i=1}^N p_i^{t-1} \left(\frac{1}{L + \mu} \|\nabla \ell(\mathbf{w}_i^{t-1})\|^2 + \frac{L\mu}{L + \mu} \|\mathbf{w}_i^{t-1} - \mathbf{w}^*\|^2 \right) \\
&= \left(r_{t-1}^2 - \frac{1}{L + \mu} \right) \sum_{i=1}^N p_i^{t-1} \left(\|\nabla \ell(\mathbf{w}_i^{t-1})\|^2 \right) + \sum_{i=1}^N p_i^{t-1} \|\mathbf{w}^{t-1} - \mathbf{w}_i^{t-1}\|^2 \\
&\quad - \frac{2r_{t-1}L\mu}{L + \mu} \|\mathbf{w}^{t-1} - \mathbf{w}^*\|^2
\end{aligned}$$

From Assumption 3.4.1 and Jensen inequality, we can derive:

$$\|\nabla \ell(\mathbf{w}_i^{t-1}) - \nabla \ell(\mathbf{w}^*)\|^2 \leq L^2 \|\mathbf{w}_i^{t-1} - \mathbf{w}^*\|^2 \quad (3.12)$$

Hence for **A1**, by Jensen inequality and Equation 3.12, we have

$$\begin{aligned}
\mathbf{A1} &\leq \left(r_{t-1}^2 - \frac{1}{L + \mu} \right) \sum_{i=1}^N p_i^{t-1} \left(\|\nabla \ell(\mathbf{w}_i^{t-1})\|^2 \right) + \sum_{i=1}^N p_i^{t-1} \|\mathbf{w}^{t-1} - \mathbf{w}_i^{t-1}\|^2 \\
&\quad - \frac{2r_{t-1}L\mu}{L + \mu} \|\mathbf{w}^{t-1} - \mathbf{w}^*\|^2 \\
&\leq \left(r_{t-1}^2 - \frac{1}{L + \mu} \right) \sum_{i=1}^N p_i^{t-1} \|\mathbf{w}_i^{t-1} - \mathbf{w}^*\|^2 + \sum_{i=1}^N p_i^{t-1} \|\mathbf{w}^{t-1} - \mathbf{w}_i^{t-1}\|^2 \\
&\quad - \frac{2r_{t-1}L\mu}{L + \mu} \|\mathbf{w}^{t-1} - \mathbf{w}^*\|^2 \\
&\leq \left(r_{t-1}^2 - \frac{2r_{t-1}L\mu + 1}{L + \mu} \right) \|\mathbf{w}^{t-1} - \mathbf{w}^*\|^2 + \sum_{i=1}^N p_i^{t-1} \|\mathbf{w}^{t-1} - \mathbf{w}_i^{t-1}\|^2
\end{aligned}$$

Similar for **A2**, we have

$$\begin{aligned}
\mathbf{A2} &= r_{t-1}^2 \left\| \sum_{i=1}^N p_i^{t-1} \nabla \ell(\mathbf{w}_i^{t-1}) \right\|^2 \leq r_{t-1}^2 \sum_{i=1}^N p_i^{t-1} \|\nabla \ell(\mathbf{w}_i^{t-1})\|^2 \\
&\leq r_{t-1}^2 L^2 \sum_{i=1}^N p_i^{t-1} \|\mathbf{w}_i^{t-1} - \mathbf{w}^*\|^2 = r_{t-1}^2 L^2 \|\mathbf{w}^{t-1} - \mathbf{w}^*\|^2
\end{aligned}$$

Then we combine results of **A1** and **A2** for Equation 3.11, it follows that

$$\begin{aligned} \|\mathbf{w}^{t-1} - r_{t-1} \nabla \mathcal{L}(\mathbf{w}^{t-1}) - \mathbf{w}^*\|^2 &\leq \left(r_{t-1}^2 (1 + L^2) - \frac{2r_{t-1}L\mu + 1}{L + \mu} \right) \|\mathbf{w}^{t-1} - \mathbf{w}^*\|^2 \\ &\quad + \sum_{i=1}^N p_i^{t-1} \|\mathbf{w}^{t-1} - \mathbf{w}_i^{t-1}\|^2 \end{aligned} \quad (3.13)$$

□

Proof of Lemma 3.4.7

Proof. For each E step FL necessitates communication. As a result, for any $t - 1 \geq 0$, $\exists t^* \leq t - 1$ that $t - t^* \leq E$, $t^* \in T$, accordingly $\forall i, j \in \mathcal{S}^*$, $\mathbf{w}_i^{t^*} = \mathbf{w}_j^{t^*} = \mathbf{w}^{t^*}$. Then, based on $\mathbb{E} \|\mathbf{X} - \mathbb{E} \mathbf{X}\|^2 \leq \mathbb{E} \|\mathbf{X}\|^2$, Jensen inequality, **A2** in Proof of Lemma 3.4.6 and Assumption 3.4.3, we have

$$\begin{aligned} \mathbb{E} \sum_{i=1}^N p_i^{t-1} \|\mathbf{w}^{t-1} - \mathbf{w}_i^{t-1}\|^2 &= \mathbb{E}_{\mathcal{S}^{t^*}} \sum_{i \in \mathcal{S}^{t^*}} p_i^{t-1} \left\| (\mathbf{w}_i^{t-1} - \mathbf{w}^{t^*}) - (\mathbf{w}^{t-1} - \mathbf{w}^{t^*}) \right\|^2 \\ &= \mathbb{E}_{\mathcal{S}^{t^*}} \left[\mathbb{E}_{\mathcal{S}^{t^*}} \left\| (\mathbf{w}_i^{t-1} - \mathbf{w}^{t^*}) - \mathbb{E}_{\mathcal{S}^{t^*}} [\mathbf{w}_i^{t-1} - \mathbf{w}^{t^*}] \right\|^2 \right] \\ &\leq \mathbb{E}_{\mathcal{S}^{t^*}} \left[\mathbb{E}_{\mathcal{S}^{t^*}} \left\| (\mathbf{w}_i^{t-1} - \mathbf{w}^{t^*}) \right\|^2 \right] = \mathbb{E}_{\mathcal{S}^{t^*}} \sum_{i \in \mathcal{S}^{t^*}} p_i^{t-1} \|\mathbf{w}_i^{t-1} - \mathbf{w}^{t^*}\|^2 \\ &= \mathbb{E}_{\mathcal{S}^{t^*}} \sum_{i \in \mathcal{S}^{t^*}} p_i^{t-1} \left\| \sum_{t=t^*}^{t-2} \nabla \ell(\mathbf{w}_i^{t-1}, D_i^{t-1}) \right\|^2 \\ &\leq \sum_{i \in \mathcal{S}^{t^*}} p_i^{t-1} \mathbb{E}_{\mathcal{S}^{t^*}} (t - 1 - t^*) \sum_{t=t^*}^{t-2} r_{t-1}^2 \|\nabla \ell(\mathbf{w}_i^{t-1}, D_i^{t-1})\|^2 \\ &\leq \sum_{i \in \mathcal{S}^{t^*}} p_i^{t-1} (E - 1) \sum_{t=t^*}^{t-2} r_{t-1}^2 \|\nabla \ell(\mathbf{w}_i^{t-1}, D_i^{t-1})\|^2 \\ &= (E - 1) \sum_{t=t^*}^{t-2} r_{t-1}^2 \sum_{i \in \mathcal{S}^{t^*}} p_i^{t-1} \|\nabla \ell(\mathbf{w}_i^{t-1}, D_i^{t-1})\|^2 \\ &\leq (E - 1)^2 r_{t-1}^2 \left(\rho^2 \left\| \sum_{i \in \mathcal{S}^{t^*}} p_i^{t-1} \nabla \ell(\mathbf{w}_i^{t-1}, D_i^{t-1}) \right\|^2 + \mathcal{Q}^2 \right) \\ &\leq (E - 1)^2 r_{t-1}^2 \left(\rho^2 L^2 \|\mathbf{w}^{t-1} - \mathbf{w}^*\|^2 + \mathcal{Q}^2 \right) \end{aligned} \quad (3.14)$$

□

Proof of Lemma 3.4.8

Proof. Due to Assumption 3.4.4 and Algorithm 3, we have

$$\begin{aligned}
\mathbb{E} \left\| \mathcal{F}(\mathbf{w}^{t-1}) - \nabla \mathcal{L}(\mathbf{w}^{t-1}) \right\|^2 &= \text{Var}(\mathcal{F}(\mathbf{w}^{t-1})) = \mathbb{E}_{\mathcal{S}^{t-1}} \left\| \sum_{i \in \mathcal{S}^{t-1}} p_i^{t-1} (\nabla \ell(\mathbf{w}_i^{t-1}; \mathcal{D}_i^{t-1}) - \nabla \ell(\mathbf{w}_i^{t-1})) \right\|^2 \\
&= \sum_{i \in \mathcal{S}^{t-1}} (p_i^{t-1})^2 \mathbb{E} \left\| \nabla \ell(\mathbf{w}_i^{t-1}; \mathcal{D}_i^{t-1}) - \nabla \ell(\mathbf{w}_i^{t-1}) \right\|^2 \\
&\leq \sum_{i \in \mathcal{S}^{t-1}} (p_i^{t-1})^2 \mathcal{V}_w \leq \mathcal{V}_w \sum_{i \in \mathcal{S}^{t-1}} \left(\frac{v_i^{t-1}}{\sum_{i \in \mathcal{S}^{t-1}} v_i^{t-1}} \right)^2 \\
&\leq \mathcal{V}_w \frac{\sum_{i \in \mathcal{S}^{t-1}} (v_i^{t-1})^2}{\left(\sum_{i \in \mathcal{S}^{t-1}} v_i^{t-1} \right)^2} \leq \mathcal{V}_w \frac{\sum_{i \in \mathcal{S}^{t-1}} (v_i^{t-1})^2}{\sum_{i \in \mathcal{S}^{t-1}} v_i^{t-1}} \\
&\leq \mathcal{V}_w \sum_{i \in \mathcal{S}^{t-1}} v_i^{t-1} \leq (1 - 2\theta) qN \mathcal{V}_w \sqrt{B}
\end{aligned} \tag{3.15}$$

□

Secure Convergence Theorem

Under these four mild and standard assumptions, along with the support of Lemmas, we have:

Theorem 3.4.9. (Secure Convergence Without Attack) Under Assumptions 3.4.1, 3.4.2, 3.4.3 and 3.4.4, and $m = 0$. Choose $\alpha = \frac{L+\mu}{\mu L}$, then FEDQV satisfies

$$\mathbb{E} \mathcal{L}(\mathbf{w}^T) - \mathcal{L}(\mathbf{w}^*) \leq \frac{L}{2\varphi + T} \left(\varphi \mathbb{E} \left\| \mathbf{w}^0 - \mathbf{w}^* \right\|^2 + \frac{\alpha^2}{2} \Delta \right) \tag{3.16}$$

Where

$$\Delta = (E - 1)^2 \mathcal{Q}^2 + (1 - 2\theta) C \mathcal{V}_w \sqrt{B}, \quad \varphi = \alpha (1 + L + LE\rho)$$

Proof of Theorem 3.4.9

Proof. In t round, due to $m = 0$, we have:

$$\begin{aligned}
\left\| \mathbf{w}^t - \mathbf{w}^* \right\|^2 &= \left\| \mathbf{w}^{t-1} - r_{t-1} \mathcal{M}(\mathbf{w}^{t-1}) - \mathbf{w}^* \right\|^2 = \left\| \mathbf{w}^{t-1} - r_{t-1} \mathcal{F}(\mathbf{w}^{t-1}) - \mathbf{w}^* \right\|^2 \\
&= \underbrace{\left\| \mathbf{w}^{t-1} - r_{t-1} \nabla \mathcal{L}(\mathbf{w}^{t-1}) - \mathbf{w}^* \right\|^2}_{\text{A}} + \underbrace{r_{t-1}^2 \left\| \mathcal{F}(\mathbf{w}^{t-1}) - \nabla \mathcal{L}(\mathbf{w}^{t-1}) \right\|^2}_{\text{B}} \\
&\quad + \underbrace{2r_{t-1} \left\langle \mathbf{w}^{t-1} - r_{t-1} \nabla \mathcal{L}(\mathbf{w}^{t-1}) - \mathbf{w}^*, \mathcal{F}(\mathbf{w}^{t-1}) - \nabla \mathcal{L}(\mathbf{w}^{t-1}) \right\rangle}_{\text{C}}
\end{aligned} \tag{3.17}$$

Where

$$\mathcal{M}(\mathbf{w}^{t-1}) = \sum_{i \in \mathcal{S}^{t-1}} p_i^{t-1} \mathcal{M}_i(\mathbf{w}_i^{t-1})$$

Note that $\mathbb{E} \mathbf{C} = 0$. For the expectation of $\mathbf{A}$, from Lemma 3.4.6 and Lemma 3.4.7, it follows that

$$\begin{aligned} \mathbb{E}[\mathbf{A}] &= \mathbb{E} \left\| \mathbf{w}^{t-1} - r_{t-1} \nabla \mathcal{L}(\mathbf{w}^{t-1}) - \mathbf{w}^* \right\|^2 \\ &\leq \left(r_{t-1}^2 (1 + L^2) - \frac{2r_{t-1}L\mu + 1}{L + \mu} \right) \left\| \mathbf{w}^{t-1} - \mathbf{w}^* \right\|^2 \\ &\quad + (E - 1)^2 r_{t-1}^2 \left(\rho^2 L^2 \left\| \mathbf{w}^{t-1} - \mathbf{w}^* \right\|^2 + \mathcal{Q}^2 \right) \end{aligned} \quad (3.18)$$

We use Lemma 3.4.8 to bound $\mathbf{B}$, we have

$$\mathbb{E}[\mathbf{B}] \leq r_{t-1}^2 (1 - 2\theta) qN\mathcal{V}_w \sqrt{B} \quad (3.19)$$

Hence, we have

$$\mathbb{E} \left\| \mathbf{w}^t - \mathbf{w}^* \right\|^2 \leq \left(r_{t-1}^2 (1 + L^2) - \frac{2r_{t-1}L\mu + 1}{L + \mu} + (E - 1)^2 r_{t-1}^2 \rho^2 L^2 \right) \mathbb{E} \left\| \mathbf{w}^{t-1} - \mathbf{w}^* \right\|^2 + r_{t-1}^2 \Delta \quad (3.20)$$

where

$$\Delta = (E - 1)^2 \mathcal{Q}^2 + (1 - 2\theta) qN\mathcal{V}_w \sqrt{B}$$

For the learning rate r_t , $\exists \alpha \geq \frac{L+\mu}{2\mu L}$, $\exists \beta > 0$, such that $r_t = \frac{\alpha}{\beta+t} \leq \frac{1}{1+L+LE\rho}$. We use mathematical induction to prove the following statement:

Proposition: $\forall t \in \mathbb{N}$, $\mathbb{E} \left\| \mathbf{w}^t - \mathbf{w}^* \right\|^2 \leq \frac{\gamma}{\beta+t}$, where $\gamma = \max \left\{ \frac{(L+\mu)\alpha^2\Delta}{2L\mu\alpha-L-\mu}, \beta \mathbb{E} \left\| \mathbf{w}^0 - \mathbf{w}^* \right\|^2 \right\}$.

Let $P(t)$ be the statement $\mathbb{E} \left\| \mathbf{w}^t - \mathbf{w}^* \right\|^2 \leq \frac{\gamma}{\beta+t}$, we give a proof by induction on t .

Base case: The statement $P(0)$ holds for $t = 0$:

$$\mathbb{E} \left\| \mathbf{w}^0 - \mathbf{w}^* \right\|^2 \leq \frac{\gamma}{\beta}$$

Inductive step: Assume the induction hypothesis that for a particular j , the single case $t = j$ holds, meaning $P(j)$ is true:

$$\mathbb{E} \left\| \mathbf{w}^j - \mathbf{w}^* \right\|^2 \leq \frac{\gamma}{\beta + j}$$

It follows that:

$$\begin{aligned} \mathbb{E} \left\| \mathbf{w}^{j+1} - \mathbf{w}^* \right\|^2 &\leq \left(r_j^2 (1 + L^2) - \frac{2r_jL\mu + 1}{L + \mu} + (E - 1)^2 r_j^2 \rho^2 L^2 \right) \mathbb{E} \left\| \mathbf{w}^j - \mathbf{w}^* \right\|^2 + r_j^2 \Delta \\ &\leq \left(r_j^2 (1 + L^2 + (E - 1)^2 \rho^2 L^2) - \frac{2L\mu\alpha}{(L + \mu)(\beta + j)} \right) \frac{\gamma}{\beta + j} + \left(\frac{\alpha}{\beta + j} \right)^2 \Delta \\ &\leq \left[\frac{\alpha^2\Delta}{(\beta + j)^2} - \frac{2L\mu\alpha - L - \mu}{(\beta + j)^2(L + 1)} \gamma \right] + \frac{\beta + j - 1}{(\beta + j)^2} \gamma \\ &\leq \frac{\gamma}{\beta + j + 1} \end{aligned}$$

Therefore, the statement $P(j + 1)$ also holds true, establishing the inductive step. Since both the base case and the inductive step have been proved as true, by mathematical induction the statement $P(t)$ holds for $\forall t \in \mathbb{N}$.

We choose $\alpha = \frac{L+\mu}{\mu L}$, and $\beta = 2 \frac{(1+L+LE\rho)(L+\mu)}{\mu L}$, and we have

$$\begin{aligned} \gamma &= \max \left\{ \frac{(L+\mu)\alpha^2\Delta}{2L\mu\alpha - L - \mu}, \beta \mathbb{E} \|\mathbf{w}^0 - \mathbf{w}^*\|^2 \right\} \\ &\leq \frac{(L+\mu)\alpha^2\Delta}{2\alpha\mu L - L - \mu} + \beta \mathbb{E} \|\mathbf{w}^0 - \mathbf{w}^*\|^2 \\ &= \alpha^2\Delta + 2(1 + L + LE\rho)\alpha \mathbb{E} \|\mathbf{w}^0 - \mathbf{w}^*\|^2 \end{aligned}$$

Then based on Assumption 3.4.1 and Taylor expansion, we have the quadratic upper-bound of $\mathcal{L}(\cdot)$:

$$\mathcal{L}(\mathbf{w}_1) - \mathcal{L}(\mathbf{w}_2) \leq (\mathbf{w}_1 - \mathbf{w}_2)^T \nabla \mathcal{L}(\mathbf{w}_2) + \frac{L}{2} \|\mathbf{w}_1 - \mathbf{w}_2\|^2$$

It follows that

$$\begin{aligned} \mathbb{E} \mathcal{L}(\mathbf{w}^T) - \mathcal{L}(\mathbf{w}^*) &\leq \frac{L}{2} \mathbb{E} \|\mathbf{w}^T - \mathbf{w}^*\|^2 \leq \frac{\gamma L}{2(\beta + T)} \\ &\leq \frac{L}{2\alpha(1 + L + LE\rho) + T} \left(\frac{\alpha^2}{2} \Delta + \alpha(1 + L + LE\rho) \mathbb{E} \|\mathbf{w}^0 - \mathbf{w}^*\|^2 \right) \\ &= \frac{L}{2\varphi + T} \left(\varphi \mathbb{E} \|\mathbf{w}^0 - \mathbf{w}^*\|^2 + \frac{\alpha^2}{2} \Delta \right) \end{aligned}$$

Where

$$\Delta = (E - 1)^2 \mathcal{Q}^2 + (1 - 2\theta) C \mathcal{V}_w \sqrt{B}, \quad \varphi = \alpha(1 + L + LE\rho)$$

□

Remark 3.4.10. According to Theorem 3.4.9, FEDQV obtains a secure convergence rate of $\mathcal{O}(\frac{1}{T})$ in the absence of malicious parties, which is comparable to the convergence rate of FEDAVG [160].

Robust Convergence Theorem

Suppose the percentage of attackers in the whole parties is m , we denote

$$\mathcal{M}_i(\mathbf{w}_i^t) = \begin{cases} * & \text{if } i \in \text{malicious parties} \\ \nabla \ell(\mathbf{w}_i^t; \mathcal{D}_i^t) & \text{if } i \in \text{honest parties} \end{cases}$$

Where $*$ stands for an arbitrary value from the malicious parties.

Assumption 3.4.11. The expected square norm of gradients $\mathbf{w}$ is bounded:

$$\forall \mathbf{w} \in \mathbb{R}^d, \exists \mathcal{G}_{\mathbf{w}} < \infty, \mathbb{E} \|\nabla \ell(\mathbf{w}; \mathcal{D})\|_2^2 \leq \mathcal{G}_{\mathbf{w}}^2$$

With one more assumption above, we have:

Theorem 3.4.12. (Robust Convergence Under Attack) Under Assumptions 3.4.1, 3.4.2, 3.4.3 3.4.4 and 3.4.11, Choose $\alpha = \frac{L+\mu}{\mu L}$, then FedQV satisfies

$$\mathbb{E} \mathcal{L}(\mathbf{w}^T) - \mathcal{L}(\mathbf{w}^*) \leq \frac{L + 2Lr_{T-1}\varpi}{2\varphi + T} \left(\varphi \mathbb{E} \|\mathbf{w}^0 - \mathbf{w}^*\|^2 + \frac{\alpha^2}{2} \Delta \right) + \frac{L\varpi^2}{2} \quad (3.21)$$

Where

$$\Delta = (E - 1)^2 \mathcal{Q}^2 + (1 - 2\theta) C \mathcal{V}_{\mathbf{w}} \sqrt{B}, \quad \varphi = \alpha (L + 1), \quad \varpi = mN \mathcal{G}_{\mathbf{w}} r_{T-1} \sqrt{4 + 6\theta - \theta^2}$$

Proof of Theorem 3.4.12

Proof. In the t round, we have:

$$\begin{aligned} \|\mathbf{w}^t - \mathbf{w}^*\|^2 &= \|\mathbf{w}^{t-1} - r_{t-1} \mathcal{M}(\mathbf{w}^{t-1}) - \mathbf{w}^*\|^2 \\ &= \|\mathbf{w}^{t-1} - r_{t-1} \mathcal{F}(\mathbf{w}^{t-1}) - \mathbf{w}^* + r_{t-1} \mathcal{F}(\mathbf{w}^{t-1}) - r_{t-1} \mathcal{M}(\mathbf{w}^{t-1})\|^2 \\ &= \underbrace{\|\mathbf{w}^{t-1} - r_{t-1} \mathcal{F}(\mathbf{w}^{t-1}) - \mathbf{w}^*\|^2}_{\mathbf{A}} + \underbrace{r_{t-1}^2 \|\mathcal{F}(\mathbf{w}^{t-1}) - \mathcal{M}(\mathbf{w}^{t-1})\|^2}_{\mathbf{B}} \\ &\quad + \underbrace{2r_{t-1} \langle \mathbf{w}^{t-1} - r_{t-1} \mathcal{F}(\mathbf{w}^{t-1}) - \mathbf{w}^*, \mathcal{F}(\mathbf{w}^{t-1}) - \mathcal{M}(\mathbf{w}^{t-1}) \rangle}_{\mathbf{C}} \end{aligned} \quad (3.22)$$

Where

$$\mathcal{M}(\mathbf{w}^{t-1}) = \sum_{i \in \mathcal{S}^{t-1}} p_i^{t-1} \mathcal{M}_i(\mathbf{w}_i^{t-1})$$

For the expectation of $\mathbf{A}$, from Theorem 3.4.9, it follows that

$$\mathbb{E}[\mathbf{A}] \leq \frac{1}{2\varphi + t} \left(2\varphi \mathbb{E} \|\mathbf{w}^0 - \mathbf{w}^*\|^2 + \alpha^2 \Delta \right) \quad (3.23)$$

Where

$$\Delta = (E - 1)^2 \mathcal{Q}^2 + (1 - 2\theta) C \mathcal{V}_{\mathbf{w}} \sqrt{B}, \quad \varphi = \alpha (1 + L + LE\rho)$$

For $\mathbf{B}$, we have

$$\begin{aligned} \mathbb{E}[\mathbf{B}] &= r_{t-1}^2 \left\| \sum_{i \in \mathcal{S}^{t-1}} p_i^{t-1} \nabla \ell(\mathbf{w}_i^{t-1}) - \sum_{i \in \mathcal{S}^{t-1}} p_i^{t-1} \mathcal{M}_i(\mathbf{w}_i^{t-1}) \right\|^2 \\ &= r_{t-1}^2 \left\| \sum_{i \in \mathcal{S}^{t-1}} p_i^{t-1} (\nabla \ell(\mathbf{w}_i^{t-1}) - \mathcal{M}_i(\mathbf{w}_i^{t-1})) \right\|^2 \\ &\leq r_{t-1}^2 \left\| \sum_{i \in mN} p_i^{t-1} (\nabla \ell(\mathbf{w}_i^{t-1}) - \mathcal{M}_i(\mathbf{w}_i^{t-1})) \right\|^2 \end{aligned} \quad (3.24)$$

Where m is the percentage of the malicious parties.

Due to Equation 3.3, we have

$$\theta \leq \frac{\langle \nabla \ell(\mathbf{w}_i^{t-1}), \mathcal{M}_i(\mathbf{w}_i^{t-1}) \rangle}{\|\nabla \ell(\mathbf{w}_i^{t-1})\| \cdot \|\mathcal{M}_i(\mathbf{w}_i^{t-1})\|} \leq 1 - \theta \quad (3.25)$$

Thus,

$$\theta \|\nabla \ell(\mathbf{w}_i^{t-1})\| \|\mathcal{M}_i(\mathbf{w}_i^{t-1})\| \leq \langle \nabla \ell(\mathbf{w}_i^{t-1}), \mathcal{M}_i(\mathbf{w}_i^{t-1}) \rangle \leq (1 - \theta) \|\nabla \ell(\mathbf{w}_i^{t-1})\| \|\mathcal{M}_i(\mathbf{w}_i^{t-1})\| \quad (3.26)$$

Due to this, we have

$$\begin{aligned} & \|\nabla \ell(\mathbf{w}_i^{t-1})\|^2 - 2(1 - \theta) \|\nabla \ell(\mathbf{w}_i^{t-1})\| \|\mathcal{M}_i(\mathbf{w}_i^{t-1})\| + \|\mathcal{M}_i(\mathbf{w}_i^{t-1})\|^2 \\ & \leq \|\nabla \ell(\mathbf{w}_i^{t-1}) - \mathcal{M}_i(\mathbf{w}_i^{t-1})\|^2 \\ & \leq \|\nabla \ell(\mathbf{w}_i^{t-1})\|^2 - 2\theta \|\nabla \ell(\mathbf{w}_i^{t-1})\| \|\mathcal{M}_i(\mathbf{w}_i^{t-1})\| + \|\mathcal{M}_i(\mathbf{w}_i^{t-1})\|^2 \end{aligned} \quad (3.27)$$

Hence we have

$$\begin{aligned} & \theta(2 - \theta) \|\nabla \ell(\mathbf{w}_i^{t-1})\|^2 + \|(1 - \theta) \|\nabla \ell(\mathbf{w}_i^{t-1})\| - \|\mathcal{M}_i(\mathbf{w}_i^{t-1})\|\|^2 \\ & \leq \|\nabla \ell(\mathbf{w}_i^{t-1}) - \mathcal{M}_i(\mathbf{w}_i^{t-1})\|^2 \\ & \leq (1 - \theta^2) \|\nabla \ell(\mathbf{w}_i^{t-1})\|^2 + \|\theta \|\nabla \ell(\mathbf{w}_i^{t-1})\| - \|\mathcal{M}_i(\mathbf{w}_i^{t-1})\|\|^2 \end{aligned} \quad (3.28)$$

Hence,

$$\theta(2 - \theta) \|\nabla \ell(\mathbf{w}_i^{t-1})\|^2 \leq \|\nabla \ell(\mathbf{w}_i^{t-1}) - \mathcal{M}_i(\mathbf{w}_i^{t-1})\|^2 \quad (3.29)$$

Due to the Triangle Inequality, we have

$$\sqrt{\theta(2 - \theta)} \|\nabla \ell(\mathbf{w}_i^{t-1})\| \leq \|\nabla \ell(\mathbf{w}_i^{t-1}) - \mathcal{M}_i(\mathbf{w}_i^{t-1})\| \leq \|\nabla \ell(\mathbf{w}_i^{t-1})\| + \|\mathcal{M}_i(\mathbf{w}_i^{t-1})\| \quad (3.30)$$

It follows that:

$$(\sqrt{\theta(2 - \theta)} - 1) \|\nabla \ell(\mathbf{w}_i^{t-1})\| \leq \|\mathcal{M}_i(\mathbf{w}_i^{t-1})\| \quad (3.31)$$

By incorporating Equation 3.28 and leveraging the AM-GM inequality, we can derive the following expression

$$\begin{aligned} \|\nabla \ell(\mathbf{w}_i^{t-1}) - \mathcal{M}_i(\mathbf{w}_i^{t-1})\|^2 & \leq (1 - \theta^2) \|\nabla \ell(\mathbf{w}_i^{t-1})\|^2 + \|\theta \|\nabla \ell(\mathbf{w}_i^{t-1})\| - \|\mathcal{M}_i(\mathbf{w}_i^{t-1})\|\|^2 \\ & \leq \left(1 - \theta^2 + \left(1 + \theta + \sqrt{\theta(2 - \theta)}\right)^2\right) \|\nabla \ell(\mathbf{w}_i^{t-1})\|^2 \\ & \leq (4 + 6\theta - \theta^2) \|\nabla \ell(\mathbf{w}_i^{t-1})\|^2 \end{aligned} \quad (3.32)$$

Therefore,

$$\mathbb{E}[\mathbf{B}] \leq r_{t-1}^2 \left\| \sum_{i \in mN} p_i^{t-1} \left(\sqrt{4 + 6\theta - \theta^2} \|\nabla \ell(\mathbf{w}_i^{t-1})\| \right) \right\|^2 \leq (4 + 6\theta - \theta^2) m^2 N^2 r_{t-1}^2 \mathcal{G}_{\mathbf{w}}^2 \quad (3.33)$$

Hence for $\mathbf{C}$, we have

$$\mathbb{E}[\mathbf{C}] \leq \frac{2mN\mathcal{G}_w r_{t-1}^2 \sqrt{4 + 6\theta - \theta^2}}{2\varphi + t} \left(2\varphi \mathbb{E} \|\mathbf{w}^0 - \mathbf{w}^*\|^2 + \alpha^2 \Delta \right) \quad (3.34)$$

Then based on Assumption 3.4.1 and Taylor expansion, we have the quadratic upper-bound of $\mathcal{L}(\cdot)$:

$$\mathcal{L}(\mathbf{w}_1) - \mathcal{L}(\mathbf{w}_2) \leq (\mathbf{w}_1 - \mathbf{w}_2)^T \nabla \mathcal{L}(\mathbf{w}_2) + \frac{L}{2} \|\mathbf{w}_1 - \mathbf{w}_2\|^2$$

It follows that

$$\mathbb{E} \mathcal{L}(\mathbf{w}^T) - \mathcal{L}(\mathbf{w}^*) \leq \frac{L}{2} \mathbb{E} \|\mathbf{w}^T - \mathbf{w}^*\|^2 \leq \frac{L + 2Lr_{T-1}\varpi}{2\varphi + T} \left(\varphi \mathbb{E} \|\mathbf{w}^0 - \mathbf{w}^*\|^2 + \frac{\alpha^2}{2} \Delta \right) + \frac{L\varpi^2}{2}$$

Where

$$\Delta = (E - 1)^2 \mathcal{Q}^2 + (1 - 2\theta) C \mathcal{V}_w \sqrt{B}, \quad \varphi = \alpha(1 + L + LE\rho), \quad \varpi = mN\mathcal{G}_w r_{T-1} \sqrt{4 + 6\theta - \theta^2}$$

□

Remark 3.4.13. According to Theorem 3.4.12, FEDQV converge to a near-optimal solution in the presence of malicious parties, with the resulting performance gap determined by the percentage of malicious parties.

Remark 3.4.14. The error rate exhibits dependence on the budget B , the similarity threshold θ , and the percentage of malicious parties m . It is noteworthy that a larger budget allocation, a reduction in the similarity threshold, or an augmentation in the proportion of malicious parties induce more pronounced disparities in model updates, consequently resulting in an elevated error rate.

This theoretical error bound informs the design of our experiments. In particular, it motivates several key practical considerations:

- **Impact of Attacker Proportion (m):** An increase in the proportion of attackers m theoretically leads to a higher error rate. Our experimental results consistently validate this prediction across various datasets and scenarios. As shown in Figure 3.4 and Tables 3.2, 3.4, a higher proportion of attackers consistently results in a measurable decrease in model accuracy.
- **Impact of Hyperparameters (B, θ):** The hyperparameters B and θ influence the error rate as predicted by our theoretical analysis. Specifically, a larger budget B is associated with a higher error rate, while a larger similarity threshold θ tends to reduce the error rate. Systematic experiments (see Figure 3.6 in Section 3.5.11) corroborate these theoretical predictions, demonstrating that careful tuning of B and θ significantly impacts system performance.

3.4.2. Truthfulness

The FEDQV mechanism belongs to a single-parameter domain, where the real parameter v_i plays a direct role in determining the eligibility of party i for aggregation. The FEDQV mechanism is normalised on a single parameter domain according to game theory principles [154], where for every v_i and v_{-i} such that $f(v_i, v_{-i}) \notin W_i$, $p_i(v_i, v_{-i}) = 0$. Here, v_{-i} represents votes cast by all parties except for i , W_i is the subset of participating parties in FEDQV aggregation, f represents the voting scheme, and p_i is the payment function with $p_i(v_i, v_{-i}) = v_i^2$ in FEDQV. The upcoming definition of truthfulness and the following lemmas contribute to the proof of Theorem 3.4.19 in alignment with monotone and critical value concepts in game theory [154].

Definition 3.4.15. A mechanism $(f, p_1, \dots, p_n)$ is called truthfulness if for every party i , we denote $a = f(v_i, v_{-i})$ and $a' = f(v'_i, v_{-i})$ as the outcome of the voting, then $v_i(a) - p_i(v_i, v_{-i}) \geq v_i(a') - p_i(v'_i, v_{-i})$, where $v_i(a)$ denotes the gain of party i if the outcome of the voting is a .

Here, $v_i(a) - p_i(v_i, v_{-i})$ is the utility of party i , indicating the gain from voting ($v_i(a)$) minus its cost ($p_i(v_i, v_{-i})$). Intuitively this implies that party i with v_i would prefer “telling the truth” v_i to the server rather than any possible “lie” v'_i since this gives him higher (in the weak sense) utility.

Lemma 3.4.16. f is monotone: $\forall v_{-i}$ and $\forall v'_i > v_i$, if $f(v_i, v_{-i}) \in W_i$, then $f(v'_i, v_{-i}) \in W_i$.

Lemma 3.4.17. In FEDQV, $\forall i, v_i, v_{-i}$ that $f(v_i, v_{-i}) \in W_i$, we have that $p_i(v_i, v_{-i}) = \Phi_i(v_{-i})$, where Φ_i is the critical value of the monotone function f on the single parameter domain that $\Phi_i(v_{-i}) = \sup_{v_i: f(v_i, v_{-i}) \notin W_i} v_i$.

Proof of Lemmas

Proof of Lemmas 3.4.16

Proof. Suppose the party i who holds the vote v_i , and is included in the aggregation. This implies that for the voting scheme f , $\forall v_{-i}$, $f(v_i, v_{-i}) \in W_i$.

Now consider $\forall v'_i > v_i$. If party i holds vote $v'_i > v_i$, in FEDQV, it will also be included in the aggregation. Therefore, $f(v'_i, v_{-i}) \in W_i$.

Thus, f is monotone. □

For a monotone function f , $\forall v_{-i}$ for party i , f can either include or exclude party i in the aggregation. There exists a particular threshold, termed the *critical value*, such that if the vote of party i is below this threshold, it prevents their participation in the aggregation, whereas if their vote exceeds this threshold, they are included in the aggregation.

Definition 3.4.18. The critical value of a monotone voting choice function f on a single parameter domain is $\Phi_i(v_{-i}) = \sup_{v_i: f(v_i, v_{-i}) \notin W_i} v_i$. The critical value at v_{-i} is undefined if $\{v_i | f(v_i, v_{-i}) \notin W_i\}$.

Proof of Lemmas 3.4.17

Proof. The number of participating parties is $|\mathcal{S}|$ in each round. Assuming there are top k parties in the ranking joining aggregation, whose votes are $v_{i \in k}$ and payments are $c_{i \in k}$. Notice in FEDQV, the payment function $p_i(v_i, v_{-i}) = v_i^2$.

If party i pays $p'_i(v_i, v_{-i}) > \Phi_i(v_{-i})$, it will still remain among top k and join the aggregation, according to function f . Conversely, if party i pays $p'_i(v_i, v_{-i}) < \Phi_i(v_{-i})$, it will be replaced by the party $k + 1$ in the ranking, and party i will not be able to join the aggregation regardless of whether party $k + 1$ joins aggregation or not.

Therefore, in order to participate in the aggregation, parties must pay the critical value, that is, $\forall i, v_i, v_{-i}$ that $f(v_i, v_{-i}) \in W_i$, we have that $p_i(v_i, v_{-i}) = \Phi_i(v_{-i}) = \sup_{v_i: f(v_i, v_{-i}) \notin W_i} v_i$ $\square$

Truthfulness Theorem

Based on Lemma 3.4.16 and Lemma 3.4.17, we establish the following theorem:

Theorem 3.4.19. FEDQV is incentive compatible (truthful).

Proof of Theorem 3.4.19

Proof. According to Theorem 9.36 [154]: a normalised mechanism on a single parameter domain is incentive compatible(truthful) if and only if:

- (i) The voting scheme f is monotone.
- (ii) For every party i participants in the aggregation pays the critical value:

$$\Phi_i(v_{-i}) = \sup_{v_i: f(v_i, v_{-i}) \notin W_i} v_i$$

The conditions (i) and (ii) are proofed in Lemma 3.4.16 and Lemma 3.4.17 respectively. Thus, the proposed scheme FEDQV is incentive-compatible (truthful). $\square$

Remark 3.4.20. Regarding the concept of truthfulness, it theoretically ensures that being honest is the dominant strategy since providing manipulated similarity scores may lead to penalties and removal from the system due to the masked voting rule $\mathcal{H}$ and limited budget B . This is an integral part of the nature of QV embedded within our FEDQV framework.

3.5. Experiments

The objectives of our experimental evaluation are the following: (a) To assess the performance of our aggregation method relative to FEDAVG. (b) To benchmark our method across 10 distinct state-of-the-art poisoning attack scenarios. (c) To validate the alignment of our experimental findings with our prior theoretical analyses. (d) To demonstrate the ease of integration and compatibility of our method within Byzantine-robust FL defence schemes and privacy-preserving mechanisms.

3.5.1. Experimental Setting

Datasets and global models. We implement the typical FL setting where each party owns its local data and transmits/receives information to/from the central server. To demonstrate the generality of our method, we train different global models on different datasets. We use four popular benchmark datasets: MNIST [161], Fashion-MNIST [162], FEMNIST [163] and CIFAR10 [164]. We consider a multi-layer CNN same as in [153], consisting of 2 convolutional layers and 2 fully connected layers for MNIST, Fashion-MNIST and FEMNIST, and the ResNet18 [118] for CIFAR10.

Non-IID setting. In order to fulfil the setting of a heterogeneous and unbalanced dataset for FL, we sample from a Dirichlet distribution with the concentration parameter $\iota = 0.9$ as the Non-IID degree as in [34], [165], with the intention of generating non-IID and unbalanced data partitions. Moreover, we have examined the performance across varying levels of non-IID data in Section 3.5.10.

Parameter Settings. The server selects 10 ($\mathcal{C}$) out of 100 (N) parties to participate in each communication round and trains the global models for 100 communication rounds ($\frac{T}{E}$). The local training process is conducted with 5 epochs per round ($E = 5$), a local batch size of 10 ($b = 10$), and an initial learning rate of 0.01 ($r_0 = 0.01$). We set the model hyper-parameters budget B and the similarity threshold θ to 30 and 0.2 respectively based on the hyper-parameter searching.

3.5.2. Threat Model

We assume the parties are *malicious* while the server is *honest-but-curious*. Within this context, parties may deliberately submit false or manipulated local model updates to the central server. Their objectives could range from injecting biases and compromising the model’s performance to extracting sensitive information from the aggregated global model. The server correctly follows the FL protocol steps but remains curious to discover any private information. In our case, the server has full visibility of all local models and launches privacy attacks upon receiving an update from a target party to reconstruct users’ training data.

Furthermore, the percentage of malicious party m is an important factor in determining the success of the poisoning attack. In our analysis, we assume that the number of malicious parties is less than the number of honest parties, a common setting in such types of analysis and recent works [1], [39], [53].

3.5.3. Evaluated Poisoning Attacks

Our paper addresses three distinct attack schemes:

- **Data poisoning:** Attackers submit the true similarity score based on their poisoned updates, including **Labelflip Attack** [32], **Gaussian Attack** [166], **Backdoor** [167], **Scaling Attack** [34], **Neurotoxin** [35].
- **Model poisoning:** Attackers submit the true similarity score based on their clean updates and poison their model, including: **Krum Attack** [32], **Trim Attack** [32], and Aggregation-agnostic attacks: **Min-Max** and **Min-Sum** [33]
- **QV-tailored Attack:** Attackers submit both poisoned similarity score and the local model to exploit vulnerabilities in FEDQV. This dual-pronged attack, termed **QV-Adaptive**, presents a heightened challenge for FEDQV.

Here are the details of the aforementioned attacks. We begin with data poisoning attacks:

Labelflip Attack [32]: In the Label-Flip scenario, all the labels of the training data for the malicious clients are set to zero. This scenario simulates a directed attack, with the goal to disproportionally bias the jointly trained model towards one specific class. This is a data poisoning attack that does not require knowledge of the training data distribution. Under this attack, the malicious parties train with clean data but with flipped labels. Specifically, we flip a label k as $K - k - 1$, where K is the total class number.

Gaussian Attack [166]: This attack forges local model updates via Gaussian distribution on the malicious parties. malicious parties forge local model updates via Gaussian distribution.

Backdoor Attack [167] Malicious parties inject specific backdoor triggers into the training data and modify their labels to the attacker-chosen target label. Specifically, we use the same backdoor pattern trigger and attacker-chosen target label as in [168] as our trigger and set the attacker-chosen target label as 5. The backdoor can be introduced into a model by an attacker who poisons the training data with specially crafted inputs. A backdoor transformation applied to any input causes the model to mis-classify it to an attacker-chosen label The pattern must be applied by the attacker during local training, by modifying the digital image.

Scaling attack [34] The malicious parties generate poisoned local model updates by backdoor attack and only launch this attack during the last communication round after scaling these updates by a factor of N .

Neurotoxin attack [35] In this attack, the adversary starts by downloading the gradient from the previous round and employs it to approximate the benign gradient for the upcoming round. The attacker identifies the top- $k\%$ coordinates of the benign gradient and treats them as the constraint set. Over several epochs of Projected Gradient Descent (PGD), the attacker computes gradient updates on the manipulated dataset and projects this gradient onto the constraint set, which consists of the bottom- $k\%$ coordinates of the observed benign gradient. PGD is employed to approach the optimal solution within the span of the bottom- $k\%$ coordinates. We adopt the original parameter setting from the paper, where k is set to 0.1.

Next, we move to model poisoning attacks:

Krum Attack [32]: Malicious parties craft poisoned local model updates opposite from benign ones, and enable them to circumvent the defence of Krum [39].

Trim Attack [32] The poisoned local model updates constructed by malicious parties are optimised for evading the Trim-mean and Median [53].

Min-Max Attack [33] In order to ensure that the malicious gradients closely align with the benign gradients within the clique, attackers strategically compute the malicious gradient. This computation is carried out to limit the maximum distance of the malicious gradient from any other gradient, which is constrained by the maximum distance observed between any two benign gradients.

Min-Sum [33] The Min-Sum attack enforces an upper bound on the sum of squared distances between the malicious gradient and all the benign gradients. This upper bound is determined by the sum of squared distances between any one benign gradient and the rest of the benign gradients.

Finally, we introduce the QV-tailored attack:

QV-Adaptive attack Tailored for FEDQV, this attack leverages the Aggregation-agnostic optimisations [33] within the LMP framework [32]. This attack manipulates both the similarity score and the local model, following the procedure below:

1. The malicious party i generates benign updates w_i^t using clean data $\mathcal{D}_i$ in round t and calculates the corresponding similarity score;
2. Malicious parties (with counts of m) collectively normalise all the similarity scores and employ the Aggregation-agnostic Min-Max optimisation to select the optimal similarity score. This optimisation objective aims to increase the likelihood of the score being accepted by the server.
3. The adaptive attack focuses on local model poisoning to optimise the following

problem:

$$\begin{aligned} & \max \nu \\ \text{s.t. } & \mathbf{w}_{i \in m}^{t'} = \text{FedQV}(\mathbf{w}_1^t, \mathbf{w}_2^t, \dots, \mathbf{w}_m^t) \\ & \mathbf{w}_{i \in m}^{t'} = \mathbf{w}_i^t - \nu \hat{\mathbf{d}} \end{aligned}$$

Here, $\hat{\mathbf{d}}$ represents a column vector encompassing the estimated changing directions of all global model parameters. The variables $\mathbf{w}_{i \in m}^t$ and $\mathbf{w}_{i \in m}^{t'}$ correspond to the local model before and after the attack. ν denotes the extent of the attack's impact on the model.

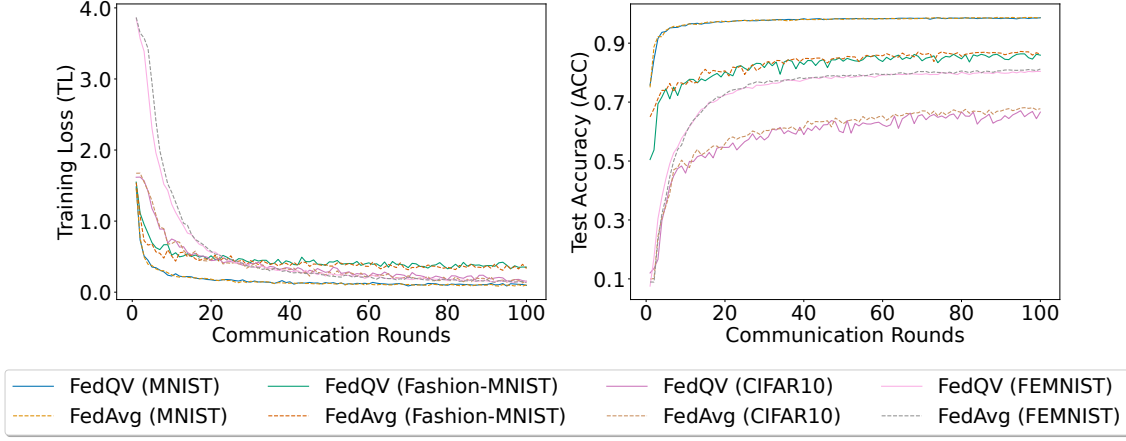
Notice, Labelflip, Gaussian, Krum, Trim, Min-Max, Min-Sum and QV Adaptive attacks are untargeted attacks, whereas, Backdoor, Scaling and Neurotoxin attacks are targeted attacks. We confine our analysis to the worst-case scenario in which the attackers submit the poisoned updates in every round of the training process for all attack strategies with the exception of the Scaling attack.

3.5.4. Performance Metrics

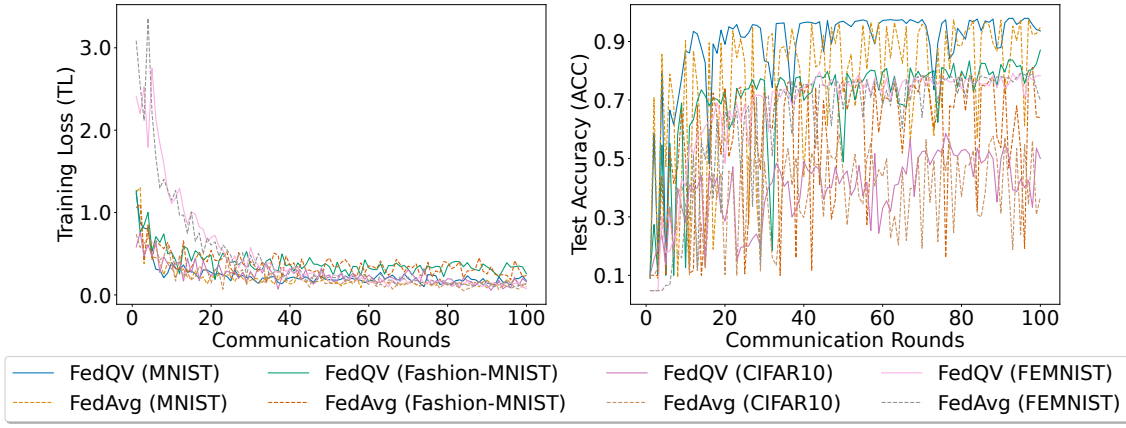
We use the average test accuracy (ACC) of the global model to evaluate the result of the aggregation defence for poisoning attacks, in which attackers aim to mislead the global model during the testing phase. ACC is the percentage of testing examples with the correct predictions by the global model in the whole testing dataset, which is defined as $\text{ACC} = (\# \text{ correct predictions}) / (\# \text{ testing samples})$. In addition, there are targeted attacks that aim to attack a specific label while keeping the accuracy of classification on other labels unaltered. Therefore, besides ACC, we choose the attack success rate (ASR) to measure how many of the samples that are attacked, are classified as the target label chosen by malicious parties. A robust federated aggregation method would obtain a higher Avg-ACC as well as a lower ASR under poisoning attacks. An ideal aggregation method can achieve 100% Avg-ACC and has the ASR as low as the fraction of attacked samples from the target label.

3.5.5. Convergence

We evaluate the convergence of FEDAVG and FEDQV across the four datasets under both benign and adversarial conditions. The training loss (TL) and accuracy (ACC) of global models trained using FEDQV and FEDAVG are depicted in Figure 3.3a without any attack and in Figure 3.3a under labelflip attacks with 40% malicious parties. In the absence of Byzantine attacks, the convergence of the global model trained with FEDQV matches that of FEDAVG across all datasets, consistent with Theorem 3.4.9. However, under attack, FEDAVG struggles to converge due to its susceptibility to poisoning attacks, as shown in Figure 3.3a. While FEDQV also exhibits slower convergence under attack, it eventually



(a) Without attack.



(b) Under labelflip attack with 40% malicious parties.

Figure 3.3: Convergence comparison of FedAvg and FedQV in Training Loss (TL) and Accuracy (ACC) over 100 epochs across four benchmark datasets, both under no attack and under attack scenarios.

converges and outperforms FedAvg. This outcome aligns with Theorem 3.4.12, indicating that FedQV achieves convergence to a near-optimal solution even in the presence of malicious parties, with the extent of performance improvement determined by the percentage of malicious parties.

3.5.6. Defence against Poisoning Attacks

Static percentage of attackers: We present ACC and ASR of global models trained using both FedAvg and FedQV under the 10 aforementioned poisoning attacks across all four datasets in Table 3.1. The experiments involve 30% malicious parties, same as in previous studies such as [1], [39], which is also a common byzantine consensus threshold for resistance to failures in a typical distributed system [116]. In data poisoning attacks, the results consistently demonstrate that FedQV outperforms FedAvg, achieving the highest ACC with the smallest standard error. When considering targeted attacks, FedQV again stands out, displaying the highest ACC along with the lowest ASR when compared

	MNIST		Fashion-MNIST		CIFAR10		FEMNIST	
	FEDAVG	FEDQV	FEDAVG	FEDQV	FEDAVG	FEDQV	FEDAVG	FEDQV
Data Poison								
Labelflip	98.81±0.03	98.54±0.05	86.70±0.02	85.22±0.05	66.88±0.48	67.36±0.22	74.92±2.55	78.42±0.65
Gaussian	9.68±0.41	10.49±0.46	10.00±0.00	27.38±17.38	15.29±0.57	19.76±3.66	4.64±0.13	4.83±0.25
Backdoor								
ACC	37.38±19.82	98.30±0.15	74.27±9.12	78.40±3.95	59.85±2.18	60.65±1.72	49.78±22.38	75.20±3.96
ASR	68.49±22.00	0.19±0.07	14.58±12.53	7.05±6.35	18.20±5.27	3.21±1.30	30.88±7.52	28.26±9.57
Scaling								
ACC	10.33±0.05	11.16±0.88	10.22±0.09	11.27±0.99	10.00±0.00	28.55±18.55	26.30±21.55	64.80±1.38
ASR	99.94±0.06	98.96±1.04	99.74±0.10	98.21±1.45	100.00±0.00	67.66±32.34	0.47±0.08	0.56±0.06
Neurotoxin								
ACC	81.17±15.39	95.73±1.45	70.00±7.85	79.58±1.60	22.40±7.16	45.40±3.22	47.29±18.07	79.99±0.70
ASR	23.19±2.25	18.11±1.67	20.65±2.21	18.12±4.16	51.63±1.03	57.42±1.91	40.42±4.35	9.00±1.29
Model Poison								
Krum	10.57±0.39	97.96±0.14	10.00±0.00	79.43±0.86	10.00±0.00	53.27±1.12	5.20±0.22	51.86±3.06
Trim	10.04±0.16	98.36±0.11	10.00±0.00	84.45±0.70	10.00±0.00	57.33±2.34	5.09±0.33	52.19±4.52
Min-Max	35.00±25.38	85.32±6.45	10.00±0.00	67.25±7.44	10.00±0.00	19.07±6.97	56.37±13.67	72.58±2.11
Min-Sum	96.69±0.94	95.97±0.59	10.88±0.87	83.93±0.81	17.40±4.27	43.94±3.56	52.56±23.91	72.36±1.61
QV-Adaptive	71.43±22.67	56.94±23.95	35.92±4.60	62.13±11.25	10.00±0.00	11.14±1.14	22.08±18.72	43.78±20.72

Table 3.1: Comparison of FEDQV and FEDAVG on four benchmark datasets under 10 attack scenarios with 30% malicious parties. The best results are highlighted in bold.

to FEDAVG. In the context of model poisoning attacks, FEDQV outperforms FEDAVG, except for the QV-Adaptive attack, which is tailored for FEDQV. Especially for local model poisoning attacks: Trim and Krum attacks, FEDQV outperforms FEDAVG by at least *4 times* in terms of accuracy. Setting this observation as the alternative hypothesis H_1 and using the Wilcoxon signed-rank test, we can reject the null hypothesis H_0 at a confidence level of 1% in favour of H_1 , proving the robustness of FEDQV.

Varying the percentage of attackers: Then we examine the performance of our method as the proportion of attackers increases. Figure 3.4 shows the changes in performance metrics for varying percentages of attackers for both FEDQV and FEDAVG under the backdoor attack (depicted in Figure 3.4a) and Neurotoxin attack (shown in Figure 3.4b) for both FEDQV and FEDAVG. Across scenarios where the percentage of attackers m ranges from 10% to 50%, FEDQV consistently outperforms FEDAVG in terms of both ACC and ASR under both attacks, even in scenarios where half the parties are malicious. Notably, the Neurotoxin attack exhibits a more pronounced impact on ACC reduction and ASR increase in both FEDQV and FEDAVG compared to the backdoor attack as the percentage of attackers varies.

However, we notice that none of these methods yields satisfactory accuracy results for Gaussian and Scaling attacks. To address this, we present the enhanced version of FEDQV with an adaptive budget assigned according to a reputation model.

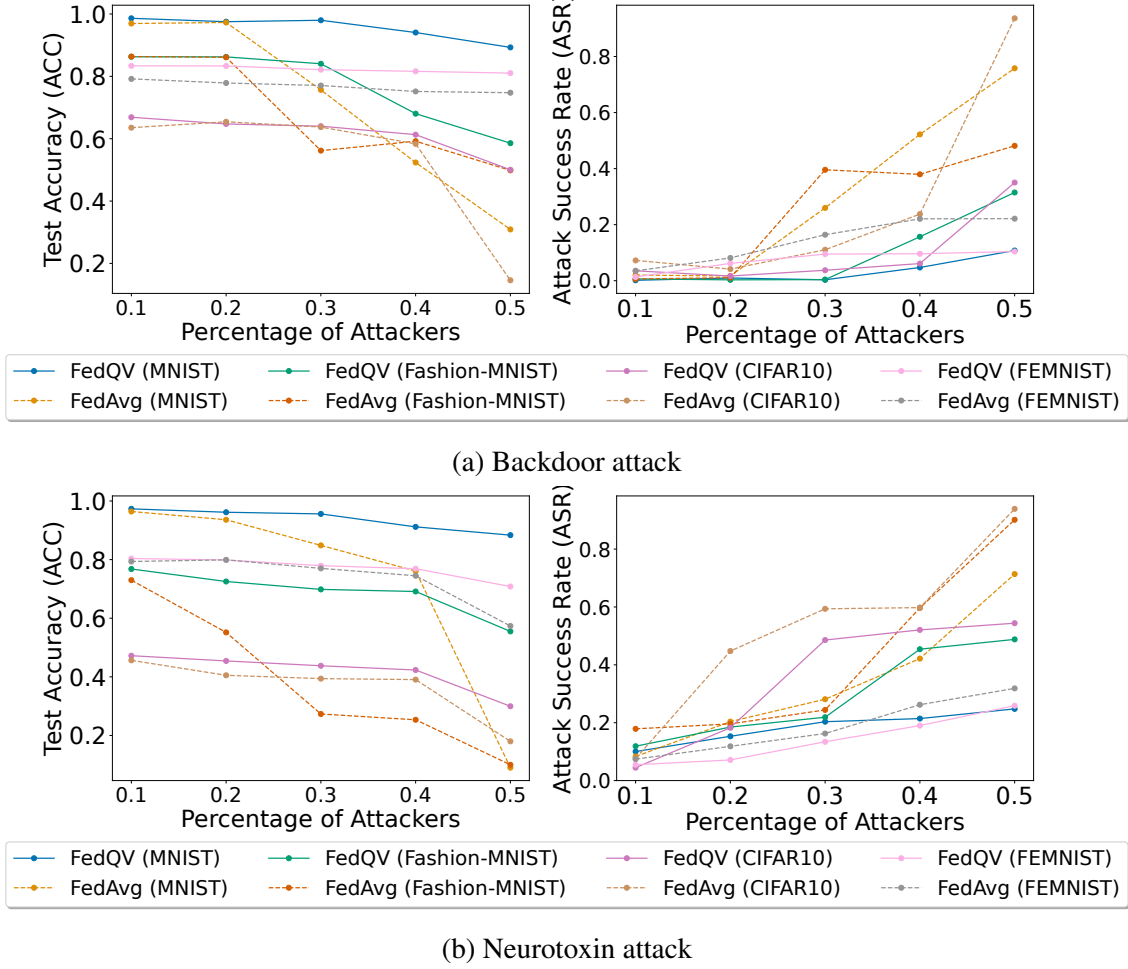
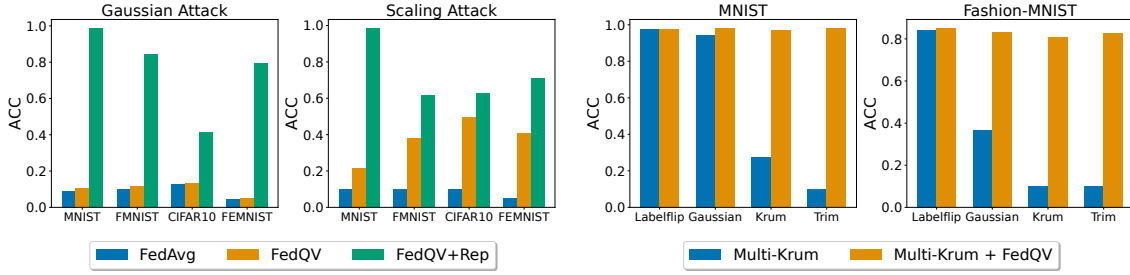


Figure 3.4: Performance comparison of FedQV and FedAvg in terms of Accuracy (ACC) and Attack Success Rate (ASR) over 100 epochs across four benchmark datasets under two targeted attacks, with varying percentages of attackers m from 10% to 50%.

3.5.7. Adaptive Budget

To enhance the robustness of FedQV, we integrate it with the reputation model proposed in [1] to allocate unequal budgets based on the reputation scores of parties in each round t , as detailed in Section 3.3.4. The performance of FedQV with an adaptive budget (referred to as FedQV+REP) is compared with FedAvg and FedQV during the Gaussian and Scaling attacks, with the percentage of attackers increased to 50%. As depicted in Figure 3.5a, the combination of FedQV and the reputation model significantly enhances resistance against Gaussian and Scaling attacks by at least a factor of 26%. This observation, formulated as the alternative hypothesis H_1 , is supported by the Wilcoxon signed-rank test, where the null hypothesis H_0 can be rejected at a confidence level of 1% in favour of H_1 . This integration effectively enhances the robustness of FedQV, rendering it successful in defending against the two attacks that it previously failed to counter.



(a) Performance comparison of FEDAVG, FEDQV, and FEDQV+REP, in terms of ACC for 100 epochs in four benchmark datasets under 2 attack scenarios with 50% malicious parties. (b) Test accuracy (ACC) for 100 epochs of both Krum alone and Multi-Krum + FEDQV on two benchmark datasets under 4 untargeted attack scenarios with 30% malicious parties.

Figure 3.5: Performance comparison of FEDAVG, FEDQV, and FEDQV+REP (FEDQV with adaptive budgets based on reputation model). Performance comparison of Krum alone and Multi-Krum integrated with FEDQV.

	Multi-Krum	Multi-Krum + FedQV	Trimmed-Mean	Trimmed-Mean + FedQV	Rep	Rep + FedQV
Neurotoxin						
1%	78.99±1.03/1.05±0.01	80.61±0.66/0.86±0.17	85.23±1.77/0.59±0.13	84.75±0.84/0.45±0.06	80.99±1.15/0.84±0.32	85.82±0.55/0.38±0.06
5%	76.21±0.73/3.29±0.92	80.32±1.07/ 1.34±0.34	85.15±0.38/0.87±0.17	85.09±0.97/0.73±0.05	80.48±1.17/1.62±0.11	84.12±0.38/1.35±0.40
10%	72.79±1.02/21.73±7.07	77.41±1.36/16.30±3.01	85.13±0.70/2.32±0.27	84.68±0.73/1.45±0.21	80.86±0.86/1.30±0.10	83.78±0.09/0.66±0.04
Min-Max						
10%	71.62±4.48	79.48±0.82	75.41 ±0.77	78.46±0.67	72.66±1.34	76.98±1.48
30%	52.29±0.34	58.42±4.92	59.62 ±1.20	60.24±6.81	54.94±0.82	58.02±0.71
50%	10.28±0.28	22.95±9.94	9.47±0.42	10.64±0.63	11.23±1.00	13.64±2.58
QV-Adaptive						
10%	52.98±1.78	73.35±3.44	83.17±1.85	85.55±0.33	12.60±2.36	41.55±19.78
30%	34.93±14.60	55.07±11.40	29.14±19.14	42.17±25.95	12.44±2.44	38.44±14.32
50%	10.20±0.20	12.24±1.12	10.00±0.00	13.35±3.37	10.00±0.00	10.55 ±0.44

Table 3.2: Comparison of Multi-Krum, Trimmed-Mean, Reputation, and their integration with FEDQV under three State-of-the-Art attacks on FEMNIST dataset. The best results are in bold. The results of targeted attacks are in the form of “ACC / ASR”.

3.5.8. Integration with Byzantine-robust Aggregation

Our objective is *not to position FedQV in competition with existing defence techniques but rather to demonstrate that FedQV can act as a complementary approach to advanced defences*. Hence, FEDQV can be seamlessly integrated into Byzantine-robust defence to enhance the overall defence performance, by adapting the weight calculation process. We illustrate this with examples using Muilt-Krum [39], Trim-mean [53] and Reputation [1]. Table 3.2 presents a comparative analysis illustrating the performance of these three defense methods both individually and when integrated with FEDQV under three state-of-the-art attacks conducted on the FEMNIST dataset. The results highlight that the integration of FEDQV with these defense mechanisms consistently yields superior performance characterised by higher ACC and lower ASR compared to their standalone counterparts.

Especially, in the context of Multi-Krum, as illustrated in Figure 3.5b, the integration of FEDQV (referred to as Multi-Krum + FEDQV) results in a significant enhancement

in accuracy under 4 attack scenarios. Notably, under local poisoning attacks such as the Krum attack and Trim attack, the accuracy improves by a factor of 2.5. The results presented in Table 3.3 elucidate that the incorporation of FEDQV into Multi-Krum yields an average ACC enhancement of 26.72% and a significant average reduction of 70% in the ASR under two targeted attack scenarios: backdoor and scale attack. Setting this observation as the alternative hypothesis H_1 and Employing the Wilcoxon signed-rank test, we reject the null hypothesis H_0 at a confidence level of 1% in favour of H_1 .

	MNIST		Fashion-MNIST	
	Multi-Krum	+ FEDQV	Multi-Krum	+ FEDQV
Backdoor				
ACC	70.20±9.99	89.96±1.85	33.24±13.24	70.89±3.17
ASR	32.03±11.20	9.59±2.28	68.87±17.77	9.72±4.50
Scaling				
ACC	68.35±16.76	96.55±0.41	59.43±14.22	82.48±0.24
ASR	33.65±19.15	0.41±0.06	33.64±19.08	0.91±0.18

Table 3.3: Comparison of Multi-Krum and Multi-Krum + FEDQV under two targeted attacks with 30% malicious parties in MNIST and Fashion-MNIST dataset. The best results are in bold.

Under small percentages of attackers: Unlike untargeted attacks, strong targeted attacks can be mounted with just a single attacker and data poisoning [169]. To investigate the behaviour of FEDQV in scenarios with finer gradations, we also evaluate it with small, realistic percentages of attackers, same as in [169], in Table 3.2 and Table 3.4. In Table 3.2, we assess the performance of three robust aggregation methods and their integration with FEDQV under the Neurotoxin attack, with 1%, 5%, and 10% attacker proportions. Table 3.4 examines the performance of Trimmed-Mean and Trimmed-Mean Integrated with FEDQV under two attack types, Backdoor and QV-Adaptive, with 1%, 5%, and 10% attacker proportions.

The results indicate that under these small, realistic percentages of attackers, the performance of FEDQV remains consistent with its behaviour under larger attack scenarios consistently outperforming the robust aggregation alone. The small percentage of attackers did not significantly impact the performance of the models, with almost unaffected accuracy (ACC) and a small attack success rate (ASR). This outcome underscores that integration with FEDQV enhances the robustness of the original defence mechanism across varying threat levels.

These findings underscore the potential of FEDQV *as a promising complementary method* to augment the performance of existing defence mechanisms.

	Trimmed-Mean	Trimmed-Mean-QV
Backdoor	ACC(%) / ASR(%)	ACC(%) / ASR(%)
1%	84.99/0.57	85.67/0.52
5%	84.83/0.93	85.66/0.46
10%	85.45/2.27	85.06/1.79
QV-adaptive	ACC(%)	ACC(%)
1%	84.13	86.38
5%	83.88	85.51
10%	79.49	85.74
30%	10.00	73.95
50%	10.00	10.00

Table 3.4: Comparison of Trimmed-Mean and Trimmed-Mean Integrated with FedQV Methods under Targeted Attacks (Backdoor and QV-Adaptive) Across Varying Percentages of Malicious Parties.

	Fashion-MNIST			FEMNIST		
	FEDPV	FEDCV	FEDQV	FEDPV	FEDCV	FEDQV
Data Poison						
Labelflip	55.64±27.33	81.79±0.25	82.59±0.42	26.36±23.44	52.42±25.21	76.76±1.74
Neurotoxin						
ACC	55.60±22.26	74.63±6.28	82.96±1.48	33.31±32.60	9.37±0.58	4.53±0.65
ASR	33.31±32.60	9.37±0.58	4.53±0.65	56.39±27.94	18.01±0.98	10.62±0.10
Model Poison						
Krum	47.35±18.74	71.68±5.85	75.88±2.31	5.04±0.38	69.45±0.43	76.58±0.78
Trim	10.00±0.00	75.98±1.06	76.65±1.30	5.04±0.05	72.88±0.59	78.74±0.22
Min-Max	48.44±19.70	74.98±0.96	79.54±1.30	50.28±22.22	68.89±2.22	72.91±2.31

Table 3.5: Comparison of FEDPV, FEDCV and FEDQV on two benchmark datasets under 5 attack scenarios with 20% malicious parties. The best results are highlighted in bold.

3.5.9. Comparison with Alternative Voting Strategies

As mentioned in the Background, many voting strategies employed in FL rely on majority voting, which can encounter the issue of the tyranny of the majority problem in election systems. QV presents a potential solution to mitigate this problem. In addition to comparing FEDQV with alternative voting methods, we integrate Plurality voting [170] into FL. In Plurality voting, the candidate with the highest number of votes wins, regardless of whether they secure an absolute majority (more than 50%) of the total votes. We assign the weight of aggregation of the winner as one, while the rest are zero. We denote this integration of Plurality voting into FL as FEDPV. Furthermore, to underscore the unique attributes of QV compared to other nonlinear functions, we not only provide the theo-

retical evidence as mentioned in Section 3.3.2, but also introduce a voting strategy using cubic voting instead of quadratic on the top of FEDQV, denoted as FEDCV.

To empirically assess the performance of these three mechanisms we conduct five poisoning attacks (comprising two data poisoning attacks and three model poisoning attacks) across two datasets, Fashion-MNIST and FEMNIST, each with 20% malicious parties. The results, presented in Table 3.5, demonstrate that under these poisoning attacks, FEDQV exhibits superior performance compared to the alternative voting strategies. More specifically, although FEDCV also has a better performance than FEDPV, indicating the efficacy of the defense mechanism in FEDQV, it nonetheless fails to surpass FEDQV when employing cubic vote pricing rule. This underscores FEDQV’s capability to effectively restrain malicious behaviour while not imposing overly restrictive constraints on the benign ones—a balance not achieved by other nonlinear functions.

3.5.10. Non-IID Degree

To fulfil the fundamental setting of a heterogeneous and unbalanced dataset for FL, our experimental setup integrates datasets exhibiting non-IID properties, with a non-IID degree (ι) of 0.9, consistent with the settings outlined in previous studies such as [1], [34]. The ι represents the concentration parameter utilised in sampling from a Dirichlet distribution [104], which governs the degree of dataset imbalance. This parameter choice aims to generate non-IID and unbalanced data partitions conducive to our experimental objectives.

Moreover, we have examined the performance of FEDQV and FEDAvg across varying degrees of non-IID data, ranging from 0.1 to 0.9, as depicted in Table 3.6. These results demonstrate that as the non-IID degree increases among the parties, the performance of the global model declines. Notably, FEDQV consistently maintains a superior performance compared to FEDAvg, even when confronted with different degrees of data heterogeneity under attack conditions.

	Non-IID	0.1	0.3	0.5	0.7	0.9
FedQV	ACC(%)	84.94	86.01	83.88	81.37	75.96
	ASR(%)	3.39	4.55	17.64	20.59	24.18
FedAvg	ACC(%)	81.27	81.1	82.44	80.77	65.68
	ASR(%)	3.37	13.39	20.84	22.99	60.35

Table 3.6: Comparison of Accuracy (ACC) and Attack Success Rate (ASR) for FedQV and FedAvg under Backdoor Attack over 100 epochs with varying Non-IID Degrees on Fashion-MNIST Dataset.

3.5.11. Impact of Hyper-parameters

As noted, Theorem 3.4.12 provides general guidelines for hyperparameters tuning. As shown in Remark 3.4.14, the error rate is influenced by B and θ . To demonstrate the impact of these two hyper-parameters, we conduct a grid search over B in $[10, 20, 30, 40, 50]$ and θ in $[0.1, 0.2, 0.3, 0.4, 0.5]$. The setup is the same as on the MNIST dataset under the backdoor attack with 30% malicious parties. Figure 3.6 illustrates the stability of the metrics ACC and ASR when varying the hyperparameters B and θ . Specifically, as B increases, there is a discernible decline in ACC accompanied by an increase in ASR, attributable to the augmented voting budget allotted to malicious parties. Conversely, an increase in θ yields an elevation in ACC coupled with a reduction in ASR, as a higher threshold implies stricter acceptance criteria for abnormal similarity scores. The optimal values of B and θ are determined to be 30 and 0.2, respectively.

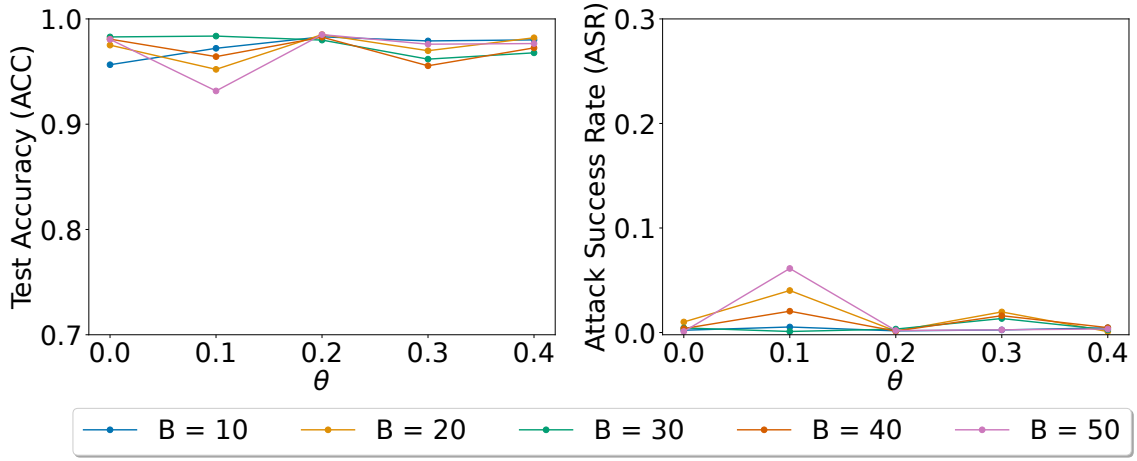


Figure 3.6: Impact of varying hyper-parameters budget B and similarity threshold θ of FEDQV on ACC and ASR under a backdoor attack scenario with 30% malicious parties, evaluated on the MNIST dataset.

We can see from Theorem 3.4.12, that the number of malicious devices m will affect the algorithm, and more malicious devices can lead to increased damage. However, this does mean the server needs to know the number of malicious devices to do the fine-tuning. We agree that determining optimal parameters can be challenging, especially in the absence of complete knowledge about the FL system. A better tuning is possible if more information is available. For specific tasks, more information can indeed be collected from which practical parameter sets can be extracted either via exhaustive search or via simpler online algorithms using trial and error. We will add this to our future work and consider it when we study particular domain-specific problems using our method.

3.6. Defence against Privacy Attacks

To illustrate the compatibility of FEDQV with existing privacy-guaranteeing mechanisms, we integrate Secure Aggregation into our FEDQV framework. Through this integration, we aspire to establish a more privacy-preserving defence mechanism in robust FL against potential privacy attacks.

3.6.1. Secure Aggregation for FL

Due to the sensitive nature of the updates sent by clients and the threat from the aforementioned privacy attacks, secure aggregation protocols have been proposed as potential countermeasures [146]. Secure aggregation can be achieved using four main privacy-enhancing technologies: differential privacy [147], trusted-execution environment [148], secure shuffling under anonymity assumptions [149], and cryptography. Among those, secure aggregation based on cryptography is the most widely studied [110]–[112]. Existing secure aggregation solutions have been proposed for different contexts [146], including under malicious clients, a malicious aggregator(s), and for varying communication and computation costs, etc. Although crypto-based secure aggregation provides strong security guarantees compared to alternatives, i.e., differential privacy, it also suffers from high computational and communication overhead making it impractical for many real-life use cases. Among the different proposals, the one based on masking [110] outperforms its alternatives in terms of performance and ensures a strong security guarantee.

A secure aggregation protocol can guarantee *input confidentiality* by safeguarding the local input model updates of each participant and revealing only the globally aggregated model during FL training, thereby efficiently averting or alleviating concerns regarding the aforementioned privacy attacks. For our work, we utilise the SECAGG proposed by [110] due to its performance in terms of communication and computational overhead. Below we overview the SECAGG protocol of [110] (see [110] for their complete protocol). The secure aggregation protocol consists of three phases: SECAGG.Setup, SECAGG.Protect, and SECAGG.Aggregate.

- In the SECAGG.Setup phase, each client and the aggregator get the public parameters and the key materials. Each client generates two masks: (k_i, b_i) where k_i is generated using a key agreement protocol (e.g., Diffie Hellman [171]), and b_i is generated via a pseudorandom generator. Each client secret shares its mask with other clients using threshold secret sharing [172]. With the secret sharing, the masks of dropped clients can be recovered as long as threshold-many clients are still alive.
- In the SECAGG.Protect phase, each client protects their local inputs using their unique private key and sends the protected input, denoted by $[w_i^t]$, to the aggregator.

$$[w_i^t] = w_i^t + k_i + b_i \quad (3.35)$$

- In the **SECAGG.Aggregate** phase, after the aggregator collects all the protected inputs from the clients, it executes an aggregation algorithm to retrieve the sum of the client's inputs. The aggregator first collects threshold-many shares of the seed of each mask $\mathbf{b}_i$ for every alive client i and reconstructs it. Then it gets threshold-many shares of the Diffie-Hellman's secret key of the dropped clients and thus reconstructs the missing masks $\mathbf{k}_j$ for every dropped client j .

$$\sum_{i \in [N]} \mathbf{w}_i^t = \sum_{i=1}^N [\mathbf{w}_i^t] - \sum_{i \in \text{Drop}} \mathbf{b}_i + \sum_{j \in \text{Live}} \mathbf{k}_j \quad (3.36)$$

All masking and unmasking operations involve modular addition.

3.6.2. Implementing SECAGG in FEDQV

FEDQV differs from FEDAVG by instead using a voting mechanism to weigh the contributions of $\mathbf{w}_i^t$ to the global model. This makes it not-trivial to integrate upon FEDQV secure aggregation protocols developed for FEDAVG. Related integration challenges include the following:

- *Aggregation of $\{\mathbf{w}_i^t\}_{i \in [N]}$.* Recall that the aggregation formulated by FEDQV is

$$\frac{1}{\sum_{i=1}^N v_i^t} \sum_{i=1}^N v_i^t \cdot \mathbf{w}_i^t$$

Assume a client i masks the value of $\mathbf{w}_i^t$ as in Equation 3.36. However, there is one more parameter v_i^t to be added in the calculation. As shown by the equation below, if the server simply multiplies $[\mathbf{w}_i^t]$ by its corresponding v_i^t , the masks do not cancel each other as supposed. Thus, v_i^t must be added before a client masks its $\mathbf{w}_i^t$.

$$\begin{aligned} & \sum_{i=1}^n v_i^t \cdot [\mathbf{w}_i^t] - \sum_{i \in \text{Drop}} \mathbf{b}_i + \sum_{j \in \text{Live}} \mathbf{k}_j \\ & \sum_{i \in [n]} \mathbf{w}_i^t \neq \sum_{i=1}^n v_i^t \cdot (\mathbf{w}_i^t + \mathbf{k}_i + \mathbf{b}_i) - \sum_{i \in \text{Drop}} \mathbf{b}_i + \sum_{j \in \text{Live}} \mathbf{k}_j \end{aligned}$$

- *Calculation of v_i^t .* As mentioned, each client shall add its v_i^t to its $\mathbf{w}_i^t$, but in FEDQV v_i^t can only be calculated by the server based on the budget B_i , similarity s_i^t of client i . To overcome this shortcoming, the server, after receiving s_i^t of each client i , computes v_i^t using Equation 3.4 and sends v_i^t back to its corresponding client. Then each client computes **SECAGG.Protect** using $\langle \mathbf{k}_i, \mathbf{b}_i \rangle$ in which the input is assigned as $\mathbf{w}_i^t \cdot v_i^t$. Note that this does not harm the security and privacy of the clients while introducing negligible computational overhead.

Considering the crucial adjustments above, let us form the securely aggregated FEDQV. Our approach is in line with other securely aggregated FL designs consisting of three main stages: 1) *Setup*; 2) *Generation and Protect*; and 3) *Aggregate*. Our integration of secure aggregation to FEDQV is depicted in Figure 3.7. We further discuss its details below:

Stage 0 (Setup). In this stage, the server and the partis compute SECAGG.Setup and initialise FEDQV by having the server send values of the protocol's parameters to each party, i.e. w^{t-1} .

Stage 1 (Generation and Protect). After the setup and initialisation stage, each party first computes its local model w_i^t as in FEDQV and computes its s_i^t . Later, it transmits the message $\langle |D_i|, s_i^t \rangle$. Upon receiving $\langle |D_i|, s_i^t \rangle$, the server computes v_i^t as in Equation 3.4 and transmits v_i^t to the corresponding party. Then party i first computes $w_i^t \cdot v_i^t$ and protects its input $w_i^t \cdot v_i^t$ by treating it an input to the SECAGG.Protect as $[w_i^t] = w_i^t \cdot v_i^t + k_i + b_i$. Party i sends $[w_i^t]$ to the server.

Stage 2 (Aggregate). Upon receiving $[w_i^t]$, the server first calls SECAGG.Aggregate as described previously which returns the sum of all w_i^t . The server requires one more step to finalise by multiplying the sum retrieved by $\frac{1}{\sum_{i=1}^N v_i^t}$, as it is the sole entity possessing complete knowledge of all v_i^t .

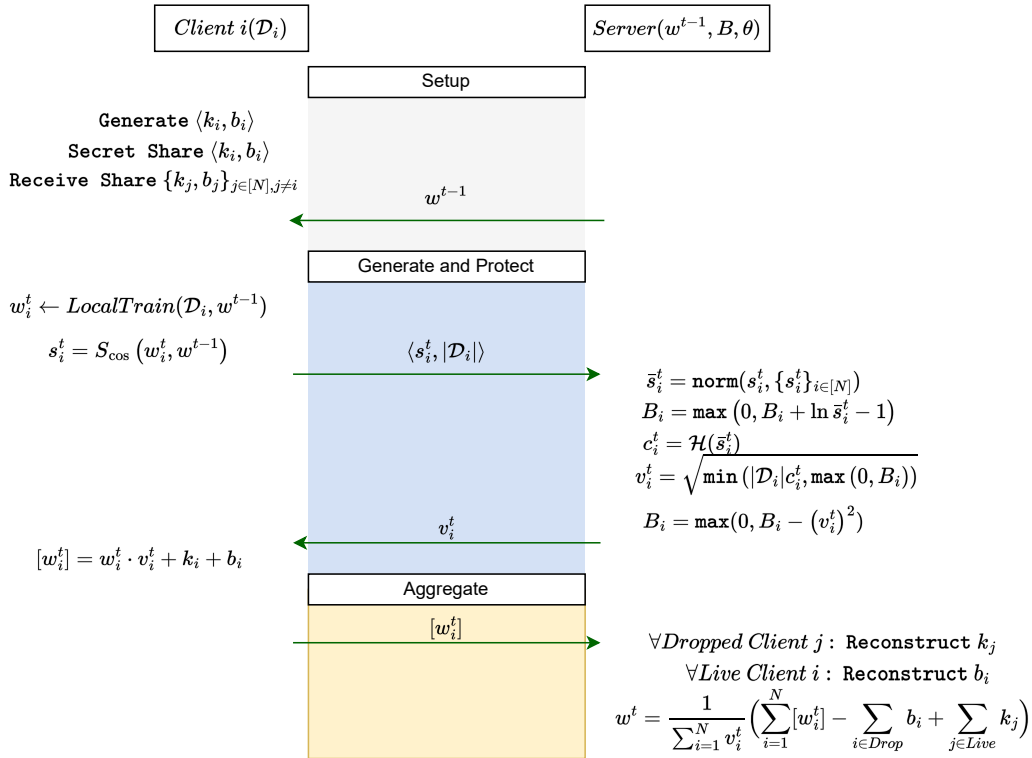


Figure 3.7: Overview of implementation of the SECAGG protocol in FEDQV.

Due to the nature of FEDQV and our adjustments, there are two attack scenarios we consider: 1) Privacy attacks by the server; and 2) Client attacking v_i^t . In the former sce-

nario, the server knows only s_i^t of each client, therefore the server can't reconstruct the local model ($\mathbf{w}_i^t$) of clients by relying solely on similarity scores. Moreover, s_i^t can be protected from the clients by utilising cryptography. In the latter scenario, it can be a concern that each client possesses the knowledge of their vote. This issue can be effectively mitigated by enforcing clients to provide proof (e.g., via zero-knowledge proofs [173]) that the s_i^t and v_i^t sent are indeed computed honestly and correctly, adhering to the protocol rules. Moreover, the server can also protect v_i (as clients do for their $\mathbf{w}_i^t$) and then share the protected version. Homomorphic Encryption [174] can be deployed to actualise our approach, but we leave further investigation as future work.

3.6.3. Privacy Attacks against FEDQV with SECAGG

we conduct a comprehensive evaluation to quantify the privacy-preserving properties of FEDQV with and without SECAGG. By evaluating these methods under real-world conditions, we seek to evaluate their effectiveness under true attack scenarios.

Experimental Setup We evaluate models Conv-2 [163] along with the FEMNIST [163]. In each round of training, we randomly select 10 clients from a pool of 193 users for the FEMNIST dataset. The training process involves 100 iterations with a batch size of 20, 1 local training epoch and a learning rate of 0.25.

Privacy Attack We mount the Deep Leakage from Gradients (DLG) attack [70] and the Gradient Inversion (GI) attack [37] on FEDQV. These attacks aim to generate dummy data and corresponding labels by leveraging a gradient-matching objective. The detailed algorithms are outlined in Algorithm 5. In each iteration t , the attack algorithms proceed

Algorithm 5: DLG and GI Attacks

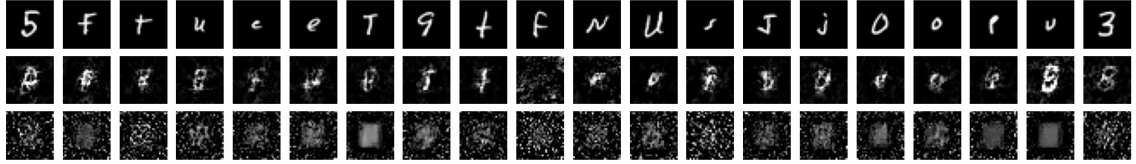
- 1 **Input:** $F(\mathcal{D}_i; \mathbf{w}_i^t)$: model at round t from targeted user i ; learning rate η for inverting gradient optimiser; S : max iterations for attack; τ : regularisation term for cosine loss in inverting gradient attack; d : the model size
 - 2 **Output:** reconstructed training data $(\mathcal{D}_i, y_i)$ at round t
 - 3 Initialise $\mathcal{D}'_0 \leftarrow \mathcal{N}(0,1)$, $y'_0 \leftarrow \text{Randint}(0, \max(y))$
 - 4 **for** $s \leftarrow 0, 1, \dots, S-1$ **do**
 - 5 $\nabla \mathbf{w}'_s \leftarrow \partial \ell(F(\mathcal{D}'_s, \mathbf{w}_i^t), y'_s) / \partial \mathbf{w}_i^t$
 - 6 **switch Case do**
 - 7 **case DLG attack do** $\mathcal{L}'_s \leftarrow \|\nabla \mathbf{w}'_s - \nabla \mathbf{w}_i^t\|^2$
 - 8 **case GI attack do** $\mathcal{L}'_s \leftarrow 1 - \frac{\nabla \mathbf{w}_i^t \cdot \nabla \mathbf{w}'_s}{\|\nabla \mathbf{w}_i^t\| \|\nabla \mathbf{w}'_s\|} + \tau$
 - 9 $\mathcal{D}'_{s+1} \leftarrow \mathcal{D}'_s - \eta \nabla_{\mathcal{D}'_s} \mathcal{L}'_s$, $y'_{s+1} \leftarrow y'_s - \eta \nabla_{y'_s} \mathcal{L}'_s$
 - 10
 - 11
 - 12 **return** $\mathcal{D}'_S, y'_S$
-

as follows:

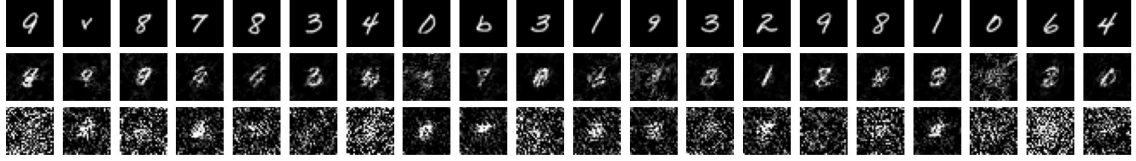
1. The attack randomly initialises a set of dummy data, comprising dummy inputs $\mathcal{D}'_0$ and dummy labels y'_0 ;
2. After the dummy gradient w' is acquired, the server then updates the dummy data in the direction that minimises the Euclidean distance (for DLG) or the cosine distance (for GI) between the dummy gradient and the real gradient ∇w_i^t .

We initiate these attacks with a learning rate of 0.01 and perform 10,000 attack iterations.

Evaluation Results We conduct the Deep Leakage from Gradients (DLG) and Gradient Inversion (GI) attacks on the FEDQV framework, both with and without the integration of SECAGG.



(a) The Deep Leakage from Gradients (DLG) Attack.



(b) The Gradient Inversion (GI) Attack.

Figure 3.8: Visual Comparison of Images Reconstructed by Privacy Attacks (DLG and GI) on FEDQV with and without SECAGG. The first row displays the original private training images from the targeted client. The second row illustrates reconstructed images resulting from the privacy attacks (DLG and GI) on FEDQV without SECAGG. In the third row, reconstructed images from the privacy attacks (DLG and GI) on FEDQV with SECAGG are presented. The use of SECAGG is observed to contribute significantly to mitigating information leakage from the images.

Figure 3.8a provides visualisations of the reconstructed images by the DLG attack. In the absence of SECAGG, the DLG attack can successfully recover approximately 8 out of 20 private images from the party. However, upon integrating SECAGG, the DLG attack fails to recover any of the party's images. Figure 3.8b presents visualisations of the reconstructed images by the GI attack. Without SECAGG, the GI attack successfully recovers half of the private images from the party. Yet, with SECAGG, the GI attack is unable to recover any party images, although it may vaguely leak the overall shape of the image. These results highlight the efficacy of SECAGG in strengthening FEDQV against privacy attacks, with the GI attack exhibiting a higher level of attack potency compared to the DLG attack.

The integration of SECAGG into the FEDQV framework demonstrated significant improvements in resisting privacy attacks. Notably, SECAGG successfully prevented information leakage in scenarios where FEDQV alone exhibited vulnerabilities. This empirical evidence contributes valuable insights, affirming the effectiveness of implementing SECAGG atop FEDQV as a privacy-enhancing measure in robust FL.

3.7. Conclusion

In this chapter, we have proposed FEDQV, a novel aggregation scheme for FL based on quadratic voting instead of $1p1v$, which is the underlying principle that makes the currently employed FEDAVG vulnerable to poisoning attacks. The proposed method aggregates local models based upon the votes from a truthful mechanism employed in FEDQV. The efficiency of the proposed method has been comprehensively analysed from both a theoretical and an experimental point of view. Collectively, our performance evaluation has shown that FEDQV achieves superior performance than FEDAVG in defending against various poisoning attacks. Moreover, FEDQV is a reusable module that can be integrated with reputation models to assign unequal voting budgets, incorporate Byzantine-robust techniques, and employ privacy-preserving mechanisms. This versatility provides robustness against both poisoning and privacy attacks, positioning FEDQV as a promising complement to existing aggregation in FL.

4. QUANTIFYING AND PRESERVING PRIVACY IN PRUNED FEDERATED LEARNING

After addressing the resilience of FL to poisoning attacks, we now shift focus to its privacy aspects. In this chapter, we focus on quantifying and preserving privacy in model-pruned FL. Model pruning has been proposed as a technique for reducing the size and complexity of Federated learning models. By making local models coarser, pruning is intuitively expected to improve protection against privacy attacks. However, the level of this expected privacy protection has not been previously characterized, or optimized jointly with utility.

We first characterize the privacy offered by pruning. We establish information-theoretic upper bounds on the information leakage from pruned FL and we experimentally validate them under state-of-the-art privacy attacks across different FL pruning schemes. Second, we introduce *PriPrune*— a privacy-aware algorithm for pruning in FL. *PriPrune* uses defense pruning masks, which can be applied locally after *any* pruning algorithm, and adapts the defense pruning rate to jointly optimize privacy and accuracy. Another key idea in the design of *PriPrune* is *pseudo-pruning*: it undergoes defense pruning within the local model and only sends the pruned model to the server; while the weights pruned out by defense mask are withheld locally for future local training rather than being removed. We show that *PriPrune* significantly improves the privacy-accuracy tradeoff compared to state-of-the-art pruned FL schemes. For example, on the FEMNIST dataset, *PriPrune* improves the privacy of *PruneFL* by 45.5% without reducing accuracy.

4.1. Introduction

Federated Learning (FL) is known to have several benefits in terms of reducing the communication, computation and storage costs for the users, training better global models, and raising the bar for privacy by not sharing the local data. FL allows users to train models locally on their (user) devices without revealing their local data but instead collaborate by sharing only model updates that can be combined to build a global model through a central server. A typical FL scenario involves numerous users and resource-constrained devices [175] making it challenging to perform resource-intensive tasks like training Deep Neural Networks (DNNs) on them.

In parallel, significant efforts have been put toward optimizing sparse DNNs to create lightweight models suitable for training on edge devices [176]–[178]. Model pruning, which involves the removal of a certain percentage of parameters, has gained significant attention [179]–[181]. Its primary objective is to derive a sparse DNNs model that decreases the computational requirements and enhances efficiency without compromis-

ing accuracy [182]. Several model pruning techniques have been developed for FL to optimize model sparsity under given constraints on the processing and communication capabilities of devices, especially under heterogeneous clients, [78], [79].

It is intuitively expected that by reducing the number of weights of FL models, the pruning method should, as a side-effect, also improve their resistance to reconstruction attacks. These attacks aim at using these weights to reverse-engineer the model and obtain information about the input data; a.k.a. gradient inversion attacks launched by the server. Prior work on model pruning focuses solely on the scalability/accuracy trade-off and thus neither considers nor quantifies the privacy impact of pruning on FL. To the best of our knowledge, our paper is the first to consider privacy in the context of FL model pruning, and to address the following key questions: **Q1.** *Can we quantify privacy in FL model pruning?* **Q2.** *How can we further optimize pruning for privacy, and jointly with accuracy?* To address these questions, we make the following two contributions.

First, **we theoretically and empirically quantify privacy leakage in model pruning.** We derive information-theoretic upper bounds on the amount of information revealed about any single user’s dataset via model updates, in *any* pruned FL scheme. This is the first theoretical characterization of privacy leakage in pruned FL models. We also conduct a comprehensive empirical evaluation that quantifies the actual amount of privacy leakage of six different pruning schemes, considering several state-of-the-art privacy attacks. Within the family of gradient inversion attacks (including Deep Leakage from Gradients [70] and Gradient Inversion (GI) [37]), we also design a novel privacy attack (referred to as Sparse Gradient Inversion, or SGI), specifically tailored to exploit vulnerabilities inherent in FL model pruning. This evaluation combined with our theoretical analysis provides valuable insights into the choices and parameters that affect the privacy provided by pruning.

Building upon these insights, we make our second contribution: we design ***PriPrune*—a privacy-preserving pruning mechanism.** *PriPrune* defends against gradient inversion attacks in pruned FL, by performing additional local (defense) pruning, after any (base) FL pruning scheme. *PriPrune* applies a personalized defense mask and adapts the defense pruning rate, so as to jointly optimize model accuracy *and* privacy, using back-propagation augmented with Gumbel Softmax Sampling. Another key idea in the design of *PriPrune* is what we refer to as *Pseudo-Pruning*: it entails pruning with the defense mask within the local model and transmitting the pruned model to the server, thereby enhancing privacy. However, the weights pruned out by the defense mask in the local model are not discarded; instead, they are retained locally for subsequent rounds of local training, thereby preserving model accuracy. We evaluate *PriPrune* via comprehensive experiments across various FL pruning schemes, a state-of-the-art attack and several benchmark datasets. We show that it achieves a significantly better privacy-accuracy tradeoff than privacy-unaware pruned FL baselines. For example, on the FEMNIST dataset, *PriPrune* improves the privacy *PruneFL* [78] by 45.5% without imposing any accuracy loss.

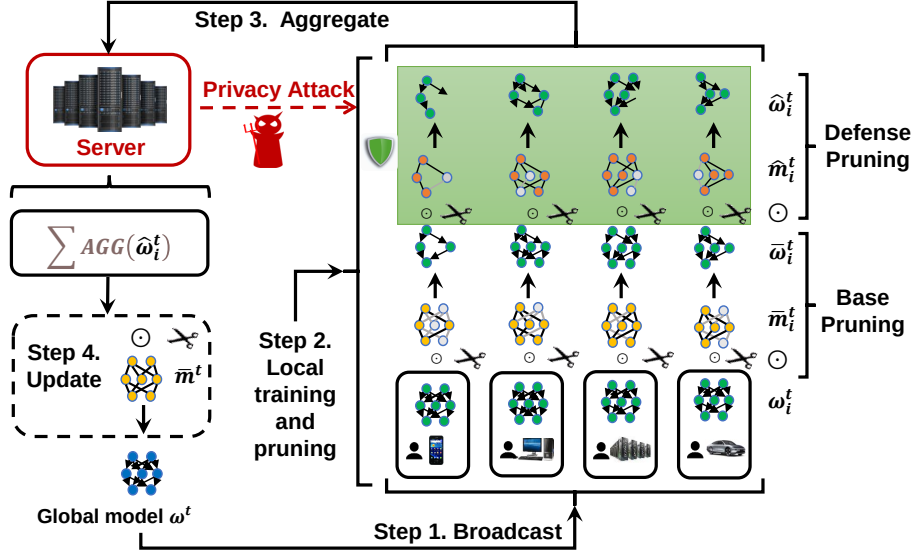


Figure 4.1: FL with model pruning, under a privacy attack by an honest-but-curious server. Step 1: the server broadcasts the global model with weight w^t to users. Step 2: After local training, user i employs base pruning with pruning mask $\bar{m}_i^t$, resulting in a pruned model with weights $\bar{w}_i^t$. The user then applies a defense pruning technique, yielding a pruned model with weights $\hat{w}_i^t$. Step 3, the server aggregates the model updates and launches a gradient inversion attack upon receiving $\hat{w}_i^t$. Step 4: the server applies global pruning with pruning mask $\bar{m}^t$, and updates the global model accordingly. Notice *Base* pruning can be *any* FL pruning scheme. *Defense* pruning is introduced, in this paper, specifically to protect against privacy attacks.

4.2. Related Work

Model Pruning in FL State-of-the-art research for model pruning in FL utilizes two main approaches: one involves the pruning mask being decided on the server side, and the client only using this mask without changing it, and the other involves the collaborative selection of the pruning mask by both the user and the server side. *PruneFL* [78] follows the first approach, aiming to reduce the size and complexity of FL models without compromising accuracy. It achieves this by removing less important weights and connections in each layer of the neural network based on their magnitude and importance scores. However, the pruning mask is only decided by the server without considering user-side local representations. *LotteryFL* [183] belongs to the second approach, introducing a method that allows users to maintain local representations by selecting a subset of the global network using personalized masks. However, the system of sparse models it produces performs well only on local datasets. *FedDST* [79] involves dynamic sparse training on both the client and server sides, extracting and training sparse sub-networks. Other recent state-of-the-art studies have introduced various gradient-based one-shot pruning algorithms at initialization. *Snip* [75] leverages connection sensitivity to retain critical connections and remove less significant ones, aiming to limit the loss due to pruning. On

the other hand, *GraSP* [77] focuses on preserving the gradient flow through the network by scoring weights based on the Hessian-gradient product. *SynFlow* [76], being a data-agnostic pruning algorithm, emphasizes preserving the total flow of synaptic strengths through the neural network to achieve Maximal Critical Compression during initialization. More importantly, none of them considered privacy implications in their design, as *PriPrune* does. By focusing on the privacy implications of pruning alone, our study seeks to provide a clear understanding of how pruning can be optimized for privacy in FL, independent of other privacy-preserving techniques like differential privacy or secure aggregation.

Reconstruction Attacks in FL In terms of privacy leakage, communicating gradients throughout the training process in federated learning can reveal sensitive information about the participants. *Deep leakage from gradients* (DLG) [70] demonstrated that sharing the gradients can leak private training data, including images and text. *Improved DLG* (iDLG) [36], presented an analytical procedure to extract the ground-truth labels from the shared gradients and showed the advantages of iDLG over DLG. *Inverting Gradients* [37] introduced cosine similarity as a cost function in their reconstruction attacks and employed total variation loss $\mathcal{L}_{TV}$ as an image prior. *Inverting Gradients* [37] and *GradInversion* [184] demonstrated the capability to reconstruct high-resolution images with increased batch sizes. The reconstruction quality of the attack was improved in *Reconstruction From Obfuscated Gradient* (ROG) [185]. ROG proposed conducting the reconstruction optimization in low dimensional space and trained a neural network as a postprocessing module. While other forms of privacy attacks have been proposed—such as those employing generative adversarial networks (GANs) to reconstruct user data or class prototypes [71], [72]—these methods typically rely on strong assumptions, including homogeneous data distributions across clients and access to auxiliary datasets. They are also often evaluated under idealised conditions, where intra-class variability is minimal. In contrast, gradient leakage attacks have proven to be more generalizable and have emerged as the predominant threat model in the FL privacy literature [73], [74].

Although these gradient leakage attacks have demonstrated the privacy vulnerabilities of standard federated learning, none of them has considered pruned federated learning. Therefore, we do not currently know how effective existing attacks are against the sparser models from pruned FL.

Privacy-Preserving Techniques in FL FL is particularly vulnerable to the aforementioned inverting-the-gradient type of reconstruction attacks, since it relies on gradient updates to operate. There exists a large body of literature on privacy-enhancing methods that protect against this type of attacks in FL. The adversary is typically an “honest-but-curious” server, *i.e.*, a server that participates correctly in the FL protocol, but tries to infer training data of individual clients based on their gradient updates. One family of defense approaches is based on differential privacy (DP) [156], [186]–[190], which can be ap-

plied locally (at the clients), centrally (at the server), or in a distributed way. In the local DP model for FL, each client adds noise to the local update before sharing with the server, which makes it difficult for the server to invert the true gradient and infer the client’s training data. However, local DP is known to significantly degrade utility, in this case model accuracy. Another defense approach applied to FL is Secure Aggregation (SecAgg) [111], [112], [191]–[193]. SecAgg ensures that individual updates are encrypted (thus preventing the server from inverting the gradients of individual users), while their aggregate can still be decrypted (thus enabling the server to train a good model). SecAgg is powerful but has computational overhead, although significant advances have been done recently.

Both DP and SecAgg are important but orthogonal to the underlying FL and can be combined with a number of FL protocols, including all the pruning protocols we consider as base in this study. Our proposed *PriPrune* can also be combined with DP and/or SA. For instance, after applying *PriPrune* to base pruning, clients can add local DP to the pruned updates before sending them to the server, and/or they can encrypt their pruned updates to enable SA. This paper focuses on pruning itself and the privacy-utility tradeoff that can be achieved, *before* adding orthogonal defense techniques. Future work could explore adding DP and/or SecAgg, which are out of the scope of this paper.

Recent works like *FedMap* [194] and *Fed-LTP* [195] discuss privacy specifically in the context of pruned FL. *FedMap*, which appeared after this submission, enhances privacy by training models from scratch and by avoiding the sharing of detailed pruning masks. However, it does not quantify the privacy leakage that may still occur from the pruned models. *FedMap* can be considered as a base pruning technique, and *PriPrune* can still be applied to protect it. *Fed-LTP*, on the other hand, combines Lottery Ticket Pruning with Zero-concentrated Differential Privacy (zCDP) to enhance privacy. Integrating zCDP, or any type of DP, with our proposed *PriPrune* is orthogonal to our core contribution, as discussed above, and can be explored as part of future research.

4.3. Problem Setup

Federated Learning (FL) with Pruning. We consider a general FL system with multiple users and one server, employing *any* model pruning scheme on the users and/or the server. We refer to pruning at this stage as *base* pruning with base pruning rate $\bar{p}$, and it can be any state-of-the-art pruning FL scheme [78], [79]. This is depicted in Fig. 4.1 and formalized in Algorithm 6, with consistent notations throughout the paper.

The goal is to train a global model $F(\mathcal{D}, \mathbf{w})$ through FL. At the beginning of round t , user i with the local data $\mathcal{D}_i$ and labels y_i has local model weights $\mathbf{w}_i^t$; the server broadcasts the most recent global model $F(\mathcal{D}, \mathbf{w}^t)$ to all participating users. Then the round proceeds as follows:

Local Training and Base Pruning: Each user i utilizes base pruning strategy $\mathbb{P}_i(\cdot)$ with

Algorithm 6: FL with Pruning under Privacy Attack

- 1: **Given:** T number of global training rounds; N number of users indexed by i ; E number of local epochs; the server aims to reconstruct the local data of the target user.
 - Server executes:**
 - 2: **Initialize** $\mathbf{w}^0$
 - 3: **for** $t \leftarrow 1$ **to** T **do**
 - 4: **for** each user $i \in N$ **in parallel do**
 - 5: $\hat{\mathbf{w}}_i^t \leftarrow \text{UserUpdate}(\mathbf{w}^{t-1}, \mathbf{m}^{t-1}, i)$
 - 6: **if** user i is targeted **then**
 - 7: $\nabla \hat{\mathbf{w}}_i^t \leftarrow \hat{\mathbf{w}}_i^t - \hat{\mathbf{w}}_i^{t-1}$
 - 8: Privacy Attack on $(F(\mathcal{D}_i; \hat{\mathbf{w}}_i^t), \hat{\mathbf{w}}_i^t, \nabla \hat{\mathbf{w}}_i^t)$
 - 9: **end if**
 - 10: **end for**
 - 11: $\mathbf{w}^t \leftarrow \left(\sum_{i=1}^I \frac{\|\mathcal{D}_i\|}{\|\mathcal{D}\|} \hat{\mathbf{w}}_i^t \right) \odot \bar{\mathbf{m}}^t$;
 - 12: **end for**
 - function** $\text{UserUpdate}(\mathbf{w}^{t-1}, \mathbf{m}^{t-1}, i)$:
 - 13: **for** local epoch $e \leftarrow 1$ **to** E **do**
 - 14: Local Training and Base Pruning: $\bar{\mathbf{w}}_i^t \leftarrow \text{Eq. (4.1)}$
 - 15: Defense Pruning: $\hat{\mathbf{w}}_i^t \leftarrow \bar{\mathbf{w}}_i^t \odot \hat{\mathbf{m}}_i^t$
 - 16: **end for**
 - 17: **Return** $\hat{\mathbf{w}}_i^t$ to server
-

base pruning masks $\bar{\mathbf{m}}_i^t$ on its trained local model $F(\mathcal{D}_i, \mathbf{w}_i^t)$:

$$\bar{\mathbf{w}}_i^t := \left(\mathbf{w}^t - r \frac{\partial \ell_i(F(\mathcal{D}_i, \mathbf{w}^t \odot \bar{\mathbf{m}}^t), y_i)}{\partial \mathbf{w}^t} \right) \odot \bar{\mathbf{m}}_i^t \quad (4.1)$$

Where $\bar{\mathbf{m}}_i^t = \mathbb{P}_i(\mathbf{w}_i^t)$, r is the learning rate and $\odot$ is the Hadamard product. The *base* pruning mask $\bar{\mathbf{m}}_i^t$ exhibits variability across different base pruning schemes, reflecting the diverse pruning strategy $\mathbb{P}_i(\cdot)$ employed.

Server: The server receives model updates $\hat{\mathbf{w}}_i^t$ from all participating users and aggregates to update the global model, as it is typical in FL. We do not consider secure aggregation [193], differential privacy [196], or other defenses known in the literature.¹

Threat Model. We consider an *honest-but-curious* server, which correctly follows the FL protocol but uses model updates to infer private information, by launching privacy attacks against target user(s), as shown in Algorithm 6. In our case, the server has full visibility of all masked local models and, upon receiving an update from a target user, it attempts to reconstruct the user's training data. We consider reconstruction attacks that invert gradients, including Deep-Leakage-from-Gradients (DLG) [70], Gradient Inversion (GI) [37],

¹We consider those defenses as out-of-the-scope for this chapter, since they are orthogonal to pruning. They can be implemented on top/combined with our pruning-based defense methods.

and a custom attack SGI described in Section 4.4.2. We focus on single-round attacks, *i.e.*, attacks that invert gradients observed at a single round² leakage in Section 4.4.1. Designing a practical multi-round attack is orthogonal to our contribution, which focuses on quantifying the leakage and designing an add-on defense, for *existing* attacks.

Defense via Pruning: In this chapter, we propose to defend against privacy attacks, via pruning itself. One could specifically design an entire pruning scheme not only for accuracy and efficiency but also for privacy, and the defense would then be optimized jointly with the particular base pruning scheme *e.g.*, PruneFL [78] and FedDST [79].

In this chapter, we take a modular and universal approach: we consider *any* given state-of-the-art base pruning scheme (with rate $\bar{p}_i$ and mask $\bar{\mathbf{m}}_i^t$) without any modification. Then, we introduce additional local *defense* pruning, with mask $\hat{\mathbf{m}}_i^t$ and pruning rate $\hat{p}_i$, specifically designed for protecting each user i from privacy attacks. User i eventually sends back to the server weights after applying both base ($\bar{\mathbf{m}}_i^t$) and defense ($\hat{\mathbf{m}}_i^t$) pruning: $\hat{\mathbf{w}}_i^t = \bar{\mathbf{m}}_i^t \odot \hat{\mathbf{m}}_i^t \odot \mathbf{w}_i^t$. This is depicted in Fig. 4.1 and Algorithm 6 (see Line 15 and 16), and elaborated upon on Fig. 4.11. Given a base pruning ($\bar{\mathbf{m}}_i^t$), there are many possible defense pruning strategies ($\hat{\mathbf{m}}_i^t$). In Section 4.5.2, we propose *PriPrune* for $\hat{\mathbf{m}}_i^t$, to optimize the accuracy-vs-privacy tradeoff.

4.4. Quantifying Privacy In FL Model Pruning

In this section, to address question Q1, we provide a theoretical and empirical quantification of privacy leakage in FL model pruning. We aim to assess the server’s ability to infer information about an individual user’s local dataset $\mathcal{D}_i$. We employ Mutual Information (MI) as the metric for quantifying privacy leakage:

$$\begin{aligned} \mathbf{I}(X; Y) &= \int_{\mathcal{X} \times \mathcal{Y}} \log \frac{d\mathbb{P}_{X,Y}(x, y)}{d\mathbb{P}_X(x) \otimes \mathbb{P}_Y(y)} d\mathbb{P}_{X,Y}(x, y) \\ &= \mathbf{H}(X) + \mathbf{H}(Y) - \mathbf{H}(X, Y) \end{aligned} \quad (4.2)$$

Where $(X; Y)$ is a pair of random variables with values over the space $\mathcal{X} \times \mathcal{Y}$. Their joint distribution is $\mathbb{P}_{X,Y}(x, y)$ and the marginal distributions are $\mathbb{P}_X(x)$ and $\mathbb{P}_Y(y)$.

This concept, rooted in information theory and Shannon entropy, captures the interdependence between two random variables. It is very powerful for privacy analysis, given its applicability across various domains and tasks [198], [199]. This applicability extends across various datasets, pruning schemes, and attack scenarios, as recently demonstrated in the quantification of privacy within the context of FL aggregation [197].

²To the best of our knowledge, there are currently no known *practical* multi-round inverting-the-gradient types of attacks that combine information from multiple rounds. There is theoretical analysis of upper *bounds* of multi-round information leakage, including [192], [197] and we derive our own multi-round

4.4.1. Theoretical Quantification

We seek to quantify how much information the server can infer about the private data $\mathcal{D}_i$ of user i over T global training rounds, based on the pruned updates $\{\mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t\}_{t \in [T]}$ submitted by user i , via:

$$\mathbf{I}_i = \mathbf{I}(\mathcal{D}_i; \{\mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t\}_{t \in [T]}) \quad (4.3)$$

where $\mathbf{I}$ represents mutual information between $\mathcal{D}_i$ and $\{\mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t\}_{t \in [T]}$, $\mathbf{w}_i^t \in \mathbb{R}^d$ is the local model weights of user i at round t .

This quantification is based on two mild assumptions:

Assumption 4.4.1. The random vector $\hat{g}_i(\mathbf{w}_i^t, b)$ has non-singular covariance matrix Σ_i and mean 0.

Here, $\hat{g}_i(\mathbf{w}_i^t, b) \in \mathbb{R}^{d^*}$ is the largest sub-vector of the $g_i(\mathbf{w}_i^t, b)$, where d^* is the rank of $g_i(\mathbf{w}_i^t, b)$, $d^* \leq d$, d is the model size, b denotes the size of the random samples. $g_i(\mathbf{w}_i^t, b)$ denotes the stochastic estimate of the gradient of the local loss function for user i , computed based on a random sample b , and $\mathbf{B}_i^t$ is a data batch of size B sampled uniformly at random from it local dataset $\mathcal{D}_i$.

Assumption 4.4.2. $\bar{p}_i$ the pruning rate of user i remains unchanged throughout the training process.

Building upon Equation 4.2, Equation 4.3 can be written as:

$$\begin{aligned} \mathbf{I}(\mathcal{D}_i; \{\mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t\}_{t \in [T]}) &\stackrel{(a)}{\leq} \sum_{t=1}^T \mathbf{I}(\mathcal{D}_i; \mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) \\ &\stackrel{(b)}{=} \sum_{t=1}^T \left(\mathbf{I}(\mathbf{w}_i^t; \mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) \right) \\ &\quad + \sum_{t=1}^T \left(\mathbf{I}(\mathcal{D}_i; \mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t | \mathbf{w}_i^t, \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) \right) \\ &\quad - \sum_{t=1}^T \left(\mathbf{I}(\mathcal{D}_i; \mathbf{w}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t]}) \right) \\ &\stackrel{(c)}{\leq} \sum_{t=1}^T \left(\mathbf{I}(\mathbf{w}_i^t; \mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) \right) \\ &= \sum_{t=1}^T \left(\mathbf{H}(\mathbf{w}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) \right) + \sum_{t=1}^T \left(\mathbf{H}(\mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) \right) \\ &\quad - \sum_{t=1}^T \left(\mathbf{H}(\mathbf{w}_i^t, \mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) \right) \\ &\stackrel{(d)}{=} \sum_{t=1}^T \left(\mathbf{I}(\mathbf{x}_i^t; \mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) \right) \end{aligned} \quad (4.4)$$

Where (a) and (b) are from the chain rule; (c) is from data processing inequality $\mathcal{D}_i \rightarrow \mathbf{w}_i^t \rightarrow \mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t$ that $\mathbf{I}(\mathcal{D}_i; \mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t | \mathbf{w}_i^t, \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) = 0$; (d) is from zero conditional entropy and invariant of mutual information, where

$$\mathbf{x}_i^t = \frac{\partial \ell_i(\bar{\mathbf{w}}_i^{t-1}; \mathcal{D}_i)}{\partial \mathbf{w}} = \frac{1}{B} \sum_{b \in \mathbf{B}_i^t} g_i(\mathbf{w}_i^t, b) \quad (4.5)$$

To that end, we first characterize the privacy leakage for a *single round* t :

$$\mathbf{I}(\mathbf{x}_i^t; \mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) \quad (4.6)$$

Under these two mild assumptions, we obtain our main theoretical result.

Upper Bound for Privacy Leakage in a Single Round.

We now provide an upper bound for Equation 4.6, i.e., for the privacy leakage in a single round.

Theorem 4.4.3 (Single Round Leakage). *Under Assumption 4.4.1 and 4.4.2, we have an upper bound of $\mathbf{I}_i^t$ for a single round:*

$$\mathbf{I}_i^t \leq 1 - \frac{\bar{p}_i - 1}{2 \ln 2} + 2 \log \frac{1}{B} + 2\Delta + d^* \log(2\pi e) \quad (4.7)$$

where $\Delta = \log \left| \det(\Sigma_i^{-\frac{1}{2}}) \right|$.

Proof. Based on Equation 4.2, we have

$$\begin{aligned} \mathbf{I}(\mathbf{x}_i^t; \mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) &= \mathbf{H}(\mathbf{x}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) + \mathbf{H}(\mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) \\ &\quad - \mathbf{H}(\mathbf{x}_i^t, \mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) \\ &\leq \underbrace{\mathbf{H}(\mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]})}_{\mathbf{A}} + \underbrace{\mathbf{H}(\mathbf{x}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]})}_{\mathbf{B}} \end{aligned} \quad (4.8)$$

For part **A**, based on the chain rule of entropy, we have

$$\begin{aligned} \mathbf{H}(\mathbf{w}_i^t, \bar{\mathbf{m}}_i^t, \mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) &= \mathbf{H}(\mathbf{w}_i^t, \bar{\mathbf{m}}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) \\ &\quad + \mathbf{H}(\mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t | \mathbf{w}_i^t, \bar{\mathbf{m}}_i^t, \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) \\ &= \mathbf{H}(\mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) \\ &\quad + \mathbf{H}(\mathbf{w}_i^t, \bar{\mathbf{m}}_i^t | \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}) \geq 0 \end{aligned} \quad (4.9)$$

Due to $\mathbf{H}\left(\mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t \middle| \mathbf{w}_i^t, \bar{\mathbf{m}}_i^t, \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}\right) = 0$, and $\mathbf{H}\left(\mathbf{w}_i^t, \bar{\mathbf{m}}_i^t \middle| \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t]}\right) \geq 0$, we have

$$\begin{aligned}
\mathbf{A} &\stackrel{(a)}{\leq} \mathbf{H}\left(\mathbf{w}_i^t \middle| \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}\right) + \mathbf{H}\left(\bar{\mathbf{m}}_i^t \middle| \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}\right) \\
&\stackrel{(b)}{=} \mathbf{H}\left(\mathbf{x}_i^t \middle| \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}\right) + 1 - \frac{1}{2 \ln 2} \sum_{n=1}^{\infty} \frac{(2\bar{p}_i - 1)^{2n}}{n(2n-1)} \\
&\leq \underbrace{\mathbf{H}\left(\mathbf{x}_i^t \middle| \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}\right)}_{\mathbf{B}} + 1 - \frac{\bar{p}_i - 1}{2 \ln 2}
\end{aligned} \tag{4.10}$$

(a) is from conditioning reduces entropy; (b) is from zero conditional entropy and the Taylor series of the binary entropy function in a neighbourhood of 0.5 with the base pruning rate $\bar{p}_i$.

For part **B**,

$$\mathbf{B} = \mathbf{H}\left(\frac{1}{B} \sum_{b \in \mathbf{B}_i'} g_i(\mathbf{w}_i^t, b) \middle| \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}\right) \tag{4.11}$$

Let $X = \sum_{b \in \mathbf{B}_i'} g_i(\mathbf{w}_i^t, b)$, $Z = \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}$, $Y = \frac{1}{B}X$. $f_{X,Z}(x, z)$ and $f_{Y,Z}(y, z)$ are the corresponding joint probability density functions.

Then part **B** can be written as

$$\begin{aligned}
\mathbf{H}(Y|Z) &= - \int_{\mathcal{Y}, \mathcal{Z}} f_{Y,Z}(y, z) \log \frac{f_{Y,Z}(y, z)}{f_Z(z)} dy dz \\
&= - \int_{\mathcal{X}, \mathcal{Z}} |B| f_{X,Z}(By, z) \log |B| \frac{f_{X,Z}(By, z)}{f_Z(z)} dy dz \\
&= - \int_{\mathcal{X}, \mathcal{Z}} f_{X,Z}(x, z) \log \frac{f_{X,Z}(x, z)}{f_Z(z)} dx dz + \log \frac{1}{|B|} \\
&= \underbrace{\mathbf{H}\left(\sum_{b \in \mathbf{B}_i'} g_i(\mathbf{w}_i^t, b) \middle| \{\mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k\}_{k \in [t-1]}\right)}_{\mathbf{C}} + \log \frac{1}{B}
\end{aligned}$$

In recent theoretical results for analyzing the behaviour of SGD, the SGD vector is approximated by a distribution with independent components or by a multivariate Gaussian vector [200].

Based on the ZCA whitening transformation and Assumption 4.4.1, we have $\hat{g}_i(\mathbf{w}_i^t, b) = \Sigma_i^{-\frac{1}{2}} \mathbf{z}$, where $\mathbf{z}$ has zero mean and $\mathbb{I}_{d^*}$ covariance matrix.

For the part **C**, we set $\mathbf{v} = \sum_{b \in \mathbf{B}_i^t} \mathbf{z}$ and $\mathbf{u} = \Sigma_i^{-\frac{1}{2}} \mathbf{v}$, then we have:

$$\begin{aligned}
\mathbf{C} &= \mathbf{H} \left(\sum_{b \in \mathbf{B}_i^t} \hat{g}_i(\mathbf{w}_i^t, b) \left| \left\{ \mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k \right\}_{k \in [t-1]} \right| \right) = \mathbf{H} \left(\Sigma_i^{-\frac{1}{2}} \sum_{b \in \mathbf{B}_i^t} \mathbf{z} \left| \left\{ \mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k \right\}_{k \in [t-1]} \right| \right) \\
&= - \int \mathbb{P}_{\mathcal{U}}(\mathbf{u}) \log \mathbb{P}_{\mathcal{U}}(\mathbf{u}) d\mathbf{u} \\
&\stackrel{(a)}{=} - \int \frac{\mathbb{P}_{\mathcal{V}}(\mathbf{v})}{\left| \det(\Sigma_i^{-\frac{1}{2}}) \right|} \log \left(\frac{\mathbb{P}_{\mathcal{V}}(\mathbf{v})}{\left| \det(\Sigma_i^{-\frac{1}{2}}) \right|} \right) \left| \det(\Sigma_i^{-\frac{1}{2}}) \right| d\mathbf{v} \\
&= \log \left| \det(\Sigma_i^{-\frac{1}{2}}) \right| + \underbrace{\mathbf{H} \left(\sum_{b \in \mathbf{B}_i^t} \mathbf{z} \left| \left\{ \mathbf{w}_i^k \odot \bar{\mathbf{m}}_i^k \right\}_{k \in [t-1]} \right| \right)}_{\mathbf{D}}
\end{aligned} \tag{4.12}$$

where (a) is based on the transformation of random vectors. Based on the Maximum Entropy Upper Bound [201], which the continuous distribution $\mathbf{X}$ with prescribed variance $\mathcal{V}(\mathbf{X})$ maximizing the entropy is the Gaussian distribution of same variance. Since the entropy of a multivariate Gaussian distribution with mean μ and covariance $K_{i,j}$ is

$$\mathbf{h}(\mathcal{N}_n(\mu, K)) = \frac{1}{2} \log (2\pi e)^n |K|$$

where $|K|$ denotes the determinant of K . Hence, the maximum entropy upper bound for part **D** is

$$\mathbf{D} \leq \frac{1}{2} \log (2\pi e)^{d^*} |\mathbb{I}_{d^*}| = \frac{d^*}{2} \log (2\pi e) \tag{4.13}$$

Therefore, by combining Equation 4.12 and Equation 4.13, the upper bound for part **B** is

$$\mathbf{B} \leq \log \frac{1}{B} + \log \left| \det(\Sigma_i^{-\frac{1}{2}}) \right| + \frac{d^*}{2} \log (2\pi e) \tag{4.14}$$

By adding parts **A** and **B**, we establish the upper bound for privacy leakage for a single round as in Theorem 4.4.3.

Remark 4.4.4. The upper bounds on information leakage decrease as the pruning ratio p increases. Intuitively, higher pruning ratios result in a more substantial reduction in model parameters, leading to a decreased risk of sensitive information being retained in the pruned model.

Remark 4.4.5. Increasing the batch size B contributes to reducing information leakage in the user's local training model. Larger batch sizes enable more data points to be added during model training, which can help obscure single data characteristics and increase the privacy protection of the FL process.

Remark 4.4.6. The upper bounds on information leakage increase as d^* increases. Interestingly, the impact of model size d on information leakage is not linear. In the case of overparameterized models, some parameters may not significantly contribute to the model’s performance or information retention. Therefore, increasing the size of such models may not proportionally affect information leakage.

Upper Bound for Privacy Leakage across Multiple Rounds

Using Equation 4.4, we can also obtain an upper bound for Equation 4.3, i.e., how much information the aggregated model over T global training rounds could leak about the private data:

Corollary 4.4.7 (Multiple Rounds Leakage). *Continuing with Theorem 4.4.3, we can upper bound $\mathbf{I}_i$ after T global training rounds as follows:*

$$\mathbf{I}_i \leq T \left(1 - \frac{\bar{p}_i - 1}{2 \ln 2} + 2 \log \frac{1}{B} + 2\Delta + d^* \log(2\pi e) \right) \quad (4.15)$$

Remark 4.4.8. This shows that increasing the number of global training rounds (T) leads to a proportional rise in the upper bound of information leakage from the user’s local training model, occurring at a rate of $\mathcal{O}(T)$. As the training progresses over multiple rounds, the user’s model updates are repeatedly exposed to the central server. This repeated exposure increases the risk of potential memorization of private training information by the server.

Importance of the Upper Bounds: The derived upper bounds are important for several reasons.

- They provide a theoretical upper limit to the amount of information that can potentially be leaked from model updates in pruned models. To the best of our knowledge, this has not been previously characterized for pruned models.
- Because they are information-theoretical in nature, these bounds are universally applicable. They apply to *any* learning task, *any* data distribution, *any* pruning scheme, and *any* privacy attack. The attacker is the server that sees the pruned updates $\{\mathbf{w}_i^t \odot \bar{\mathbf{m}}_i^t\}_{t \in [T]}$ from client i , and infers information about the local training Data $\mathcal{D}_i$ of that client. Regardless of the exact reconstruction strategy, the inferred information is at most the mutual information $\mathbf{I}_i$ between the update and the local data, quantified in Equation 4.3. The derivation of the bounds assumes the existence of a pruning mask but does not depend on how this mask was generated, *e.g.*, whether through magnitude-based pruning, gradient-based pruning, or any other pruning method.

- Theorem 4.4.3 expresses the bound in terms of important parameters (e.g., the pruning rate p , model size d , etc), which provides insights into the choices and parameters that affect the privacy provided by pruning in practical algorithms. However, the bounds may or may not be tight compared to the actual information leakage, depending on the real-world scenario. In fact, these bounds are necessarily conservative since they are universal. This is why, in the next section, we complement the theoretical analysis with experimental evaluations in specific settings (i.e., considering specific base pruning, attack and defense schemes, specific data sets, and learning tasks).

4.4.2. Empirical Quantification

We perform an experimental evaluation to quantify the exact amount of privacy loss in concrete settings. This allows us to compare the theoretical bounds to experimental results and show that they are qualitatively aligned. It also allows us to obtain insights (in Section 4.5.1) that inform the design of defense pruning (in Section 4.5.2). More evaluation results are provided in Section 4.6.

Experimental Setup

Datasets and Models. We evaluate models Conv-2 [163] and VGG-11 [202]) along with their corresponding datasets FEMNIST [163] and CIFAR-10 [164]), which are commonly employed in FL studies.

Data Partitioning Our data partitioning methodology is aligned with the FEMNIST setting, utilizing its inherent 193-user partition. As for the CIFAR-10 dataset, we divided it into 100 equal-sized, non-overlapping users, following the methodology used in PruneFL [78].

Implementation details. In each round of training, we randomly select 10 clients from a pool of 193 users for the FEMNIST and 100 clients for the CIFAR-10. The training process involves 20,000 iterations, with a batch size of 1 for the CIFAR-10. Across all the examined methods, we employ 1 local training and initialize the learning rate to 0.25 and The baseline pruning rate is set at 0.3 for all datasets. Our algorithms are implemented by Pytorch, and we implement experiments on two NVIDIA RTX A5000 and two Xeon Silver 4316.

Privacy Metrics.

To quantify the extent of privacy leakage, we use the Normalized Mutual Information (NMI) between the training data and the reconstructed data. For two vectors $U, V \in \mathcal{R}^N$, NMI is computed as follows. The entropy of U is calculated by $H(U) = - \sum_{i=1}^{|U|} P(i) \log(P(i))$,

where $P(i) = |U_i|/N$. The Mutual Information between U and V is given by $I(U; V) = \sum_{i=1}^{|U|} \sum_{j=1}^{|V|} \frac{|U_i \cap V_j|}{N} \log \left(\frac{N|U_i \cap V_j|}{|U_i||V_j|} \right)$. NMI is then measured as $\frac{I(U; V)}{\text{mean}(H(U), H(V))}$.

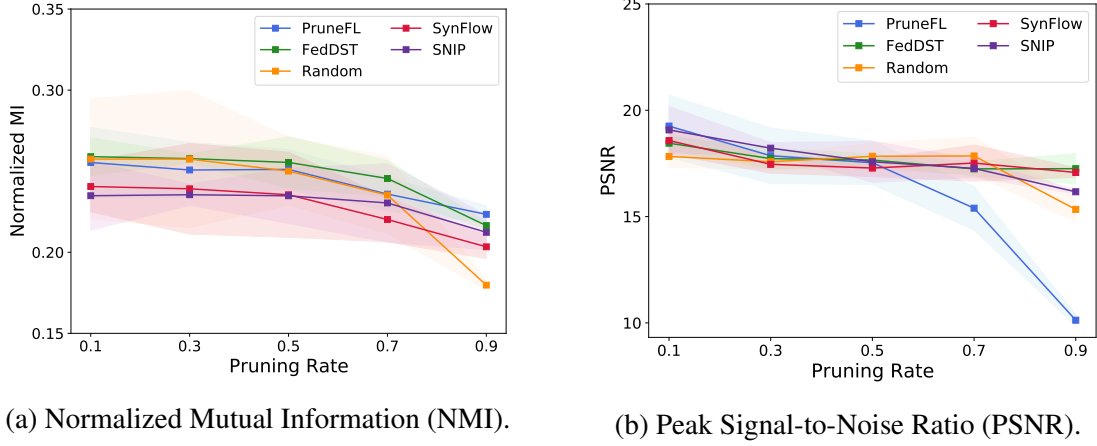


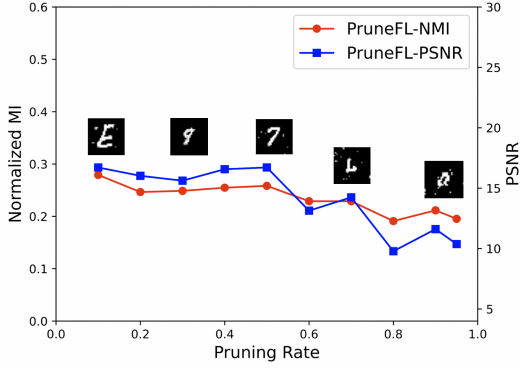
Figure 4.2: Impact of varying pruning rate on privacy leakage using key metrics: Normalized Mutual Information (NMI) displayed on (a) and Peak Signal-to-Noise Ratio (PSNR) shown on (b) under a Sparse Gradient Inversion attack (SGI) in FEMNIST for five pruning methods under varying the base pruning rate.

For image data, prior research [197] has shown that NMI aligns well with the Peak Signal-to-Noise Ratio (PSNR) [197] as an effective measure of privacy leakage. This is also the case here: we examine the relationship between NMI and PSNR and we see an agreement between NMI and PSNR with increasing pruning rates in Fig. 4.2 and Fig. 4.3a, both showing a similar decreasing trend. This result is consistent with previous research, confirming the reliability of NMI as a measure of privacy leakage, which is comparable to the widely accepted PSNR metric. In the rest of the paper, we use both NMI and PSNR as metrics to quantify the success of a reconstruction attack. Intuitively, the higher the NMI, the higher the privacy leakage, the more successful the reconstruction attack. The higher the PSNR, the closer the original and reconstructed images and the more successful the attack. Hence, Fig. 4.2 and Fig. 4.3a compare the NMI and PSNR metrics and show that increasing pruning rate reduces the success of the privacy attack.

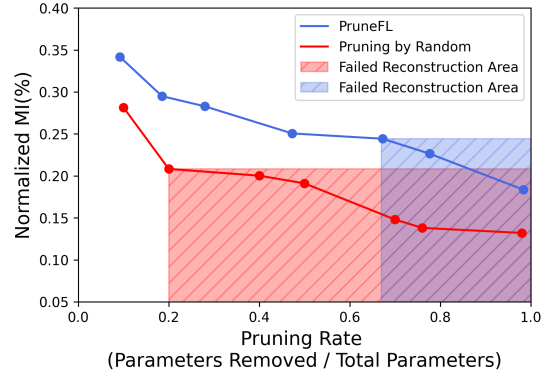
Reconstruction Attack Algorithms

We initiate our attack analysis by employing the classic Gradient Inversion (GI) attack [37], a member of the DLG attack family, applied to both the Random and PruneFL pruning methods. As depicted in Figure 4.3b, under this classic DLG attack, the reconstruction of local image data fails even with low pruning rates, such as $p = 0.2$ for Random pruning.

This could be attributed to the sparsity patterns introduced by model pruning in FL. Our exploration takes a step further to optimize the attack strategy. The goal is to capitalize on the characteristics of sparsity inherent to the model pruning context. Consequently, we introduce an advanced attack tailored for the context of model pruning in FL, the



(a) Impact on privacy leakage using NMI (left y-axis) and PSNR (right y-axis) under a Gradient Inversion attack. The ground truth images, from left to right, correspond to the characters: E, 9, 7, L, and a.



(b) Impact of varying pruning rates on reconstructed image quality using NMI metric under DLG attack.

Figure 4.3: Impact of varying pruning rates on privacy leakage for PruneFL in FEMNIST under Gradient Inversion (left) and DLG (right) attacks.

Sparse Gradient Inversion (SGI) attack. This attack is designed to recover the pruning mask within the pruned model. Moreover, its adaptability allows seamless integration with existing privacy attacks in FL, thereby amplifying their effectiveness.

Algorithm 7: Sparse Gradient Inversion (SGI) Attack

- 1: **Input:** $F(\mathcal{D}_i; \hat{\mathbf{w}}_i^t)$: model at round t from targeted user i ; learning rate η for inverting gradient optimizer; S : max iterations for attack; τ : regularization term for cosine loss in inverting gradient attack; $\mathbf{m}_{rec}$: the recovery mask generated by server; d : the model size
 - 2: **Output:** reconstructed training data $(\mathcal{D}_i, y_i)$ at round t

$$\mathbf{m}_{rec}[j]_{j \in d} = \begin{cases} 1 & \text{if } \hat{\mathbf{w}}_i^t[j] \neq 0 \\ 0 & \text{if } \hat{\mathbf{w}}_i^t[j] = 0 \end{cases}$$
 - 3: $\nabla \hat{\mathbf{w}}_i^t \leftarrow (\hat{\mathbf{w}}_i^t - \hat{\mathbf{w}}_i^{t-1}) \odot \mathbf{m}_{rec}$
 - 4: **Initialize** $\mathcal{D}'_0 \leftarrow \mathcal{N}(0, 1)$, $y'_0 \leftarrow \text{Randint}(0, \max(y))$
 - 5: **for** $s \leftarrow 0$ **to** $S - 1$ **do**
 - 6: $\nabla \mathbf{w}'_s \leftarrow \partial \ell(F(\mathcal{D}'_s, \hat{\mathbf{w}}_i^t), y'_s) / \partial \hat{\mathbf{w}}_i^t$
 - 7: $\mathcal{L}'_s \leftarrow 1 - \frac{\nabla \hat{\mathbf{w}}_i^t \cdot \nabla \mathbf{w}'_s}{\|\nabla \hat{\mathbf{w}}_i^t\| \|\nabla \mathbf{w}'_s\|} + \tau$
 - 8: $\mathcal{D}'_{s+1} \leftarrow \mathcal{D}'_s - \eta \nabla_{\mathcal{D}'_s} \mathcal{L}'_s$, $y'_{s+1} \leftarrow y'_s - \eta \nabla_{y'_s} \mathcal{L}'_s$
 - 9: **end for**
 - 10: **Return** $\mathcal{D}'_S, y'_S$
-

The SGI attack is provided in Algorithm 7. In each iteration t , the SGI algorithm proceeds as follows:

- i Based on the user i ' local model weights $\mathbf{w}_i^t$, the server attempts to recover the

pruning mask of user i (Line 2). The recovered mask is then integrated into the calculation of gradient, allowing the server to obtain a gradient $\nabla \hat{\mathbf{w}}'_i$ that closely approximates the true gradients of user i ;

- ii The SGI attack randomly initializes a set of dummy data, comprising dummy inputs $\mathcal{D}'_0$ and dummy labels y'_0 .
- iii After the dummy gradient $\mathbf{w}'$ is acquired, the server then updates the dummy data in the direction that minimizes the cosine distance between the dummy gradient and the gradient $\nabla \hat{\mathbf{w}}'_i$.

Base Pruning Schemes under Attack.

We subject a set of well-established FL pruning methods to the privacy attack to assess their vulnerabilities and effectiveness in protecting user privacy. The pruning methods under evaluation include Random pruning, Snip [75], SynFlow [76], GraSP [77], FedDST [79], and PruneFL [78], all of which are discussed in Related Work (Section 4.2). We present an overview of the pruning criteria, cycles, and schedules employed by the six evaluated methods. These specifics are elaborated in Table 4.1. By scrutinizing the table, we discern notable trends: three of the methods execute pruning exclusively at the server side, while the remaining methods engage in pruning activities at both the server and client sides. The latter approaches adopt an iterative pruning strategy throughout the course of the FL process.

Method	Pruning Criteria	Pruning Cycles	Pruning at Server	Pruning at Client
SNIP [75]	gradient-based	one shot	✓	✗
SynFlow [76]	gradient-based	one shot	✓	✗
GraSP [77]	gradient-based	one shot	✓	✗
Random Pruning	magnitude-based	iterative	✓	✓
FedDST [79]	magnitude-based	iterative	✓	✓
PruneFL [78]	magnitude-based	iterative	✓	✓

Table 4.1: Comparison for Evaluated Methods

Fig. 4.4 shows the recovered images obtained under the Sparse Gradient Inversion (SGI) attack on the evaluated pruning methods with varying pruning rates, using the FEM-NIST dataset. Notably, there is no significant difference in the quality of the recovered images across different pruning methods. Especially, when the pruning rate exceeds 0.9, SGI fails to reconstruct any images from any of the pruning methods.

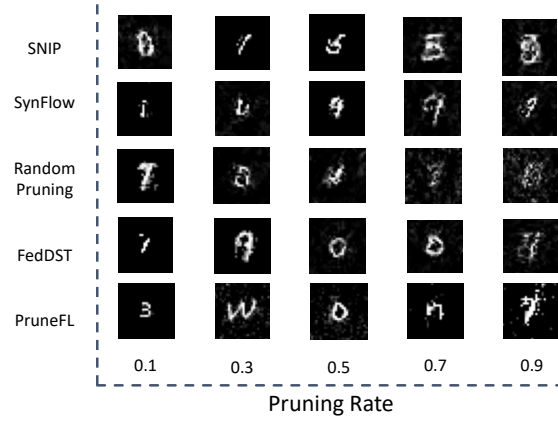


Figure 4.4: Impact of varying pruning rate on image reconstruction in FEMNIST using different base pruning methods under SGI attack.

Comparison of Privacy Attacks.

We compare the effectiveness of the Sparse Gradient Inversion (SGI) attack with the Gradient Inversion attack under varying the base pruning rate. Our evaluation is based on the normalized mutual information (NMI) metric. Specifically, we analyze the NMI achieved by each attack for different pruning methods. Figure 4.5 demonstrates that the SGI attack consistently outperforms the Gradient Inversion attack by inferring a larger amount of information (as captured by NMI) at each pruning rate level, across all tested pruning methods. This is because the SGI attack exploits the sparsity inherent to model pruning, ultimately allowing better reconstruction attacks. In the rest of the paper, we use the SGI attack as a baseline, with a learning rate of 0.01 and 10,000 attack iterations.

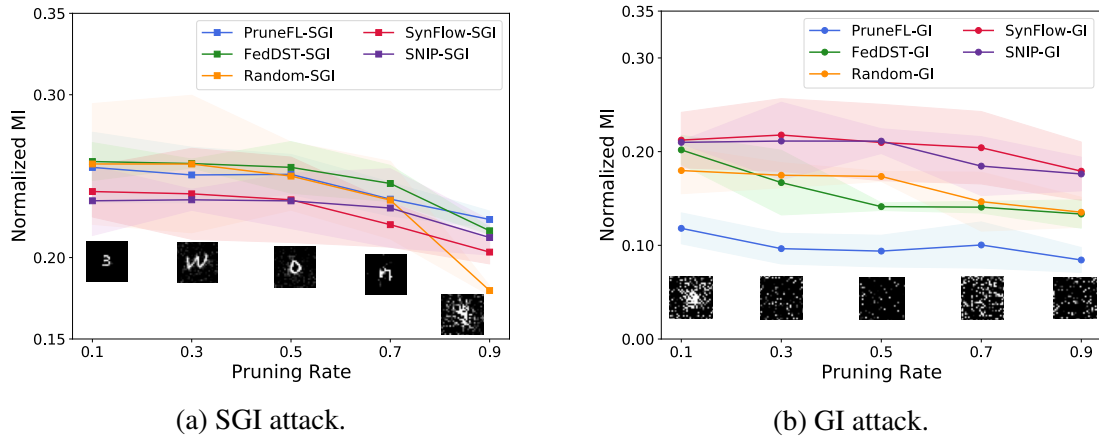


Figure 4.5: Comparison between the SGI (Sparse Gradient Inversion) attack and the GI (Gradient Inversion) attack on the FEMNIST dataset for five pruning methods under varying the base pruning rate. In addition to the NMI metric, we show the corresponding recovered images, which show the degradation with an increasing pruning rate.

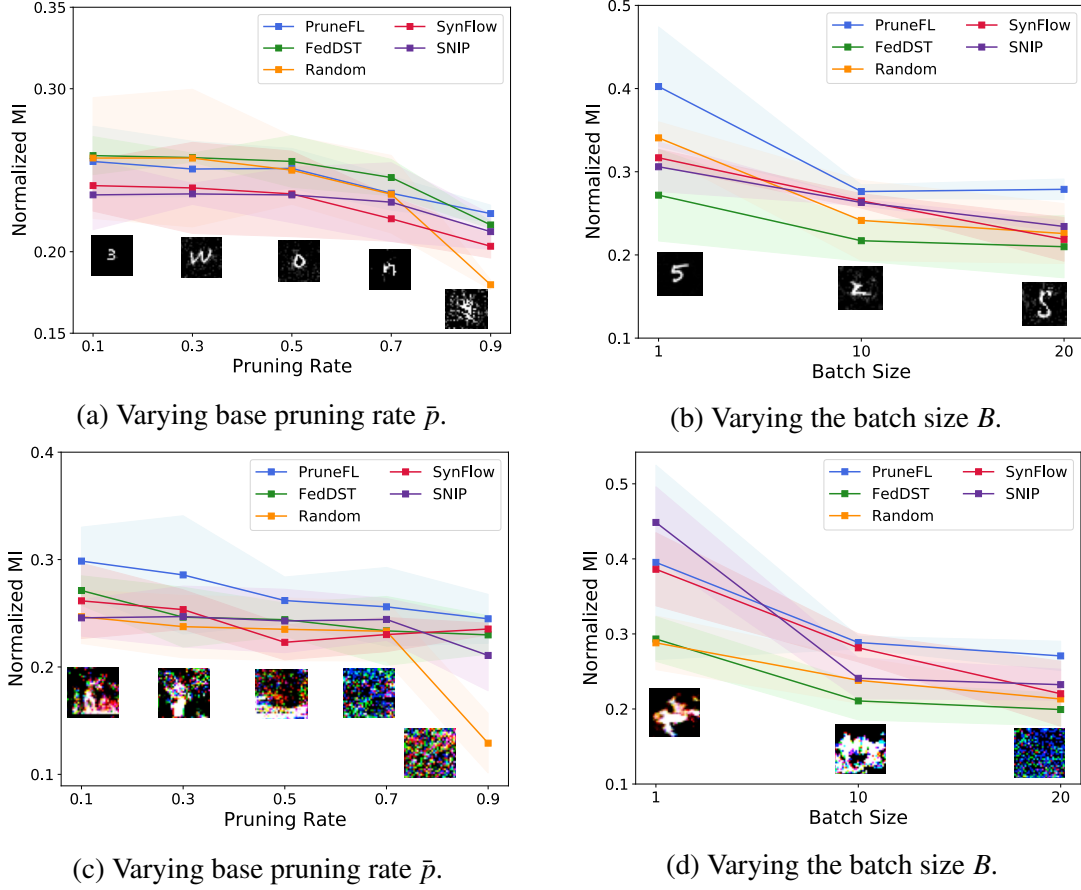


Figure 4.6: Impact of varying base pruning rate $\bar{p}$ and batch size B on privacy leakage. We consider the Normalized Mutual Information (NMI) as a privacy metric. We launch a Sparse Gradient Inversion (SGI) attack against 5 base pruning methods (Random, SNIP, SynFlow, FedDST, and PruneFL), over two datasets (FEMNIST in the First Row and CIFAR10 in the (Second Row)).

Attack Performance

We launch the SGI attack on the aforementioned FL pruning schemes, on the FEMNIST and CIFAR10 datasets, to assess how FL parameters influence privacy leakage. The experimental results in Fig. 4.6 and Fig. 4.7 demonstrate that the impact of the parameters $\bar{p}$, B , d , and T qualitatively align with our theoretical findings.

Impact of Pruning Rate ($\bar{p}$). Fig. 4.6a and Fig. 4.6c illustrate that a higher pruning rate generally leads to a decrease in NMI. This is intuitively expected as pruning removes information and makes it more difficult to attack the original model. The effect is clearer for large pruning rates (*e.g.*, above 0.5). Image reconstruction is clearer in FEMNIST than CIFAR, consistently with prior work.

Impact of Batch Size (B). Fig. 4.6b and Fig. 4.6d show increasing the batch size B contributes to reducing information leakage from the local training model. Larger batch sizes enable more data to be added during model training, which can help obscure single

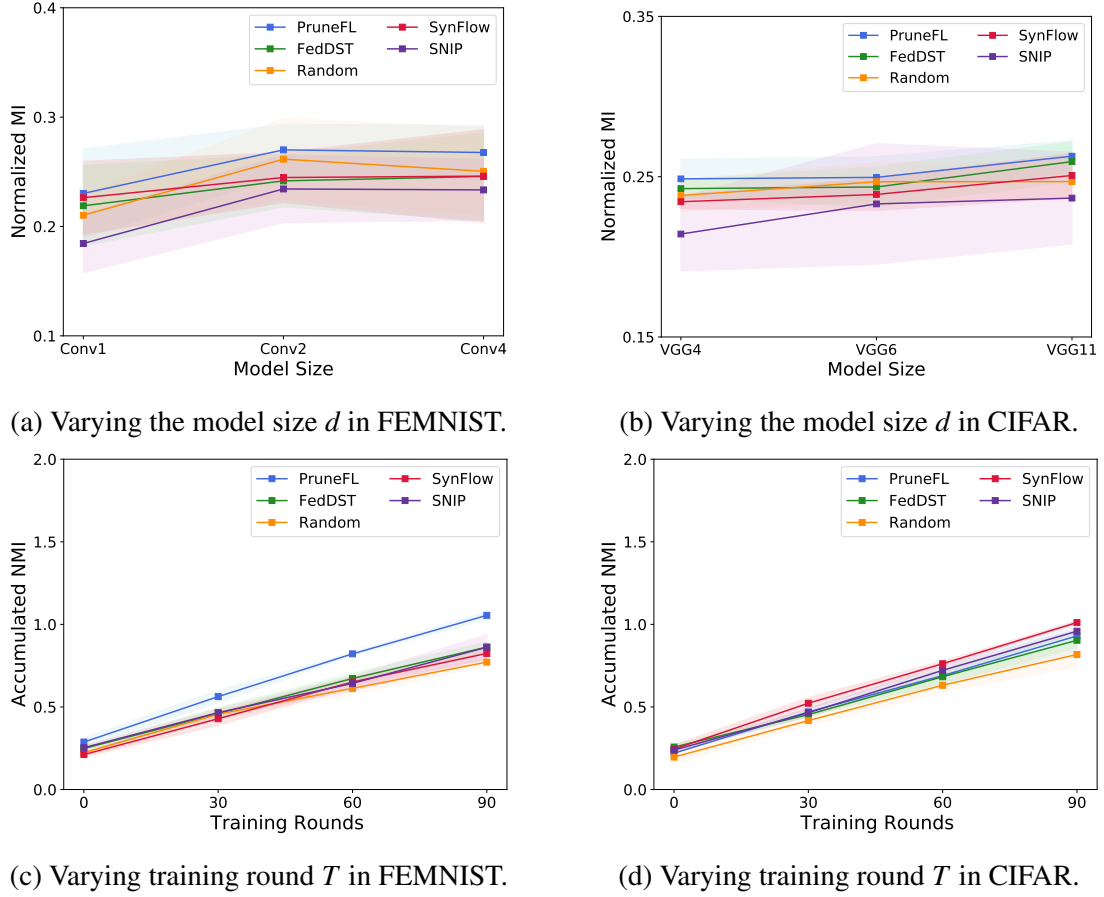


Figure 4.7: Impact of varying model size d and training rounds T on privacy leakage using Normalized Mutual Information (NMI) under a Sparse Gradient Inversion (SGI) attack for six pruning methods: Random, SNIP, SynFlow, GraSP, FedDST, and PruneFL in FEMNIST dataset (First Row) and CIFAR10 dataset (Second Row).

data and increase privacy protection.

Impact of Model Size (d). We consider 3 architectures of the models for FEMNIST dataset: Conv-1, Conv-2, Conv-4, featuring 1, 2, and 4 convolutional layers, along with 2 linear layers. These models encompass 626720, 6601504 and 13077392 parameters, respectively. We consider 3 different model architectures for CIFAR 10 dataset: VGG-4, VGG-6, and VGG-11. These models consist of 1 convolutional layer and 2 linear layers, 4 convolutional layers and 2 linear layers, and 8 convolutional layers with 3 linear layers, respectively. These models encompass 268464, 624048 and 9747136 parameters, respectively. Figure 4.7a and Figure 4.7b depicts the relationship between the model size (d) and the corresponding information leakage. The x-axis in the figure represents the number of layers in each model. Fig. 4.7a and Fig. 4.7b show the upper bounds on information leakage increase as d^* increases. Interestingly, the impact of model size d on information leakage is not linear. In the case of over-parameterised models, some parameters may not contribute to the model’s performance or information retention. Therefore, increasing the size of such models may not proportionally affect information leakage.

Impact of Global Training Rounds (T). Fig. 4.7c and Fig. 4.7d show that increasing the global training rounds (T) leads to a proportional rise in the upper bound of information leakage. As the training progresses over multiple rounds, the user’s model updates are repeatedly exposed to the central server. This repeated exposure increases the risk of potential memorization of private training information by the server.

4.5. Design of Defense Pruning

4.5.1. Insights from Privacy Quantification

As shown above, pruning designed only with model accuracy and model size in mind, does not sufficiently protect against privacy attacks. Next, we describe three insights gained in the process of privacy quantification that inform our design of privacy-aware FL model pruning (*PriPrune* described in Section 4.5.2).

Insight 1: Largest Weights Matter. Pruning weights with large gradients improves privacy the most, but also hurts model accuracy the most. The intuition is that (base) pruning criteria, as used *e.g.*, in *PruneFL* and *FedDST*, avoid pruning model weights with the largest gradients, in order to preserve the most valuable information for the model. These gradients can then be exploited by gradient inversion attacks. Fig. 4.6a, Fig. 4.6c and Fig. 4.4 confirm empirically that, as more weights with large gradients are pruned, the attack is less successful.

This implies that the defense should strategically prune certain weights with large gradients, in addition to those pruned by the base pruning scheme. We explored three defense strategies for weights to prune: (1) weights with the top- k largest gradients (denoted as *Largest*), (2) random weight pruning (denoted as *Random*), and (3) a hybrid approach combining the *Largest* and *Random* (denoted as *Mix*). We compared the three strategies for the same defense pruning rate $\hat{p}_i$ (set 0.3) on top of the base method (*PruneFL*). The results in Fig. 4.8, within the Pruning oval, show the tradeoff achieved between privacy and accuracy: the *Largest* method achieves the best privacy and the worst model accuracy.

Additionally, we explore various combinations of the pruning rates for *Largest* and *Random* within the *Mix* strategy. The combinations include 0% largest with 30% random, 10% largest with 20% random, 20% largest with 10% random, and 30% largest with 0% random. The results, depicted in Fig. 4.9, show the configuration of 0% largest with 30% random attains the highest accuracy, while 30% largest with 0% random yields the highest level of privacy protection.

Since we focus on privacy in this work, we first proceed to investigate the privacy-accuracy trade-off for the 30% largest with 0% random configurations. Figure 4.10 shows privacy-accuracy tradeoff with varying defense rate (0% to 50%) in *Largest* method based on *PruneFL* using NMI and attack loss. The outcomes are indicative of a notable trend: as more weights associated with larger gradients are pruned, privacy is enhanced at the

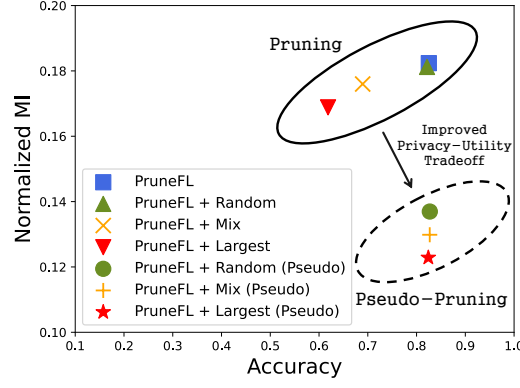


Figure 4.8: Tradeoff of privacy vs. accuracy with three defense strategies: *Largest* (red), *Random* (green), and *Mix* (yellow), and their enhanced versions incorporating *Pseudo-Pruning* (*Pseudo*). *Pseudo* could improve the overall tradeoff by elevating model accuracy and *Largest* offers the highest level of privacy protection among all defense methods.

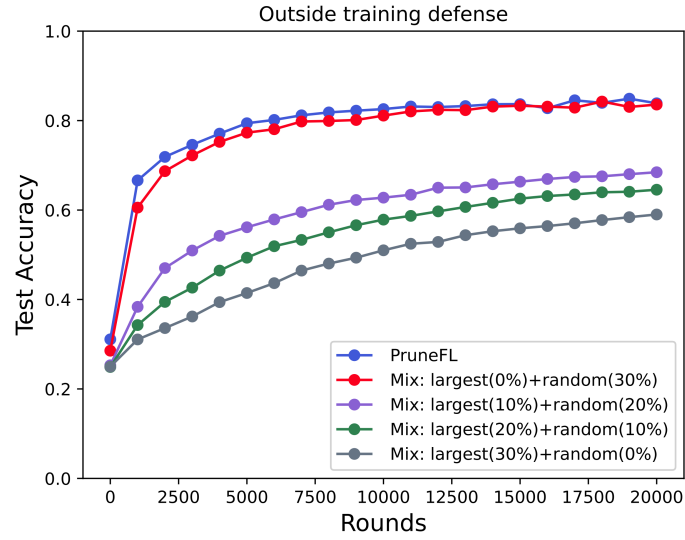
expense of accuracy.

These results substantiate our **Insight 1** that pruning weights with large gradients removes valuable information, thus improving privacy but also significantly reducing model accuracy. Therefore, to improve this tradeoff and preserve privacy without harming accuracy, we need to introduce an additional mechanism.

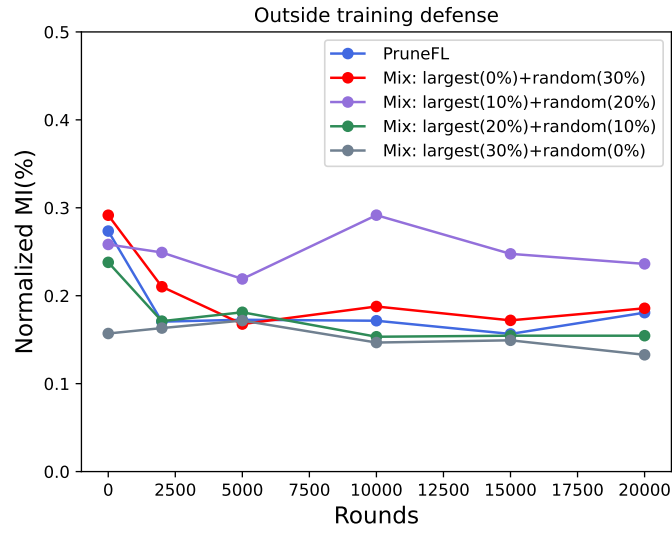
Insight 2: *Pseudo-Pruning*. Users can prune weights with large magnitude gradients when they communicate with the server (which helps privacy) *and* still keep these weights for their local training (which maintains accuracy).

We refer to this idea as *Pseudo-Pruning*, and illustrate it in Fig. 4.11. Before sending the weights $\tilde{\mathbf{w}}_i^t$ to the server user i conducts *Pseudo-Pruning* with defense mask $\hat{\mathbf{m}}_i^t$. The weights pruned out by defense pruning with large gradients $\tilde{\mathbf{w}}_i^t (= \tilde{\mathbf{w}}_i^t \odot \hat{\mathbf{m}}_i^t)$ are withheld locally, thus preserving accuracy. Meanwhile, the weights remaining after the defense pruning without large gradients, $\hat{\mathbf{w}}_i^t (= \tilde{\mathbf{w}}_i^t \odot \tilde{\mathbf{m}}_i^t)$, are transmitted to the server, thus preserving privacy. Here, $\tilde{\mathbf{m}}$ is the bit-wise complement matrix of $\hat{\mathbf{m}}$. In the next round, after user i receives the global model $\mathbf{w}^{t+1}$, the locally saved weights $\tilde{\mathbf{w}}_i^t$ are loaded into the global model, serving as the local initial model.

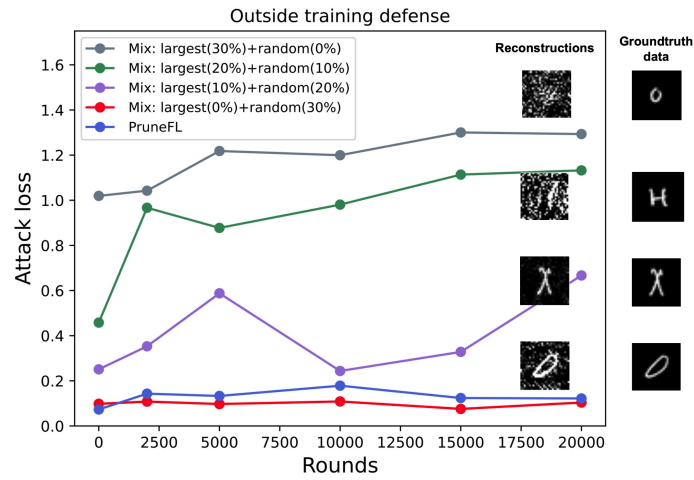
We combine *Pseudo-Pruning* with each of the three strategies (*Largest*, *Random*, and *Mix*) in **Insight 1**, and we show the evaluation results within the *Pseudo-Pruning* dashed oval in Fig. 4.8. Fig. 4.8 shows that the model accuracy is notably better with *Pseudo-Pruning* than real pruning, with the same mask. This advantage stems from saving and updating certain weights with large gradients locally, which significantly impacts model performance. In *Pseudo-Pruning*, these non-shared weights for each user are reloaded into the global model in the next round, ensuring the preservation of the local model’s utility. Additionally, not sharing these weights with the server, similar to real defense pruning, maintains the same privacy level and improves communication efficiency. Furthermore, combining the two insights into *Largest (Pseudo)* offers the most effective pri-



(a) Accuracy Performance

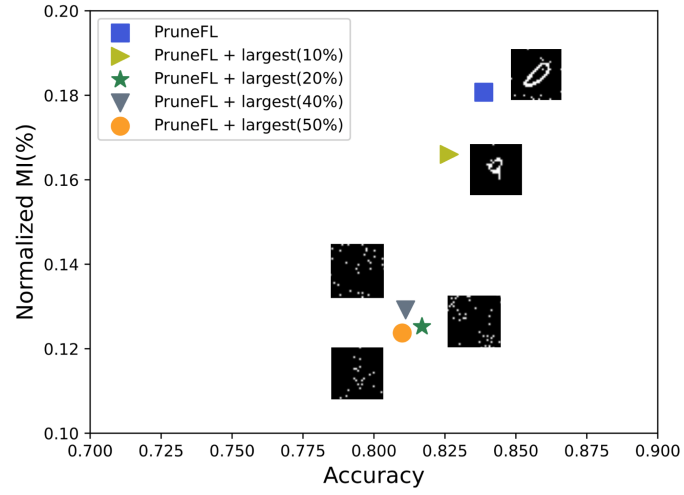


(b) NMI Performance

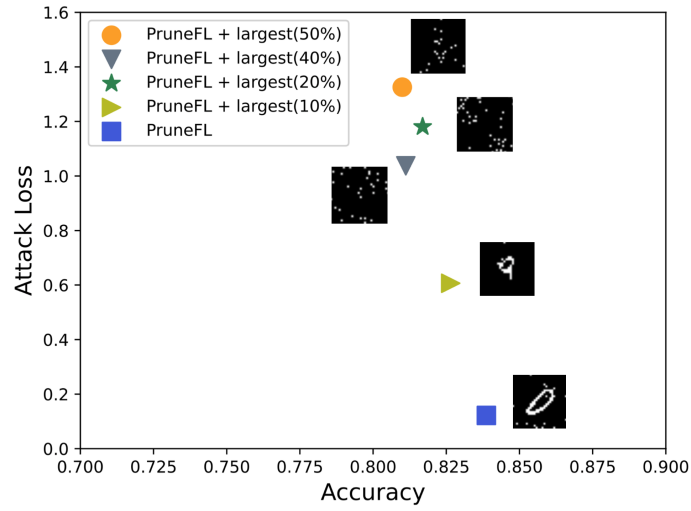


(c) Attack Loss Performance

Figure 4.9: Comparative performance of four defense rate combinations in the MIX strategy over 20,000 FL rounds, evaluated in terms of accuracy, NMI, and attack loss.



(a) ACC vs NMI



(b) ACC vs Attack Loss

Figure 4.10: Comparison of Privacy-Accuracy Trade-Off with Varying Defense Rate (0% to 50%) in Largest Method based on PruneFL using NMI and Attack Loss Privacy Metrics

vacy guarantee among all defenses discussed above. Hence, we adopt *Largest (Pseudo)* as our defense strategy. Still, a remaining shortcoming of the defense considered so far, is that their pruning rate is manually selected and remains fixed.

Insight 3: Adapt. The defense pruning rate ($\hat{p}_i$) should adapt to the model weights, to jointly optimize privacy and model accuracy. A fixed defense rate is unable to capture the evolving dynamics of the training process. The defense strategy should adapt to the changing dynamics of the training process by employing an adaptive defense pruning rate, enabling a more effective optimization of the tradeoff between privacy and accuracy.

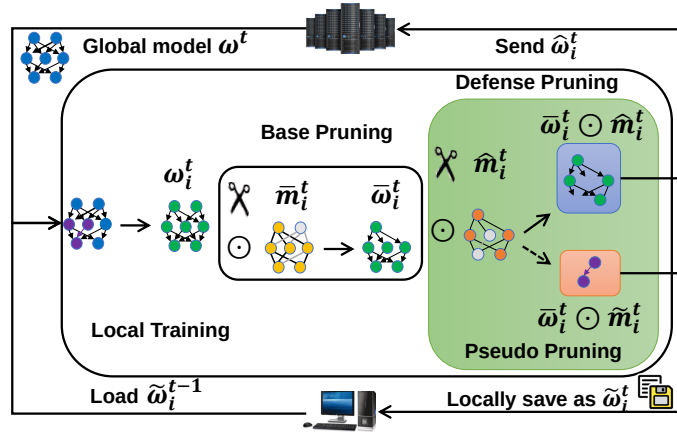


Figure 4.11: Illustration of local pruning at user i with *Pseudo-Pruning* defense: First user applies base pruning with base mask $\bar{\mathbf{m}}_i$ to derive a pruned model $\bar{\mathbf{w}}_i^t$. Then, defense pruning is conducted with mask $\hat{\mathbf{m}}_i$. Defense pruning is implemented as *Pseudo-Pruning*, where the remaining weights $\hat{\mathbf{w}}_i^t (= \bar{\mathbf{w}}_i^t \odot \hat{\mathbf{m}}_i^t)$ are sent to the server; while the pruned weights $\bar{\mathbf{w}}_i^t (= \bar{\mathbf{w}}_i^t \odot \tilde{\mathbf{m}}_i^t)$ are locally saved for future local training. Notably, these weights pruned by defense are not actually pruned, hence the term "Pseudo-Pruning."

4.5.2. The *PriPrune* Mechanism

Combining all aforementioned insights, we introduce the defense strategy, *PriPrune*, which dynamically adapts the *Pseudo-Pruning* defense mask $\hat{\mathbf{m}}$ to jointly optimize accuracy and privacy throughout the FL training process.

Loss Function. The objective of user i is to minimize a loss function that combines privacy and accuracy:

$$\mathcal{L}_{local} = \lambda_{acc} \mathcal{L}_{acc} + \lambda_{pri} \mathcal{L}_{pri} + \lambda_{sha} \sum_{l \in L, j \in d_l} \alpha_{i(l,j)}^t \quad (4.16)$$

Here, $\mathcal{L}_{acc}$ represents the local model training loss, $\mathcal{L}_{pri}$ represents the local privacy loss, and $\sum_{l \in L, j \in d_l} \alpha_{i(l,j)}^t$ serves as the privacy regularization term, where $\alpha_{i(l,j)}^t$ represents the probability of user i not sharing the j -th parameter in l -th layer with the server at round t . L represents the total number of layers in the model, and d_l refers to the number of parameters in the l -th layer. Additionally, λ_{acc} , λ_{pri} , and λ_{sha} act as weights for accuracy loss, privacy loss, and the privacy regularization term, respectively.

Next, we explain the terms of the loss function of the user in Equation 4.16 and their rationale. First, the accuracy loss is given by: $\mathcal{L}_{acc} = \mathcal{L}(\mathcal{D}_i; \hat{\mathbf{w}}_i^t | \mathbf{w}_{i,init}^t)$. The initial local weight is computed as: $\mathbf{w}_{i,init}^t = \mathbf{w}^t \odot \hat{\mathbf{m}}_i^{t-1} + \bar{\mathbf{w}}_i^{t-1} \odot \tilde{\mathbf{m}}_i^{t-1}$, where $\tilde{\mathbf{m}}$ denotes the bit-wise complement matrix of $\hat{\mathbf{m}}$. It is the composition of locally saved weights with global weights through the utilization of the defense mask $\hat{\mathbf{m}}$.

Algorithm 8: PriPrune

```
1: function UserUpdate ( $\mathbf{w}^t, \mathbf{m}^t, i$ ):  
2: Load  $\tilde{\mathbf{w}}_i^t$  from local storage of user  $i$   
3: Initialize  $\alpha_i^t = \alpha_i^{t-1}$   
4: for local epoch  $e \in E$  do  
5:   Sample defense mask  $\hat{\mathbf{m}}_i^t$  by Equation 4.18 with  $\alpha_i^t$   
6:   Combine local and global model by:  
        $\mathbf{w}_i^t[j]_{j \in d} = \begin{cases} \tilde{\mathbf{w}}_i^t[j] & \text{if } \hat{\mathbf{m}}_i^t[j] = 0 \\ \mathbf{w}_i^t[j] & \text{if } \hat{\mathbf{m}}_i^t[j] = 1 \end{cases}$   
7:   Base Pruning:  $\bar{\mathbf{w}}_i^t \leftarrow \mathbf{w}_i^t \odot \tilde{\mathbf{m}}_i^t$   
8:    $\mathcal{L}_{local} \leftarrow \text{Equation 4.16.}$   
9:   Model parameter update:  
        $\bar{\mathbf{w}}_i^t \leftarrow \bar{\mathbf{w}}_i^t - \eta \nabla \mathcal{L}_{local}(\bar{\mathbf{w}}_i^t, \alpha_i^t)$   
10:  Defense mask parameter update:  
        $\alpha_i^t \leftarrow \alpha_i^t - \eta \nabla \mathcal{L}_{local}(\bar{\mathbf{w}}_i^t, \alpha_i^t)$   
11: end for  
12:  $\hat{\mathbf{m}}_i^t \leftarrow \text{argmax}(\alpha_i^t, 1 - \alpha_i^t)$   
13: Defense Pruning:  $\hat{\mathbf{w}}_i^t \leftarrow \bar{\mathbf{w}}_i^t \odot \hat{\mathbf{m}}_i^t, \tilde{\mathbf{w}}_i^t \leftarrow \bar{\mathbf{w}}_i^t \odot \tilde{\mathbf{m}}_i^t$   
14: Return  $\hat{\mathbf{w}}_i^t$  to server, locally save  $\tilde{\mathbf{w}}_i^t$ 
```

Then, the privacy loss $\mathcal{L}_{pri}$ is given by:

$$\mathcal{L}_{pri} = \sum_{l \in L, j \in d_l} - \frac{N_l}{\sum_{l \in L} N_l} \frac{|\mathbf{g}_{l,j}|}{\sum_{j \in d_l} |\mathbf{g}_{l,j}|} \log \alpha_{i(l,j)}^t \quad (4.17)$$

N_l indicates the number of weights in l -th layer, $|\mathbf{g}_{l,j}|$ represents the magnitude of the gradient for the j -th parameter in l -th layer. The privacy loss is designed to discourage sharing weights with large gradients inspired by Insight 1, aiming to mitigate information leakage. Therefore, we assign higher weights to parameters with larger gradients in the loss function. Through this process, during the optimization of the privacy loss, the algorithm updates a higher α_i^t associated with the larger weight term. This higher α_i^t signifies a higher probability of not sharing the parameters with larger gradients with the server, thus promoting the sharing of less valuable information with the server. Moreover, we also distribute constraints to l -th layer based on its parameter count N_l , in proportion to the total model weights, which ensures uniform layer-wise sparsity.

The term $\sum_{l \in L, j \in d_l} \alpha_{i(l,j)}^t$ introduces a privacy penalty based on the magnitude of α_i^t . The optimization process aims to minimize the sum of α_i^t , resulting in a decrease in the overall probability of not sharing parameters. This, in turn, leads to an increase in the total probability of sharing parameters. Thus, while the privacy loss discourages the sharing of parameters with larger gradients with the server, this term encourages the sharing of parameters with smaller gradients, thus preserving accuracy while maintaining privacy.

Optimization. In *PriPrune*, we optimize the defense mask $\hat{\mathbf{m}}$ and model weights $\bar{\mathbf{w}}_i$ jointly through gradient descent based on our designed loss function. To overcome the discrete and non-differentiable nature of the defense mask $\hat{\mathbf{m}}$, we employ Gumbel-Softmax sampling [203], [204] to substitute the original non-differentiable sample with a differentiable sample. Specifically, $\hat{\mathbf{m}}$ is parameterized by a distribution vector $\boldsymbol{\pi}_{l,j} = [\alpha_{i(l,j)}^t, 1 - \alpha_{i(l,j)}^t]$. $\mathbf{v}_{l,j}$ represents the soft *Pseudo-Pruning* decision for the j -th parameter in layer l . The reparameterization trick is used for differentiable training:

$$\mathbf{v}_{l,j}(k) = \frac{\exp((\log \boldsymbol{\pi}_{l,j}(k) + G_{l,j}(k))/\tau)}{\sum_{k \in \{0,1\}} \exp((\log \boldsymbol{\pi}_{l,j}(k) + G_{l,j}(k))/\tau)} \quad (4.18)$$

where $G_{l,j} = -\log(-\log U_{l,j})$ is Gumbel distribution with $U_{l,j}$ sampled from a uniform i.i.d. distribution $\text{Unif}(0, 1)$, τ is the temperature of the softmax and $k \in \{0, 1\}$. We apply the hard sample trick introduced by PyTorch document [205] to acquire the defense mask $\hat{\mathbf{m}}$ and the main trick is to do $y_{\text{hard}} = f_{\text{StopGradient}}(y_{\text{soft}}) + y_{\text{soft}}$.

PriPrune. The *PriPrune* algorithm is descibed in Algorithm 8, which updates the *UserUpdate* function of Algorithm 6. As depicted in Algorithm 8, in each FL round, after user i receives the latest global model $\mathbf{w}_i^t$ from the server, they sample their defense mask $\hat{\mathbf{m}}_i$ using Equation 4.18. The defense mask $\hat{\mathbf{m}}_i$ is then employed to combine the global model $\mathbf{w}^t$ and the user's locally saved $\tilde{\mathbf{w}}_i^{t-1}$ to generate the local initial model $\mathbf{w}_{i,\text{init}}^t$. Following this composition step, the user initiates local training based on Equation 4.16, leading to updates in $\bar{\mathbf{w}}_i^t$, $\tilde{\mathbf{w}}_i^t$, and α_i^t . Then user i conducts defense pruning, retaining the non-shared weights $\tilde{\mathbf{w}}_i^t$ locally, and transmits the shared weights $\hat{\mathbf{w}}_i^t$ to the server.

4.6. Experimental Evaluation

4.6.1. Experimental Setup

The settings for datasets, models, data partitioning, and implementation details follow those described in Section 4.4.2.

Performance Metrics. To assess the effectiveness of the evaluated pruning methods, we employ the privacy metrics and utility metrics. For privacy assessment, we use the Normalized Mutual Information (NMI) between the training data and the reconstructed data as the privacy metric. Higher values of NMI indicate a higher risk of privacy leakage. Additionally, the Peak Signal-to-Noise Ratio (PSNR) is employed as a well-established metric for image quality evaluation. The utility metric relies on model accuracy (ACC) to gauge model performance. Ideally, we want improved privacy, without significantly compromising accuracy.

Hyper-parameter Selection. We conducted a search for our main parameter, the defense rate $\hat{p}_i$, from 0.1 to 0.6, as depicted in Fig. 4.12c. With regards to our hyperparameters in

loss functions, we search λ_{acc} in the list of [1, 5, 10], λ_{pri} in the list of [1, 10, 15] and λ_{sha} in the list of [2e-06, 2e-05, 2e-04]. Detailed results are provided in Appendix 4.6.3.

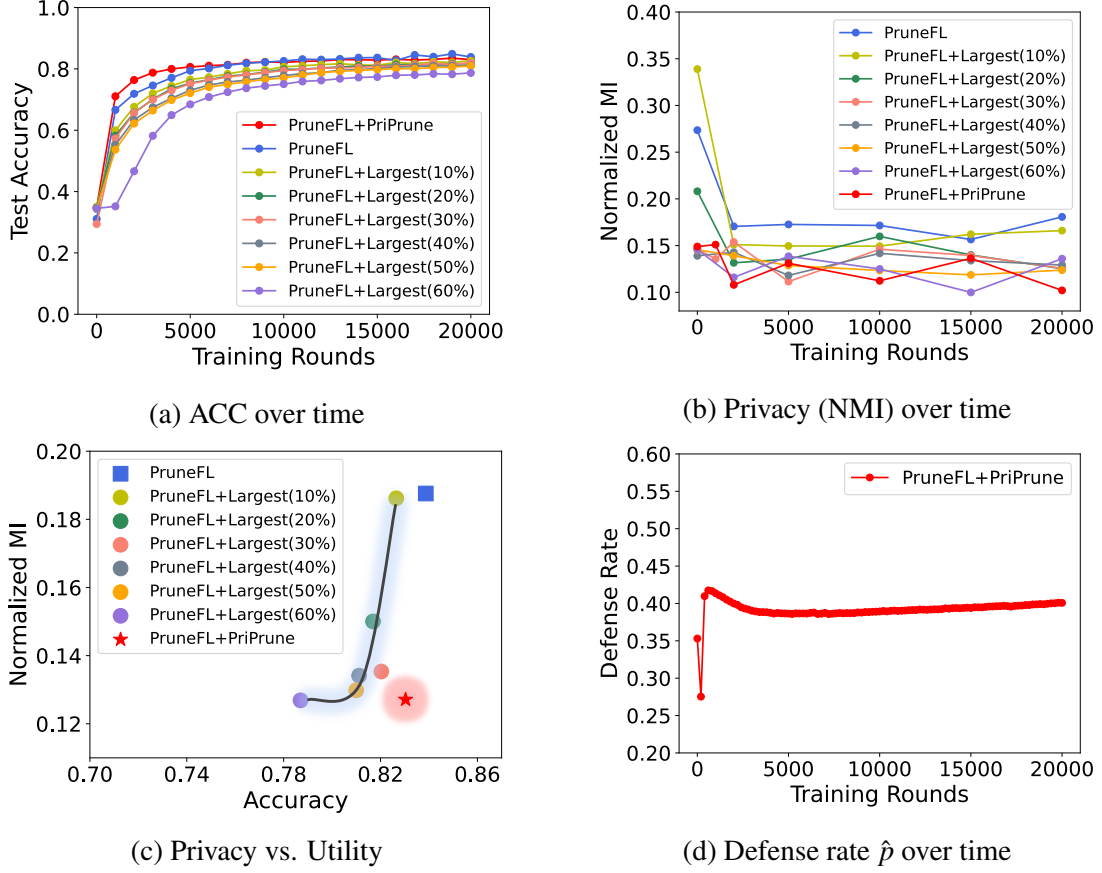


Figure 4.12: Performance evaluation of *PriPrune* with adaptive defense rate and *Largest (Pseudo)* with 6 fixed defense rates, as defense strategies, integrated with PruneFL (as the base pruning method), conducted over 20,000 training rounds across the FEMNIST dataset. (a) Comparison of ACC over training rounds under these different defense strategies. (b) Comparison of Privacy (NMI) over training rounds under these different defense strategies. (c) The Privacy-Utility tradeoff, measured in terms of NMI and ACC, respectively. (d) Change in the adaptive defense rate of *PriPrune* over training rounds.

4.6.2. Performance of *PriPrune*

Recall that in Section 4.5.1, Insight 1, we initially assessed three defense strategies atop the base pruning method, namely: *Random*, *Largest*, *Random* and *Mix*. The results in Figure 4.8 indicate that while these defense strategies enhance privacy, they also degrade utility. Consequently, in Insight 2, we introduce *Pseudo-Pruning* to optimize this tradeoff. We evaluate the aforementioned three defense strategies both with and without *Pseudo-Pruning*, revealing that their integration with *Pseudo-Pruning* leads to improved accuracy and enhanced privacy. Notably, the *Largest + Pseudo-Pruning* strategy achieves the optimal balance between privacy and utility, thus emerging as our preferred defense mecha-

nism. In Insight 3, we tackle the challenge of a fixed defense rate by proposing *PriPrune*, which incorporates *Largest + Pseudo-Pruning* (*Largest (Pseudo)*) with an adaptive defense rate. The evaluation of the privacy improvement achieved by each individual insight, in Section 4.5.1, essentially provides an ablation study. In this section, we conduct comprehensive experiments to meticulously evaluate the performance of the entire *PriPrune* across various pruning mechanisms and datasets.

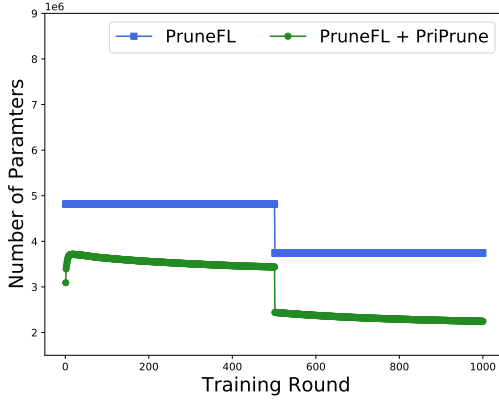
First, we present a performance comparison between the *Largest (Pseudo)* with 6 fixed defense rates and *PriPrune* with the adaptive defense rate, both are integrated with PruneFL (as the base pruning method).

Comparable Utility. Fig. 4.12a illustrates how the utility metrics, ACC, change over FL rounds. It shows the convergence curve of *PriPrune*, with its model accuracy closely aligned with that of PruneFL. This alignment indicates that *PriPrune* maintains utility comparable to PruneFL. However, for the *Largest (Pseudo)* defense, increasing its fixed defense rate from 10% to 60% results in decreased utility, rendering it unable to match the ACC achieved by PruneFL.

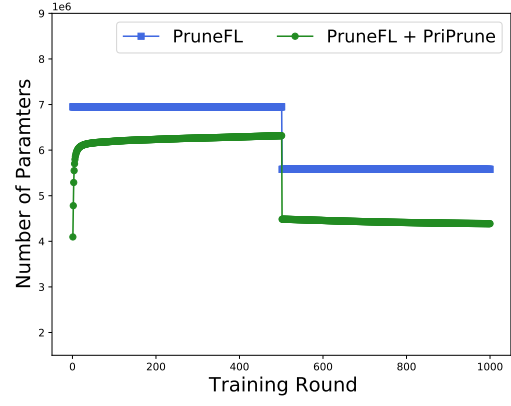
Enhanced Privacy. Figure 4.12b illustrates the change in the privacy metric NMI across FL rounds. It shows that *PriPrune* consistently maintains a low NMI throughout all FL training rounds, significantly outperforming PruneFL. Moreover, its NMI is comparable to that of *Largest (Pseudo)* with a fixed defense rate of 60%.

Optimal Tradeoff. Figure 4.12c illustrates the comparison of the privacy and utility tradeoffs among the mentioned defenses. It shows that increasing the defense rate of *Largest (Pseudo)* enhances privacy protection but at the expense of model accuracy. Conversely, *PriPrune* achieves the optimal tradeoff between privacy and accuracy. This is attributed to the adaptive nature of *PriPrune*'s defense rate $\hat{p}$, which adjusts to the model weights and is optimized jointly for privacy and model accuracy. The dynamic change of the defense rate of *PriPrune* is depicted in Fig. 4.12d.

Communication Efficiency and Training Speed. We conduct experiments to compare communication and training in PruneFL with and without *PriPrune*. Figure 4.13 demonstrates the communication cost in terms of the number of model parameters across 1000 iterations of FL. When *PriPrune* is applied, the number of model parameters is reduced by $32.20\% \pm 0.91\%$ for FEMNIST and by $15.51\% \pm 0.17\%$ for CIFAR-10, thus significantly reducing the communication cost between clients and the server. This benefit is achieved because of *Pseudo-Pruning*, which retains pseudo-pruned weights locally, but transmits fewer model updates. Figure 4.14 displays the training speed across the total time of clients' local training across 1000 iterations of FL. When *PriPrune* is applied, the training time is increased by $0.18s \pm 0.01s$ for FEMNIST and by $0.50s \pm 0.01s$ for CIFAR-10, this is a negligible increase in the training speed (due to updating the defense pruning rate). These results demonstrate that the application of *PriPrune* offers a clear advantage in reducing communication costs. Despite a minor increase in local training time, the overhead is negligible compared to the significant communication efficiency gained.

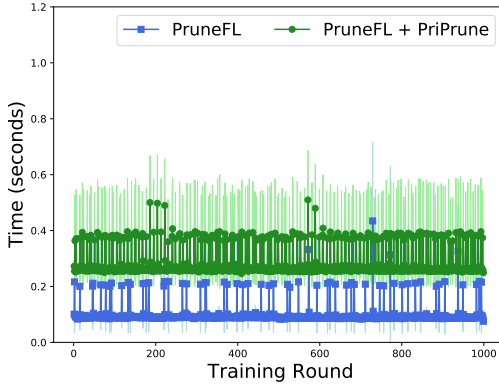


(a) Conv-2 in FEMNIST

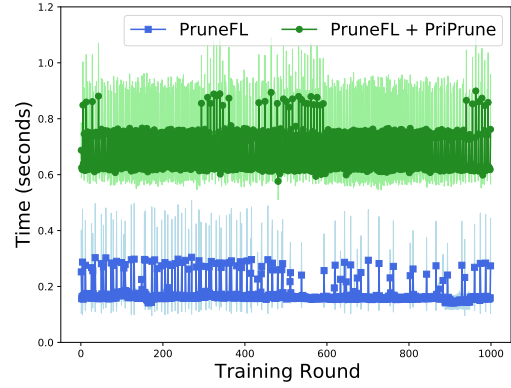


(b) VGG-11 in CIFAR-10

Figure 4.13: Comparison of the communication costs of PruneFL with and without *PriPrune* during the training process with (a) representing FL training using the Conv-2 model on the FEMNIST dataset and (b) the VGG-11 model on the CIFAR-10 dataset.



(a) Training speed in FEMNIST



(b) Training speed in CIFAR-10

Figure 4.14: Comparison of the training speed of PruneFL with and without *PriPrune* during the training process with (a) representing FL training on the FEMNIST dataset and (b) on the CIFAR-10 dataset.

These findings suggest that *PriPrune* effectively enhances the overall performance of FL by optimizing communication without significantly compromising training speed.

We have demonstrated that *PriPrune* stands out as the best defense solution among other strategies. Subsequently, we present the privacy performance of *PriPrune* when integrated with various state-of-the-art base pruning schemes in FL, such as SNIP, SynFlow, FedDST, and PruneFL, across both the FEMNIST and CIFAR-10 datasets.

***PriPrune* with State-of-the-Art Pruned FL.** Figure 4.15 shows the results of using *PriPrune* for defense after a number of state-of-the-art base pruning schemes. Across all base pruning methods, this integration consistently enhances privacy while preserving accuracy. This highlight *PriPrune*'s effectiveness and versatility in achieving an optimal tradeoff between privacy and accuracy in diverse scenarios. This outcome is primarily due

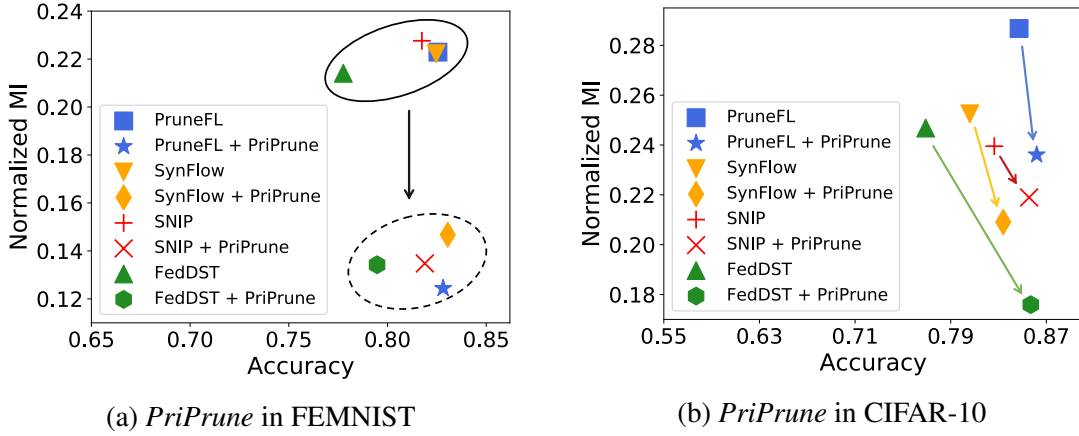


Figure 4.15: Comparison of tradeoffs between privacy-vs-accuracy among four base pruning mechanisms, both with and without the integration of *PriPrune*, across the FEMNIST and CIFAR-10 datasets.

to the nature of *PriPrune*, which not only actually prunes but also selectively retains certain critical weights for local training (through pseudo-pruning). This mechanism allows the model to maintain or even improve accuracy while reducing the information leakage that could be exploited in privacy attacks, as also shown in Fig.4.8. Specifically, *PriPrune* focuses on pseudo-pruning weights with large gradients that are most likely to leak sensitive information and keep them for local training. This selective pruning can lead to a model that is more robust against privacy attacks without a significant loss in accuracy. Additionally, the adaptive pruning rate in *PriPrune* is optimized for the tradeoff between privacy and accuracy, allowing the model to balance these two objectives effectively.

Methods	FEMNIST		CIFAR-10	
	Base	Base + <i>PriPrune</i>	Base	Base + <i>PriPrune</i>
PruneFL	0.22 ± 0.008	0.12 ± 0.007	0.29 ± 0.009	0.24 ± 0.009
FedDST	0.21 ± 0.025	0.13 ± 0.007	0.25 ± 0.008	0.18 ± 0.006
SynFlow	0.22 ± 0.009	0.15 ± 0.005	0.25 ± 0.012	0.21 ± 0.018
SNIP	0.23 ± 0.012	0.13 ± 0.014	0.24 ± 0.004	0.22 ± 0.008

Table 4.2: Comparison of privacy levels (NMI) between Base and Base + *PriPrune* for FL pruning methods. A lower NMI indicates a reduced risk of privacy leakage.

While Fig. 4.15 compares the tradeoffs between privacy-vs-accuracy, Table 4.2 further elaborates on the privacy improvement achieved by using *PriPrune* as defense after any base scheme, reporting results over three experiments. Compared to the base approach alone, adding *PriPrune* as defense consistently reduces NMI, thus improving privacy. For instance, on the FEMNIST dataset, *PriPrune* improves the privacy of *PruneFL* by 45.5% without compromising accuracy; on the CIFAR-10 dataset, *PriPrune* improves privacy of

FedDST by 28% while maintaining accuracy.

4.6.3. Hyper-parameter Selection

Table 4.3 has searched λ_{acc} in the list of [1, 5, 10], λ_{pri} in the list of [1, 10, 15] and λ_{sha} in the list of [2e-06, 2e-05, 2e-04], which showing the comparisons of privacy levels during the hyper-parameter search. As shown in Table 4.3, when we increase the λ_{pri} , the NMI value becomes lower, indicating better privacy protection. We explore different combinations of hyper-parameters and finally utilize the best set of values that could achieve the best privacy-accuracy tradeoff.

Parameter	$\lambda_{pri} = 1$			$\lambda_{pri} = 10$			$\lambda_{pri} = 15$		
	$\lambda_{sha} = 2e-04$	$\lambda_{sha} = 2e-05$	$\lambda_{sha} = 2e-06$	$\lambda_{sha} = 2e-04$	$\lambda_{sha} = 2e-05$	$\lambda_{sha} = 2e-06$	$\lambda_{sha} = 2e-04$	$\lambda_{sha} = 2e-05$	$\lambda_{sha} = 2e-06$
$\lambda_{acc} = 1$	0.166	0.171	0.156	0.138	0.153	0.154	0.138	0.159	0.151
$\lambda_{acc} = 5$	0.154	0.158	0.136	0.125	0.121	0.123	0.135	0.123	0.130
$\lambda_{acc} = 10$	0.148	0.142	0.140	0.132	0.126	0.118	0.135	0.132	0.140

Table 4.3: Comparison of privacy levels (NMI) during hyper-parameter selection: with regards to our hyperparameters in loss functions, we search λ_{acc} in the list of [1, 5, 10], λ_{pri} in the list of [1, 10, 15] and λ_{sha} in the list of [2e-06, 2e-05, 2e-04].

For different pruning methods and different datasets, we are adopting different combinations of hyper-parameters so as to achieve better privacy-utility tradeoff, as detailed in Table 4.4.

Dataset	Method	λ_{pri}	λ_{sha}	λ_{acc}
FEMNIST	PruneFL+ <i>PriPrune</i>	15	2×10^{-5}	5
	Synflow+ <i>PriPrune</i>	1	2×10^{-6}	10
	SNIP+ <i>PriPrune</i>	1	2×10^{-6}	10
	FedDST+ <i>PriPrune</i>	8	2×10^{-5}	10
CIFAR-10	PruneFL+ <i>PriPrune</i>	10	2×10^{-5}	5
	Synflow+ <i>PriPrune</i>	1	2×10^{-6}	5
	SNIP+ <i>PriPrune</i>	1	2×10^{-6}	10
	FedDST+ <i>PriPrune</i>	10	2×10^{-5}	10

Table 4.4: Parameter settings for λ_{pri} , λ_{sha} , and λ_{acc} in FEMNIST and CIFAR-10 datasets.

4.7. Conclusion and Future Work

In this chapter, we revisit federated learning with model pruning, from the point of view of privacy. First, we quantify the information leakage and privacy gain offered for model pruning in FL, both theoretically and experimentally. Inspired by insights obtained from our extensive privacy quantification, we design *PriPrune*— an adaptive privacy-preserving

local pruning mechanism in FL, that is jointly optimized for privacy and model performance. Our proposed mechanism achieves the best tradeoff between privacy and accuracy across diverse pruning methods and datasets under privacy attacks.

One direction for our future work is to combine our pruning-based defense with classic, orthogonal defenses in FL such as DP and SecAgg, to enhance privacy further. DP could be applied to the remaining weights after PriPrune to ensure additional privacy at the user level. Similarly, the remaining weights that need to be sent back to the server would be encrypted using SecAgg. In this process, each client would encrypt their remaining weights with a shared key, ensuring that no individual client’s update can be reconstructed from the aggregated result. This combination would ensure that privacy is maintained at both the client-server communication level and the aggregation process.

5. DISCUSSION

Since its introduction, Federated Learning (FL) has emerged as a groundbreaking approach to decentralized machine learning, drawing significant interest for its potential to enhance data security and user privacy. Over recent years, FL has transitioned from theoretical frameworks to practical implementations, reflecting a growing synergy between academic research and industrial innovation. This collaborative momentum has positioned FL as a transformative tool for applications requiring decentralized data processing, such as autonomous systems, personalized healthcare, and industrial automation.

Despite these advancements, FL remains an evolving discipline with unresolved challenges. Key obstacles include ensuring robust security protocols, safeguarding privacy without compromising model performance, addressing scalability in diverse environments, and embedding ethical considerations into algorithmic design. While the scope of these challenges is broad, our work focuses specifically on fortifying security and privacy foundational requirements for sustainable, trustworthy FL systems.

The adoption of FL marks a paradigm shift in distributed machine learning, offering solutions that align with legal, economic, and strategic constraints. Its real-world impact is already evident in sectors like automotive technology, where FL enables collaborative model training for autonomous vehicles, and in healthcare, where it supports privacy-preserving analysis of medical data. By refining FL's security and privacy mechanisms, this research contributes to broader efforts aimed at deploying reliable, ethical decentralized learning systems.

In this chapter, we synthesize our contributions, evaluate their limitations, and outline pathways for future works. Our investigations respond to the challenges outlined in Section 1.2 and research questions introduced in Section 1.3. By advancing methodologies to address these issues, we aim to strengthen FL's viability in sensitive, high-stakes applications while fostering trust among stakeholders.

5.1. Enhancing Byzantine Robustness and Security in FL

This section outlines the initial research questions centred on strengthening Byzantine resilience and security within FL, alongside the methodologies presented in Chapters 2 and 3. First, we explore the development of a reputation-driven aggregation framework for FL that leverages historical training data to mitigate poisoning attacks. Second, we address the core vulnerabilities in FL aggregation mechanism FEDAvg that enable poisoning threats and introduce an advanced aggregation strategy inspired from voting system to counteract these risks.

5.1.1. Reputation-based Robust Aggregation

In Chapter 2, we address the following question:

Q1. *Can we design a robust aggregation mechanism that retains information from previous rounds to build client reputations, withstand poisoning attacks, and achieve fast convergence in Non-IID data settings?*

To answer **Q1**, we propose a reputation-based aggregation framework that combines historical client behaviour analysis with residual-based anomaly detection. We compute dynamic reputation scores for clients across training rounds using subjective logic [87]. By retaining historical model update patterns, the system evaluates clients based on their consistency and reliability over time. This allows the global model to distinguish benign clients from malicious ones, even as data distributions shift or evolve.

To mitigate malicious updates, we integrate residual-based anomaly detection. This technique quantifies deviations between client updates and expected model behaviour. By combining residuals with reputation scores, our mechanism filters out poisoned contributions while preserving valuable Non-IID data from benign clients.

We prove theoretically that our method converges to a near-optimal solution under Byzantine faults, with performance gaps bounded by the fraction of malicious clients. Furthermore, the theoretical error rate provides insights for experimental design. Specifically, it highlights how model complexity, the number of participating clients, and hyperparameter choices impact model performance. It also offers guidance for tuning the reputation parameters based on observed client behavior, in order to maintain an optimal balance between convergence speed and robustness. Experimentally, our approach achieves $1.6\times$ – $2.4\times$ faster convergence compared to state-of-the-art methods (e.g., FedAvg [26], FoolsGold [55]) on real-world datasets, even under highly heterogeneous data distributions. Our reputation-residual framework reduces Attack Success Rate (ASR) by 72.3% on average compared to baseline defenses. Moreover, the system dynamically adapts to new sensitive content (e.g., COVID-19 URLs), achieving 92% accuracy on emerging labels despite noisy user inputs, as validated via the *EITR* browser extension.

Limitations and Future Work The current evaluation of *EITR* is confined to a limited user-base and predefined content categories, leaving its scalability and adaptability to user-defined classifications untested. Additionally, while our reputation scheme is designed to accommodate diverse threat models, its robustness against more sophisticated adversarial attacks and complex real-world scenarios remains to be rigorously validated. The broader applicability of our framework to FL models beyond sensitive content classification, though theoretically supported, requires empirical verification in varied contexts. Future work will focus on expanding *EITR*’s deployment to diverse user communities and evaluating its performance in classifying both sensitive and non-sensitive content, including dynamic, user-defined categories tailored to specific applications. We also plan

to explore advanced adversarial scenarios and integrate our reputation mechanism with other FL paradigms, such as collaborative model training in heterogeneous or privacy-critical environments. These efforts will further bridge the gap between theoretical trust frameworks and practical, scalable FL solutions.

5.1.2. A Superior Fundamental Aggregation

In Chapter 3, we address the following question:

Q2. *Can we design a superior aggregation mechanism based on a voting system that assigns fairer weights to clients' contributions, and can this mechanism serve as a fundamental aggregation technique that integrates with other Byzantine-robust and privacy-enhancing methods?*

We propose FEDQV, a novel FL aggregation mechanism inspired by quadratic voting principles. Unlike traditional averaging-based methods, FEDQV assigns client weights through a quadratic voting scheme that imposes a super-linear tax on malicious peers, curbing their potential impact. To bolster security, the server employs a masked voting rule, concealing weight calculations from clients and preventing manipulation of the aggregation process. The framework further integrates server-side outlier detection and a limited voting budget—confidential parameters are maintained exclusively on the server, all at low computational and communication cost.

FEDQV dynamically adjusts voting budgets based on client reputation scores, which evolve to progressively constrain adversarial influence while rewarding trustworthy participants with greater voting power. Crucially, the mechanism preserves the convergence rate of FEDAVG in non-adversarial settings, ensuring compatibility with standard FL performance benchmarks. FEDQV also converges to a near-optimal solution even in the presence of malicious parties. The corresponding theoretical error bound informs the design of our experiments, particularly guiding key considerations such as the selection of hyperparameters, including the voting budget and similarity threshold. Additionally, FEDQV enforces truthfulness: clients cannot gain strategic advantages by misreporting data preferences or inflating local data sizes, as the structure inherently discourages dishonest behaviour.

FEDQV is designed as a complementary method, enhancing—not replacing—existing defenses. For example when integrated with Multi-Krum [39] or Trimmed-Mean [53], FEDQV weakens malicious clients through quadratic weighting, improving accuracy by 26.72% and reducing attack success rates by 70%. In addition, by integrating with secure aggregation (SECAGG), FEDQV mitigates the risks of inference attacks.

These properties position FEDQV as a superior foundational aggregation method than FEDAVG, for secure FL. Its modularity allows integration with robustness protocols and orthogonal privacy mechanisms, enabling layered defenses against evolving threats. Empirical validation confirms its scalability and adaptability. By harmonizing fairness, secu-

ity, and efficiency, FEDQV establishes a new standard for building trustworthy federated systems in untrusted environments.

Limitations and Future Work Further opportunities exist to enhance FEDQV’s privacy-utility tradeoff by integrating it with other orthogonal privacy mechanisms. For instance, incorporating differential privacy (DP) could provide rigorous mathematical guarantees against data leakage. This could involve adding calibrated noise to local model updates or final aggregated predictions while carefully balancing the noise magnitude to preserve model accuracy. Furthermore, exploring synergies with secure multi-party computation (SMPC) or homomorphic encryption (HE) could strengthen defenses against malicious clients attempting to manipulate the voting process or intercept intermediate results.

Another promising direction is the development of adaptive privacy mechanisms that dynamically adjust privacy parameters based on contextual factors, such as the sensitivity of the task, the trustworthiness of participants, or the detected severity of threats. For example, a hybrid system could combine FEDQV with lightweight DP during normal operation and switch to stronger SMPC-based protocols when anomalous behavior or heightened risks are detected. Additionally, investigating hybrid aggregation frameworks that merge FEDQV with other Byzantine-resilient aggregation rules could further harden the system against data or model poisoning attacks. By addressing these challenges, future work could establish FEDQV as a foundational component of privacy-aware, robust federated learning systems capable of operating in diverse and untrusted environments.

5.2. Privacy leakage in FL

This section outlines the research questions centred on privacy-preserving within FL, alongside the methodologies presented in Chapters 4. First, we explore how to quantify the privacy in FL model pruning. Second, we bring up the mechanism to

The first research question we solved is:

Q3. *Can we quantify privacy in FL model pruning?*

We begin by establishing the first theoretical framework for quantifying privacy leakage in model pruning. Using information-theoretic principles, we derive universal upper bounds on the leakage of sensitive information about individual user datasets through model updates in any FL pruning paradigm. These bounds express the leakage in terms of key parameters such as the pruning rate, batch size, model size, and number of training rounds, offering valuable insights into the factors that influence privacy in practical pruning algorithms. This constitutes the first theoretical characterization of privacy risks in pruned FL systems.

Complementing our theoretical analysis, we perform an extensive empirical study evaluating practical privacy leakage across six distinct pruning schemes. Our investi-

gation employs multiple state-of-the-art privacy attack methodologies, including gradient inversion techniques such as Deep Leakage from Gradients [70] and Gradient Inversion [37]. Notably, we introduce Sparse Gradient Inversion (SGI), a novel attack strategy specifically designed to exploit vulnerabilities unique to pruned FL architectures. This dual theoretical-empirical approach yields critical insights into how pruning mechanisms and parameter configurations influence privacy preservation.

Q4. *How can we further optimize pruning for privacy, and jointly with accuracy?*

Based on these findings, we propose **PriPrune**, a novel privacy-aware pruning framework that establishes dual defense mechanisms against gradient inversion attacks in federated learning systems. *PriPrune* implements an additional client-side defense pruning layer using personalized mask configurations. Through Gumbel Softmax-optimized backpropagation, our framework dynamically adjusts defense pruning ratios to achieve a Pareto-optimal balance between model utility and privacy preservation.

Another core innovation in the design of *PriPrune* is the *Pseudo-Pruning*: it introduces a novel weight retention strategy where defense-pruned parameters remain active in local model updates while being excluded from server communications. This bifurcated handling of parameters simultaneously enhances privacy through transmitted model obfuscation and maintains model fidelity through local gradient preservation.

Through extensive evaluations across multiple benchmark datasets, various FL pruning schemes and state-of-the-art attacks, *PriPrune* demonstrates superior privacy-accuracy tradeoffs than privacy-unaware pruned FL baselines. For example, on the FEMNIST dataset, *PriPrune* improves the privacy *PruneFL* [78] by 45.5% without imposing any accuracy loss.

Limitations and Future Work Although *PriPrune* significantly reduces privacy risks through pruning, threats remain if adversaries compromise the pruned model or intercept communications. Employing additional layers of protection, such as secure aggregation or homomorphic encryption, could enhance privacy while balancing the resulting computational overhead.

One direction for our future work is to combine our pruning-based defense with classic, orthogonal defenses in FL, such as differential privacy (DP) and secure aggregation, to enhance privacy further. DP could be applied to the remaining weights after *PriPrune* to ensure additional privacy at the user level. Similarly, the remaining weights that need to be sent back to the server would be encrypted using secure aggregation. In this process, each client would encrypt their remaining weights with a shared key, ensuring that no individual client’s update can be reconstructed from the aggregated result. This combination would ensure that privacy is maintained at both the client-server communication level and the aggregation process.

Rigorous empirical evaluation in real-world, large-scale federated environments—such

as cross-device IoT networks or healthcare collaborations—will be critical to validate the scalability and practicality of these enhancements. Longitudinal studies assessing the long-term impact of privacy-utility tradeoffs, as well as the framework’s resilience against evolving adversarial tactics, would provide deeper insights into its operational viability.

5.3. Open-Source Code Contribution

To support reproducibility, transparency, and further research, we have open-sourced the code corresponding to each major component of this thesis. Below, we summarize the provided repositories and their relevance to different sections:

- **Reputation-Based Aggregation Framework** (Chapter 2): The implementation of the reputation-based aggregation framework, including re-weighting mechanisms, subjective logic reputation updates, and integration with FL pipelines, is available at <https://github.com/TianyueChu/FedRep>.
- **FEDQV Implementation** (Chapter 3): The implementation of the FEDQV algorithm, including the similarity computation and voting scheme, attack simulations, and evaluation pipelines, is available at <https://github.com/TianyueChu/FedQV>.

The code for *PriPrune* is currently being prepared for open-sourcing. Each repository contains detailed README files with instructions for reproducing the main results presented in this thesis. By releasing these resources, we aim to facilitate validation, extension, and further development by the broader research community.

5.4. Conclusion

This thesis has addressed critical challenges in Federated Learning (FL) by enhancing Byzantine robustness and improving privacy preservation while maintaining model efficiency. Through the integration of theoretical advancements and practical implementations, the research lays a strong foundation for secure, privacy-aware FL systems, thus facilitating their potential deployment in real-world applications.

A primary contribution is a robust aggregation approach that leverages historical data to fight against poisoning attacks in sensitive content classification without compromising convergence speed or accuracy, as evidenced by the successful deployment of the EITR browser extension. The introduction of FEDQV, a novel aggregation mechanism based on quadratic voting, further underscores the potential for robust and flexible FL aggregation. By seamlessly integrating with reputation models, Byzantine-robust methods, and privacy-preserving techniques, FEDQV offers a superior alternative to FEDAVG against a wide range of poisoning attacks. Complementing these robust aggregation strategies,

the thesis also presents *PriPrune*, an adaptive pruning mechanism that curtails information leakage in local model updates, thereby substantially improving privacy protections while maintaining strong model performance.

Rigorous theoretical analyses and extensive empirical evaluations validate the effectiveness of these methods across diverse datasets and scenarios. Collectively, the findings demonstrate how robust aggregation techniques and privacy-aware pruning fortify FL in decentralized machine-learning contexts. By mitigating malicious client behavior and addressing inherent privacy concerns, this work paves the way for secure, privacy-preserving FL systems suitable for sensitive domains. Together, these contributions represent a significant stride toward realizing the vision of collaborative, decentralized, and trustworthy machine learning.

The methods and tools introduced in this thesis—including reputation-aware defense mechanisms, quadratic voting-based aggregation, and privacy-preserving pruning FL—establish a powerful framework for developing next-generation FL architectures that simultaneously address security vulnerabilities and data confidentiality challenges. Collectively, these advances serve as a crucial step toward the broader goal of collaborative, decentralized, and trustworthy machine learning, opening new possibilities for domains where data sharing has traditionally been restricted by security or privacy concerns.

BIBLIOGRAPHY

- [1] T. Chu, Á. García-Recuero, C. Iordanou, G. Smaragdakis, and N. Laoutaris, “Securing federated sensitive topic classification against poisoning attacks,” in *30th Annual Network and Distributed System Security Symposium, NDSS 2023*, San Diego, California, USA: The Internet Society, 2023. [Online]. Available: <https://www.ndss-symposium.org/ndss-paper/securing-federated-sensitive-topic-classification-against-poisoning-attacks/>.
- [2] T. Chu and N. Laoutaris, “Fedqv: Leveraging quadratic voting in federated learning,” *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 8, no. 2, May 2024. doi: [10.1145/3656006](https://doi.org/10.1145/3656006). [Online]. Available: <https://doi.org/10.1145/3656006>.
- [3] T. Chu, M. Yang, N. Laoutaris, and A. Markopoulou, “Priprune: Quantifying and preserving privacy in pruned federated learning,” *arXiv preprint arXiv:2310.19958*, 2023.
- [4] T. Chu, D. İşler, N. Laoutaris, *et al.*, “Strengthening privacy in robust federated learning through secure aggregation,” in *Workshop on Artificial Intelligence System with Confidential Computing (AISCC 2024), co-located with NDSS Symposium 2024*, 2024.
- [5] T. Chu, M. Yang, N. Laoutaris, A. Markopoulou, *et al.*, “Information-theoretical bounds on privacy leakage in pruned federated learning,” in *ISIT 2024 Workshop on Information-Theoretic Methods for Trustworthy Machine Learning*, 2024.
- [6] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4.
- [7] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.
- [8] M. I. Jordan and T. M. Mitchell, “Machine learning: Trends, perspectives, and prospects,” *Science*, vol. 349, no. 6245, pp. 255–260, 2015. doi: [10.1126/science.aaa8415](https://doi.org/10.1126/science.aaa8415). eprint: <https://www.science.org/doi/pdf/10.1126/science.aaa8415>. [Online]. Available: <https://www.science.org/doi/abs/10.1126/science.aaa8415>.
- [9] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., vol. 25, Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

- [11] A. Toshev and C. Szegedy, “DeepPose: Human pose estimation via deep neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2014.
- [12] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015.
- [13] A. Diba, V. Sharma, A. Pazandeh, H. Pirsiavash, and L. Van Gool, “Weakly supervised cascaded convolutional networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017.
- [14] W. Wang, N. Yang, F. Wei, B. Chang, and M. Zhou, “Gated self-matching networks for reading comprehension and question answering,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 189–198.
- [15] J. D. M.-W. C. Kenton and L. K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” in *Proceedings of naacL-HLT*, Minneapolis, Minnesota, vol. 1, 2019, p. 2.
- [16] T. McCoy, E. Pavlick, and T. Linzen, “Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, A. Korhonen, D. Traum, and L. Màrquez, Eds., Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 3428–3448. doi: [10.18653/v1/P19-1334](https://doi.org/10.18653/v1/P19-1334). [Online]. Available: <https://aclanthology.org/P19-1334/>.
- [17] A. Van den Oord, S. Dieleman, and B. Schrauwen, “Deep content-based music recommendation,” *Advances in neural information processing systems*, vol. 26, 2013.
- [18] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, “Learning deep structured semantic models for web search using clickthrough data,” in *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, 2013, pp. 2333–2338.
- [19] A. M. Elkahky, Y. Song, and X. He, “A multi-view deep learning approach for cross domain user modeling in recommendation systems,” in *Proceedings of the 24th international conference on world wide web*, 2015, pp. 278–288.
- [20] 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*, vol. 112, pp. 59–67, 2018.
- [21] A. Hard *et al.*, “Federated learning for mobile keyboard prediction,” *CoRR*, 2018. arXiv: [1811.03604](https://arxiv.org/abs/1811.03604). [Online]. Available: <http://arxiv.org/abs/1811.03604>.

- [22] W. Yang, Y. Zhang, K. Ye, L. Li, and C.-Z. Xu, “Ffd: A federated learning based method for credit card fraud detection,” in *Big Data–BigData 2019: 8th International Congress, Held as Part of the Services Conference Federation, SCF 2019, San Diego, CA, USA, June 25–30, 2019, Proceedings 8*, Springer, 2019, pp. 18–32.
- [23] J. Posner, L. Tseng, M. Aloqaily, and Y. Jararweh, “Federated learning in vehicular networks: Opportunities and solutions,” *IEEE Network*, vol. 35, no. 2, pp. 152–159, 2021. doi: [10.1109/MNET.011.2000430](https://doi.org/10.1109/MNET.011.2000430).
- [24] E. Commission, *Data protection in the eu, the general data protection regulation (gdpr); regulation (eu) 2016/679*, <https://ec.europa.eu/info/law/law-topic/data-protection/>, 2018.
- [25] S. of California, *California consumer privacy act - assembly bill no. 375*, https://leginfo.ca.gov/faces/billTextClient.xhtml?bill_id=201720180AB375, 2018.
- [26] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Artificial intelligence and statistics*, PMLR, 2017, pp. 1273–1282.
- [27] J. Konečný, H. McMahan, F. Yu, P. Richtárik, A. Suresh, and D. Bacon, “Federated learning: Strategies for improving communication efficiency,” in *Proceedings of NIPS*, 2016.
- [28] J. Feng, C. Rong, F. Sun, D. Guo, and Y. Li, “Pmf: A privacy-preserving human mobility prediction framework via federated learning,” *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 4, pp. 1–21, 2020.
- [29] T. Yu *et al.*, “Learning context-aware policies from multiple smart homes via federated multi-task learning,” in *IEEE/ACM International Conference on Internet-of-Things Design and Implementation (IoTDI)*, 2020, pp. 104–115.
- [30] L. Huang, A. Shea, H. Qian, A. Masurkar, H. Deng, and D. Liu, “Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records,” *Journal of Biomedical Informatics*, vol. 99, p. 103 291, 2019.
- [31] J. Lee, J. Sun, F. Wang, S. Wang, C. Jun, and X. Jiang, “Privacy-preserving patient similarity learning in a federated environment: Development and analysis,” *JMIR Medical Informatics*, vol. 6, e7744, 2018.
- [32] M. Fang, X. Cao, J. Jia, and N. Gong, “Local model poisoning attacks to byzantine-robust federated learning,” in *29th USENIX Security Symposium (USENIX Security 20)*, Virtual: USENIX Association, 2020, pp. 1605–1622.

- [33] V. Shejwalkar and A. Houmansadr, “Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning,” in *28th Annual Network and Distributed System Security Symposium, NDSS 2021, virtually, February 21–25, 2021*, The Internet Society, 2021. [Online]. Available: <https://www.ndss-symposium.org/ndss-paper/manipulating-the-byzantine-optimizing-model-poisoning-attacks-and-defenses-for-federated-learning/>.
- [34] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, “How to backdoor federated learning,” in *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, S. Chiappa and R. Calandra, Eds., ser. Proceedings of Machine Learning Research, vol. 108, PMLR, Aug. 2020, pp. 2938–2948. [Online]. Available: <https://proceedings.mlr.press/v108/bagdasaryan20a.html>.
- [35] Z. Zhang *et al.*, “Neurotoxin: Durable backdoors in federated learning,” in *International Conference on Machine Learning (ICML)*, PMLR, 2022, pp. 26 429–26 446.
- [36] B. Zhao, K. R. Mopuri, and H. Bilen, *Idlg: Improved deep leakage from gradients*, 2020. arXiv: [2001.02610 \[cs.LG\]](https://arxiv.org/abs/2001.02610).
- [37] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, “Inverting gradients-how easy is it to break privacy in federated learning?” *Advances in Neural Information Processing Systems(NeurIPS)*, vol. 33, pp. 16 937–16 947, 2020.
- [38] S. Karimireddy, L. He, and M. Jaggi, “Learning from history for byzantine robust optimization,” in *International Conference on Machine Learning*, 2021, pp. 5311–5319.
- [39] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, “Machine learning with adversaries: Byzantine tolerant gradient descent,” in *Advances in Neural Information Processing Systems*, I. Guyon *et al.*, Eds., vol. 30, Long Beach, CA, USA: Curran Associates, Inc., 2017, pp. 119–129.
- [40] Q. Yang, J. Zhang, W. Hao, G. P. Spell, and L. Carin, “Flop: Federated learning on medical datasets using partial networks,” ser. KDD ’21, Virtual Event, Singapore: Association for Computing Machinery, 2021, pp. 3845–3853. doi: [10.1145/3447548.3467185](https://doi.org/10.1145/3447548.3467185). [Online]. Available: <https://doi.org/10.1145/3447548.3467185>.
- [41] J. Xu, Z. Xu, P. Walker, and F. Wang, “Federated patient hashing,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, pp. 6486–6493, Apr. 2020. doi: [10.1609/aaai.v34i04.6121](https://doi.org/10.1609/aaai.v34i04.6121). [Online]. Available: <https://ojs.aaai.org/index.php/AAAI/article/view/6121>.
- [42] M. Ammad-Ud-Din *et al.*, “Federated collaborative filtering for privacy-preserving personalized recommendation system,” *arXiv preprint arXiv:1901.09888*, 2019.
- [43] I. Dayan *et al.*, “Federated learning for predicting clinical outcomes in patients with covid-19,” *Nature medicine*, vol. 27, no. 10, pp. 1735–1743, 2021.

- [44] W. Zheng, L. Yan, C. Gou, and F.-Y. Wang, “Federated meta-learning for fraudulent credit card detection,” in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, 2021, pp. 4654–4660.
- [45] K. Cheng *et al.*, “Secureboost: A lossless federated learning framework,” *IEEE intelligent systems*, vol. 36, no. 6, pp. 87–98, 2021.
- [46] P. Chatterjee, D. Das, and D. B. Rawat, “Federated learning empowered recommendation model for financial consumer services,” *IEEE Transactions on Consumer Electronics*, 2023.
- [47] U. M. Aivodji, S. Gambs, and A. Martin, “Iotfla: A secured and privacy-preserving smart home architecture implementing federated learning,” in *2019 IEEE security and privacy workshops (SPW)*, IEEE, 2019, pp. 175–180.
- [48] P. Zhou, H. Xu, L. H. Lee, P. Fang, and P. Hui, “Are you left out? an efficient and fair federated learning for personalized profiles on wearable devices of inferior networking conditions,” *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 6, no. 2, pp. 1–25, 2022.
- [49] S. Baghersalimi, T. Teijeiro, A. Aminifar, and D. Atienza, “Decentralized federated learning for epileptic seizures detection in low-power wearable systems,” *IEEE Transactions on Mobile Computing*, 2023.
- [50] S. Ramaswamy, R. Mathews, K. Rao, and F. Beaufays, “Federated learning for emoji prediction in a mobile keyboard,” *arXiv preprint arXiv:1906.04329*, 2019.
- [51] A. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, “Analyzing federated learning through an adversarial lens,” in *International Conference on Machine Learning*, 2019, pp. 634–643.
- [52] S. Fu, C. Xie, B. Li, and Q. Chen, “Attack-resistant federated learning with residual-based reweighting,” in *AAAI Workshops: Towards Robust, Secure and Efficient Machine Learning*, 2021.
- [53] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, “Byzantine-robust distributed learning: Towards optimal statistical rates,” in *International Conference on Machine Learning*, PMLR, 2018, pp. 5650–5659.
- [54] C. Fung, C. Yoon, and I. Beschastnikh, “Mitigating sybils in federated learning poisoning,” *arXiv preprint arXiv:1808.04866*, 2018.
- [55] C. Fung, C. Yoon, and I. Beschastnikh, “The limitations of federated learning in sybil settings,” in *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*, <https://www.usenix.org/conference/raid2020/presentation/fung>, 2020, pp. 301–316.
- [56] E. M. El Mahdi, R. Guerraoui, and S. Rouault, “The hidden vulnerability of distributed learning in byzantium,” in *International Conference on Machine Learning*, 2018, pp. 3521–3530.

- [57] L. Zhao, J. Jiang, B. Feng, Q. Wang, C. Shen, and Q. Li, “Sear: Secure and efficient aggregation for byzantine-robust federated learning,” *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 5, pp. 3329–3342, 2022. doi: [10.1109/TDSC.2021.3093711](https://doi.org/10.1109/TDSC.2021.3093711).
- [58] Z. Ma, J. Ma, Y. Miao, Y. Li, and R. H. Deng, “Shieldfl: Mitigating model poisoning attacks in privacy-preserving federated learning,” *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 1639–1654, 2022. doi: [10.1109/TIFS.2022.3169918](https://doi.org/10.1109/TIFS.2022.3169918).
- [59] X. Chen, H. Yu, X. Jia, and X. Yu, “Apfed: Anti-poisoning attacks in privacy-preserving heterogeneous federated learning,” *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 5749–5761, 2023. doi: [10.1109/TIFS.2023.3315125](https://doi.org/10.1109/TIFS.2023.3315125).
- [60] X. Zhang *et al.*, “Fltracer: Accurate poisoning attack provenance in federated learning,” *IEEE Transactions on Information Forensics and Security*, 2024.
- [61] M. Fang, S. Nabavirazavi, Z. Liu, W. Sun, S. S. Iyengar, and H. Yang, “Do we really need to design new byzantine-robust aggregation rules?,” 2022.
- [62] X. Wu, H. Yuan, X. Li, J. Ni, and R. Lu, “Evaluating security and robustness for split federated learning against poisoning attacks,” *IEEE Transactions on Information Forensics and Security*, 2024.
- [63] Y. Cao, J. Zhang, Y. Zhao, H. Shen, and H. Huang, “Privacy-preserving federated learning with improved personalization and poison rectification of client models,” *IEEE Transactions on Information Forensics and Security*, 2024.
- [64] J. Xue, S. Sun, M. Liu, Q. Li, and K. Xu, “Enhancing federated learning robustness using locally benignness-assessable bayesian dropout,” *IEEE Transactions on Information Forensics and Security*, 2025.
- [65] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, “Federated optimization in heterogeneous networks,” *Proceedings of Machine Learning and Systems*, vol. 2, pp. 429–450, 2020.
- [66] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, “Tackling the objective inconsistency problem in heterogeneous federated optimization,” *Advances in neural information processing systems*, vol. 33, pp. 7611–7623, 2020.
- [67] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, “Scaffold: Stochastic controlled averaging for federated learning,” in *International conference on machine learning*, PMLR, 2020, pp. 5132–5143.
- [68] D. A. E. Acar, Y. Zhao, R. M. Navarro, M. Mattina, P. N. Whatmough, and V. Saligrama, “Federated learning based on dynamic regularization,” *arXiv preprint arXiv:2111.04263*, 2021.

- [69] K. Katevas, D. Perino, and N. Kourtellis, “Flaas-enabling practical federated learning on mobile environments,” in *Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services*, 2022, pp. 605–606.
- [70] L. Zhu, Z. Liu, and S. Han, “Deep leakage from gradients,” in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Vancouver Convention Center, Vancouver CANADA: Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/60a6c4002cc7b29142def8871531281a-Paper.pdf.
- [71] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, “Beyond inferring class representatives: User-level privacy leakage from federated learning,” in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 2512–2520. doi: [10.1109/INFOCOM.2019.8737416](https://doi.org/10.1109/INFOCOM.2019.8737416).
- [72] 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*, ser. CCS ’17, Dallas, Texas, USA: Association for Computing Machinery, 2017, pp. 603–618. doi: [10.1145/3133956.3134012](https://doi.org/10.1145/3133956.3134012). [Online]. Available: <https://doi.org/10.1145/3133956.3134012>.
- [73] J. Wang, S. Guo, X. Xie, and H. Qi, “Protect privacy from gradient leakage attack in federated learning,” in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, IEEE, 2022, pp. 580–589.
- [74] J. Sun, A. Li, B. Wang, H. Yang, H. Li, and Y. Chen, “Soteria: Provable defense against privacy leakage in federated learning from representation perspective,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 9311–9319.
- [75] N. Lee, T. Ajanthan, and P. H. Torr, “Snip: Single-shot network pruning based on connection sensitivity,” in *International Conference on Learning Representations(ICLR)*, 2019.
- [76] H. Tanaka, D. Kunin, D. L. Yamins, and S. Ganguli, “Pruning neural networks without any data by iteratively conserving synaptic flow,” in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 6377–6389.
- [77] C. Wang, G. Zhang, and R. Grosse, “Picking winning tickets before training by preserving gradient flow,” in *International Conference on Learning Representations(ICLR)*, 2020.
- [78] Y. Jiang *et al.*, “Model pruning enables efficient federated learning on edge devices,” *IEEE Transactions on Neural Networks and Learning Systems*, 2022.
- [79] S. Bibikar, H. Vikalo, Z. Wang, and X. Chen, “Federated dynamic sparse training: Computing less, communicating less, yet learning better,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022, pp. 6080–6088.

- [80] G. of Canada, *Amended act on the personal information protection and electronic documents act*, <https://www.priv.gc.ca/en/privacy-topics/privacy-laws-in-canada/the-personal-information-protection-and-electronic-documents-act-pipeda/>, 2018.
- [81] G. of Israel, *Protection of privacy regulations (data security) 5777-2017*, https://www.gov.il/en/Departments/legalInfo/data_security_regulation/, 2018.
- [82] J. Personal Information Protection Commission, *Amended act on the protection of personal information*, <https://www.ppc.go.jp/en/>, 2017.
- [83] A. I. Commissioner, *Australian privacy principles guidelines; australian privacy principle 5 - notification of the collection of personal information*, <https://www.oaic.gov.au/agencies-and-organisations/app-guidelines/>, 2018.
- [84] S. Matic, C. Iordanou, G. Smaragdakis, and N. Laoutaris, “Identifying sensitive urls at web-scale,” in *Proceedings of the ACM Internet Measurement Conference*, 2020, pp. 619–633.
- [85] *Curlie - the collector of urls*, <https://curlie.org/>, 2019.
- [86] X. Cao, M. Fang, J. Liu, and N. Gong, “Fltrust: Byzantine-robust federated learning via trust bootstrapping,” in *28th Annual Network and Distributed System Security Symposium, NDSS 2021, February 21-25, 2021*, virtually: The Internet Society, 2021. [Online]. Available: <https://www.ndss-symposium.org/ndss-paper/fltrust-byzantine-robust-federated-learning-via-trust-bootstrapping/>.
- [87] A. Jøsang, *Subjective logic*. Springer, 2016.
- [88] C. Iordanou, C. Soriente, M. Sirivianos, and N. Laoutaris, “Who is fiddling with prices? building and deploying a watchdog service for e-commerce,” in *Proceedings of ACM SIGCOMM*, 2017.
- [89] C. Iordanou, N. Kourtellis, J. Carrascosa, C. Soriente, R. Cuevas, and N. Laoutaris, “Beyond content analysis: Detecting targeted ads via distributed counting,” in *Proceedings of the 15th International Conference on Emerging Networking Experiments and Technologies*, 2019, pp. 110–122.
- [90] D. Su, J. Cao, N. Li, E. Bertino, and H. Jin, “Differentially private k-means clustering,” in *Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy*, 2016, pp. 26–37.
- [91] E. Ahn, A. Kumar, D. Feng, M. Fulham, and J. Kim, “Unsupervised deep transfer feature learning for medical image classification,” in *IEEE International Symposium on Biomedical Imaging (ISBI 2019)*, 2019, pp. 1915–1918.

- [92] B. Lin *et al.*, “Fednlp: Benchmarking federated learning methods for natural language processing tasks,” in *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2022.
- [93] M. Naseri, J. Hayes, and E. De Cristofaro, “Local and central differential privacy for robustness and privacy in federated learning,” in *Proceedings of NDSS*, 2022.
- [94] P. Rieger, T. Nguyen, M. Miettinen, and A. Sadeghi, “Deepsight: Mitigating back-door attacks in federated learning through deep model inspection,” in *Proceedings of NDSS*, 2022.
- [95] G. Salton and C. Buckley, “Term-weighting approaches in automatic text retrieval,” *Information Processing and Management*, vol. 24, pp. 513–523, 1988.
- [96] A. Siegel, “Robust regression using repeated medians,” *Biometrika*, vol. 69, pp. 242–244, 1982.
- [97] S. Minsker, “Geometric median and robust estimation in banach spaces,” *Bernoulli*, vol. 21, pp. 2308–2335, 2015.
- [98] H. Lopuhaa and P. Rousseeuw, “Breakdown points of affine equivariant estimators of multivariate location and covariance matrices,” *The Annals of Statistics*, pp. 229–248, 1991.
- [99] R. R. Wilcox, *Introduction to robust estimation and hypothesis testing*. Academic press, 2011.
- [100] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, “Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory,” *IEEE Internet of Things Journal*, vol. 6, pp. 10 700–10 714, 2019.
- [101] A. Jøsang, R. Hayward, and S. Pope, “Trust network analysis with subjective logic,” in *Proceedings of the Twenty-Ninth Australasian Computer Science Conference (ACSW 2006)*, 2006, pp. 85–94.
- [102] C. Xie, S. Koyejo, and I. Gupta, “Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance,” in *International Conference on Machine Learning*, PMLR, 2019, pp. 6893–6901.
- [103] S. Bubeck, “Convex optimization: Algorithms and complexity,” *Foundations and Trends in Machine Learning*, vol. 8, no. 3-4, 2014.
- [104] T. Minka, “Estimating a dirichlet distribution,” MIT, Technical report, 2000.
- [105] M. Barreno, B. Nelson, A. Joseph, and J. Tygar, “The security of machine learning,” *Machine Learning*, vol. 81, pp. 121–148, 2010.
- [106] 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)*, 2019, pp. 691–706.

- [107] B. S. Guendouzi, S. Ouchani, H. El Assaad, and M. El Zaher, “A systematic review of federated learning: Challenges, aggregation methods, and development tools,” *Journal of Network and Computer Applications*, vol. 220, p. 103 714, 2023.
- [108] P. Qi, D. Chiaro, A. Guzzo, M. Ianni, G. Fortino, and F. Piccialli, “Model aggregation techniques in federated learning: A comprehensive survey,” *Future Generation Computer Systems*, vol. 150, pp. 272–293, 2024.
- [109] E. M. El Mhamdi, R. Guerraoui, and S. Rouault, “The hidden vulnerability of distributed learning in Byzantium,” in *Proceedings of the 35th International Conference on Machine Learning*, J. Dy and A. Krause, Eds., ser. Proceedings of Machine Learning Research, vol. 80, PMLR, Jul. 2018, pp. 3521–3530. [Online]. Available: <https://proceedings.mlr.press/v80/mhamdi18a.html>.
- [110] K. Bonawitz *et al.*, “Practical secure aggregation for privacy-preserving machine learning,” in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS ’17, Dallas, Texas, USA: Association for Computing Machinery, 2017, pp. 1175–1191. doi: [10.1145/3133956.3133982](https://doi.org/10.1145/3133956.3133982). [Online]. Available: <https://doi.org/10.1145/3133956.3133982>.
- [111] G. Tsaloli, B. Liang, C. Brunetta, G. Banegas, and A. Mitrokotsa, “SfDEVA: decentralized, verifiable secure aggregation for privacy-preserving learning,” in *Information Security -International Conference, ISC*, ser. Lecture Notes in Computer Science, Springer, 2021. doi: [10.1007/978-3-030-91356-4_16](https://doi.org/10.1007/978-3-030-91356-4_16). [Online]. Available: https://doi.org/10.1007/978-3-030-91356-4_16.
- [112] D. Wu, M. Pan, Z. Xu, Y. Zhang, and Z. Han, “Towards efficient secure aggregation for model update in federated learning,” in *IEEE Global Communications Conference, GLOBECOM*, IEEE, 2020. doi: [10.1109/GLOBECOM42002.2020.9347960](https://doi.org/10.1109/GLOBECOM42002.2020.9347960). [Online]. Available: <https://doi.org/10.1109/GLOBECOM42002.2020.9347960>.
- [113] Z. Chai, Y. Chen, A. Anwar, L. Zhao, Y. Cheng, and H. Rangwala, “Fedat: A high-performance and communication-efficient federated learning system with asynchronous tiers,” in *Proceedings of the international conference for high performance computing, networking, storage and analysis*, 2021, pp. 1–16.
- [114] A. Koloskova, S. U. Stich, and M. Jaggi, “Sharper convergence guarantees for asynchronous sgd for distributed and federated learning,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 17 202–17 215, 2022.
- [115] J. Liu *et al.*, “Adaptive asynchronous federated learning in resource-constrained edge computing,” *IEEE Transactions on Mobile Computing*, vol. 22, no. 2, pp. 674–690, 2023. doi: [10.1109/TMC.2021.3096846](https://doi.org/10.1109/TMC.2021.3096846).
- [116] M. Castro and B. Liskov, “Practical byzantine fault tolerance,” in *USENIX OSDI*, New York, NY, United States: Association for Computing Machinery, 1999, pp. 173–186.

- [117] A. Paszke *et al.*, “Automatic differentiation in pytorch,” in *NIPS 2017 Workshop on Autodiff*, 2017.
- [118] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, Las Vegas, NV, USA: IEEE Computer Society, 2016, pp. 770–778. doi: [10.1109/CVPR.2016.90](https://doi.org/10.1109/CVPR.2016.90). [Online]. Available: <https://doi.org/10.1109/CVPR.2016.90>.
- [119] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE conference on computer vision and pattern recognition*, Ieee, 2009, pp. 248–255.
- [120] *Eitr system*, <https://eitr-experiment.networks.imdea.org>, 2022.
- [121] *Express - fast, unopinionated, minimalist web framework for node.js*, <https://expressjs.com/>, 2021.
- [122] *Tensorflow - an end-to-end open source machine learning platform*, <https://www.tensorflow.org/>, 2021.
- [123] *Keras - simple. flexible. powerful.* <https://keras.io/>, 2021.
- [124] *Tensorflow.js is a library for machine learning in javascript*, <https://www.tensorflow.org/js>, 2021.
- [125] Google, *Google chrome extension apis*, <https://developer.chrome.com/docs/extensions/reference/>, 2021.
- [126] J. Fleiss, “Measuring nominal scale agreement among many raters,” *Psychological Bulletin*, vol. 76, p. 378, 1971.
- [127] Z. Yuan, Z. Guo, Y. Xu, Y. Ying, and T. Yang, “Federated deep auc maximization for heterogeneous data with a constant communication complexity,” in *International Conference on Machine Learning*, 2021, pp. 12 219–12 229.
- [128] S. P. Lalley and E. G. Weyl, “Quadratic voting: How mechanism design can radicalize democracy,” in *AEA Papers and Proceedings*, vol. 108, 2018, pp. 33–37.
- [129] E. A. Posner and E. G. Weyl, “Voting squared: Quadratic voting in democratic politics,” *Vand. L. Rev.*, vol. 68, p. 441, 2015.
- [130] E. G. Weyl, “The robustness of quadratic voting,” *Public choice*, vol. 172, no. 1, pp. 75–107, 2017.
- [131] H. Garcia-Molina, “Elections in a distributed computing system,” *IEEE transactions on Computers*, vol. 31, no. 01, pp. 48–59, 1982.
- [132] M. W. Alford *et al.*, *Distributed systems: methods and tools for specification. An advanced course*. Verlag: Springer, 1985.
- [133] K. Yue, R. Jin, C.-W. Wong, and H. Dai, “Federated learning via plurality vote,” *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2022. doi: [10.1109/TNNLS.2022.3225715](https://doi.org/10.1109/TNNLS.2022.3225715).

- [134] Y. Liu, Z. Fan, X. Song, and R. Shibasaki, “Fedvoting: A cross-silo boosting tree construction method for privacy-preserving long-term human mobility prediction,” *Sensors*, vol. 21, no. 24, p. 8282, 2021.
- [135] J.-y. Sohn, D.-J. Han, B. Choi, and J. Moon, “Election coding for distributed learning: Protecting signsgd against byzantine attacks,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 14 615–14 625, 2020.
- [136] S. Rajput, H. Wang, Z. Charles, and D. Papailiopoulos, “Detox: A redundancy-based framework for faster and more robust gradient aggregation,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [137] L. Chen, H. Wang, Z. Charles, and D. Papailiopoulos, “Draco: Byzantine-resilient distributed training via redundant gradients,” in *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, ser. Proceedings of Machine Learning Research, vol. 80, Stockholmsmässan, Stockholm, Sweden: PMLR, 2018, pp. 902–911. [Online]. Available: <http://proceedings.mlr.press/v80/chen181.html>.
- [138] K. Konstantinidis and A. Ramamoorthy, “Byzshield: An efficient and robust system for distributed training,” *Proceedings of Machine Learning and Systems*, vol. 3, pp. 812–828, 2021.
- [139] G. Sartori, *The theory of democracy revisited*. NJ, 1987, vol. 2.
- [140] S. P. Lalley, E. G. Weyl, *et al.*, “Quadratic voting,” *Available at SSRN*, 2016.
- [141] B. Chandar and E. G. Weyl, “Quadratic voting in finite populations,” *Available at SSRN*: <https://ssrn.com/abstract=2571026> or <http://dx.doi.org/10.2139/ssrn.2571026>, 2019.
- [142] N. Tideman and F. Plassmann, “Efficient collective decision-making, marginal cost pricing, and quadratic voting,” *Public Choice*, vol. 172, no. 1, pp. 45–73, 2017.
- [143] A. Casella and L. Sanchez, “Storable votes and quadratic voting. an experiment on four california propositions,” National Bureau of Economic Research, Tech. Rep., 2019.
- [144] D. Quarfoot, D. von Kohorn, K. Slavin, R. Sutherland, D. Goldstein, and E. Konar, “Quadratic voting in the wild: Real people, real votes,” *Public Choice*, vol. 172, no. 1, pp. 283–303, 2017.
- [145] H. Guo *et al.*, “Siren: Byzantine-robust federated learning via proactive alarming,” in *Proceedings of the ACM Symposium on Cloud Computing*, ser. SoCC ’21, Seattle, WA, USA: Association for Computing Machinery, 2021, pp. 47–60. doi: [10.1145/3472883.3486990](https://doi.org/10.1145/3472883.3486990). [Online]. Available: <https://doi.org/10.1145/3472883.3486990>.

- [146] M. Mansouri, M. Önen, W. B. Jaballah, and M. Conti, “Sok: Secure aggregation based on cryptographic schemes for federated learning,” *Proc. Priv. Enhancing Technol.*, 2023. doi: [10.56553/POPETS-2023-0009](https://doi.org/10.56553/POPETS-2023-0009). [Online]. Available: <https://doi.org/10.56553/popets-2023-0009>.
- [147] S. Goryczka, L. Xiong, and V. S. Sunderam, “Secure multiparty aggregation with differential privacy: A comparative study,” in *EDBT/ICDT Conferences, EDBT/ICDT*, New York, NY, USA: Association for Computing Machinery, 2013. doi: [10.1145/2457317.2457343](https://doi.org/10.1145/2457317.2457343). [Online]. Available: <https://doi.org/10.1145/2457317.2457343>.
- [148] L. Zhao, J. Jiang, B. Feng, Q. Wang, C. Shen, and Q. Li, “Sear: Secure and efficient aggregation for byzantine-robust federated learning,” *IEEE Trans. Dependable Secur. Comput.*, 2022. doi: [10.1109/TDSC.2021.3093711](https://doi.org/10.1109/TDSC.2021.3093711). [Online]. Available: <https://doi.org/10.1109/TDSC.2021.3093711>.
- [149] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai, “Cryptography from anonymity,” in *Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, IEEE Computer Society, 2006. doi: [10.1109/FOCS.2006.25](https://doi.org/10.1109/FOCS.2006.25). [Online]. Available: <https://doi.org/10.1109/FOCS.2006.25>.
- [150] X. Ma, Y. Zhou, L. Wang, and M. Miao, “Privacy-preserving byzantine-robust federated learning,” *Computer Standards & Interfaces*, vol. 80, p. 103 561, 2022.
- [151] J. So, B. Güler, and A. S. Avestimehr, “Byzantine-resilient secure federated learning,” *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 7, pp. 2168–2181, 2020.
- [152] F. Marx, T. Schneider, A. Suresh, T. Wehrle, C. Weinert, and H. Yalame, “Hyfl: A hybrid approach for private federated learning,” *arXiv preprint arXiv:2302.09904*, 2023.
- [153] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, “Learning differentially private recurrent language models,” in *International Conference on Learning Representations*, 2018.
- [154] L. Blumrosen and N. Nisan, *Algorithmic game theory*. Berlin, Heidelberg: Springer, 2007, p. 297.
- [155] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, “Sparse binary compression: Towards distributed deep learning with minimal communication,” in *2019 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2019, pp. 1–8.
- [156] C. Dwork, “Differential privacy,” in *Automata, Languages and Programming: 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II 33*, Berlin, Heidelberg: Springer, 2006, pp. 1–12.

- [157] H. Yu, S. Yang, and S. Zhu, “Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33,01, 2019, pp. 5693–5700.
- [158] X. Cao, J. Jia, Z. Zhang, and N. Gong, “Fedrecover: Recovering from poisoning attacks in federated learning using historical information,” in *2023 IEEE Symposium on Security and Privacy (SP)*, Los Alamitos, CA, USA: IEEE Computer Society, May 2023, pp. 1366–1383. doi: [10.1109/SP46215.2023.00019](https://doi.org/10.1109/SP46215.2023.00019). [Online]. Available: <https://doi.ieeecomputersociety.org/10.1109/SP46215.2023.00019>.
- [159] S. Bubeck *et al.*, “Convex optimization: Algorithms and complexity,” *Foundations and Trends® in Machine Learning*, vol. 8, no. 3-4, pp. 231–357, 2015.
- [160] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, “On the convergence of fedavg on non-iid data,” in *International Conference on Learning Representations (ICLR)*, 2020.
- [161] Y. LeCun, “The mnist database of handwritten digits,” <http://yann.lecun.com/exdb/mnist/>, 1998.
- [162] H. Xiao, K. Rasul, and R. Vollgraf, “Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms,” *arXiv preprint arXiv:1708.07747*, 2017.
- [163] S. Caldas *et al.*, “Leaf: A benchmark for federated settings,” in *33rd Conference on Neural Information Processing Systems (NeurIPS)*, 2019.
- [164] A. Krizhevsky, G. Hinton, *et al.*, “Learning multiple layers of features from tiny images,” 2009.
- [165] T. H. Hsu, H. Qi, and M. Brown, “Measuring the effects of non-identical data distribution for federated visual classification,” *CoRR*, vol. abs/1909.06335, 2019. arXiv: [1909.06335](https://arxiv.org/abs/1909.06335). [Online]. Available: <http://arxiv.org/abs/1909.06335>.
- [166] B. Zhao, P. Sun, T. Wang, and K. Jiang, “Fedinv: Byzantine-robust federated learning by inverting local model updates,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022, 36(8), 9171–9179.
- [167] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg, “Badnets: Evaluating backdooring attacks on deep neural networks,” *IEEE Access*, vol. 7, pp. 47 230–47 244, 2019.
- [168] E. Bagdasaryan and V. Shmatikov, “Blind backdoors in deep learning models,” in *30th USENIX Security Symposium (USENIX Security 21)*, virtually: USENIX Association, 2021, pp. 1505–1521. [Online]. Available: <https://www.usenix.org/conference/usenixsecurity21/presentation/bagdasaryan>.

- [169] V. Shejwalkar, A. Houmansadr, P. Kairouz, and D. Ramage, “Back to the drawing board: A critical evaluation of poisoning attacks on production federated learning,” in *2022 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2022, pp. 1354–1371.
- [170] T. J. Feddersen, I. Sened, and S. G. Wright, “Rational voting and candidate entry under plurality rule,” *American Journal of Political Science*, pp. 1005–1016, 1990.
- [171] W. Diffie and M. E. Hellman, “New directions in cryptography,” *IEEE Trans. Inf. Theory*, 1976. doi: [10.1109/TIT.1976.1055638](https://doi.org/10.1109/TIT.1976.1055638). [Online]. Available: <https://doi.org/10.1109/TIT.1976.1055638>.
- [172] A. Shamir, “How to share a secret,” *Commun. ACM*, 1979. doi: [10.1145/359168.359176](https://doi.org/10.1145/359168.359176). [Online]. Available: <https://doi.org/10.1145/359168.359176>.
- [173] S. T. V. Setty, “Spartan: Efficient and general-purpose zkSNARKs without trusted setup,” in *Annual International Cryptology Conference, CRYPTO*, ser. Lecture Notes in Computer Science, Springer, 2020. doi: [10.1007/978-3-030-56877-1_25](https://doi.org/10.1007/978-3-030-56877-1_25). [Online]. Available: https://doi.org/10.1007/978-3-030-56877-1_25.
- [174] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, “A survey on homomorphic encryption schemes: Theory and implementation,” *ACM Comput. Surv.*, 2018. doi: [10.1145/3214303](https://doi.org/10.1145/3214303). [Online]. Available: <https://doi.org/10.1145/3214303>.
- [175] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, “Federated learning: Challenges, methods, and future directions,” *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020. doi: [10.1109/MSP.2020.2975749](https://doi.org/10.1109/MSP.2020.2975749).
- [176] S. Han, J. Pool, J. Tran, and W. Dally, “Learning both weights and connections for efficient neural network,” in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 28, 2015.
- [177] T. Chen *et al.*, “The lottery ticket hypothesis for pre-trained bert networks,” *Advances in neural information processing systems (NeurIPS)*, vol. 33, pp. 15 834–15 846, 2020.
- [178] X. Ma *et al.*, “Sanity checks for lottery tickets: Does your winning ticket really win the jackpot?” *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 34, pp. 12 749–12 760, 2021.
- [179] Y. LeCun, J. Denker, and S. Solla, “Optimal brain damage,” in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 2, 1989.
- [180] S. Han, H. Mao, and W. J. Dally, “Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding,” in *International Conference on Learning Representations (ICLR)*, 2016.

- [181] J.-H. Park, Y. Kim, J. Kim, J.-Y. Choi, and S. Lee, “Dynamic structure pruning for compressing cnns,” in *The Thirty-Seventh AAAI Conference on Artificial Intelligence*, 2023.
- [182] J. Frankle and M. Carbin, “The lottery ticket hypothesis: Finding sparse, trainable neural networks,” in *International Conference on Learning Representations(ICLR)*, 2019.
- [183] A. Li *et al.*, *Lotteryfl: Personalized and communication-efficient federated learning with lottery ticket hypothesis on non-iid datasets*, 2020. arXiv: [2008.03371 \[cs.LG\]](#).
- [184] H. Yin, A. Mallya, A. Vahdat, J. M. Alvarez, J. Kautz, and P. Molchanov, “See through gradients: Image batch recovery via gradinversion,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 16 337–16 346.
- [185] K. Yue, R. Jin, C.-W. Wong, D. Baron, and H. Dai, “Gradient obfuscation gives a false sense of security in federated learning,” in *Proceedings on USENIX Security*, 2023.
- [186] M. Du, R. Jia, and D. Song, “Robust anomaly detection and backdoor attack detection via differential privacy,” in *International Conference on Learning Representations(ICLR)*, 2020.
- [187] A. Girgis, D. Data, S. Diggavi, P. Kairouz, and A. T. Suresh, “Shuffled model of differential privacy in federated learning,” in *International Conference on Artificial Intelligence and Statistics*, PMLR, 2021, pp. 2521–2529.
- [188] X. Zhang, X. Chen, M. Hong, Z. S. Wu, and J. Yi, “Understanding clipping for federated learning: Convergence and client-level differential privacy,” in *International Conference on Machine Learning, ICML 2022*, 2022.
- [189] A. Triastcyn and B. Faltings, “Federated learning with bayesian differential privacy,” in *2019 IEEE International Conference on Big Data (Big Data)*, IEEE, 2019, pp. 2587–2596.
- [190] X. Yang, W. Huang, and M. Ye, “Dynamic personalized federated learning with adaptive differential privacy,” *Advances in Neural Information Processing Systems*, vol. 36, pp. 72 181–72 192, 2023.
- [191] K. Bonawitz *et al.*, “Practical secure aggregation for privacy-preserving machine learning,” ser. CCS ’17, Dallas, Texas, USA: Association for Computing Machinery, 2017, pp. 1175–1191. doi: [10.1145/3133956.3133982](#). [Online]. Available: <https://doi.org/10.1145/3133956.3133982>.
- [192] J. So, R. E. Ali, B. Güler, J. Jiao, and A. S. Avestimehr, “Securing secure aggregation: Mitigating multi-round privacy leakage in federated learning,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, 2023, pp. 9864–9873.

- [193] J. So *et al.*, “Lightsecagg: A lightweight and versatile design for secure aggregation in federated learning,” *Proceedings of Machine Learning and Systems*, vol. 4, pp. 694–720, 2022.
- [194] A. Herzog, R. Southam, I. Mavromatis, and A. Khan, “Fedmap: Iterative magnitude-based pruning for communication-efficient federated learning,” *arXiv preprint arXiv:2406.19050*, 2024.
- [195] Y. Shi *et al.*, “Efficient federated learning with enhanced privacy via lottery ticket pruning in edge computing,” *IEEE Transactions on Mobile Computing*, 2024.
- [196] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, “Learning differentially private recurrent language models,” in *International Conference on Learning Representations(ICRL)*, 2018.
- [197] A. R. Elkordy, J. Zhang, Y. H. Ezzeldin, K. Psounis, and S. Avestimehr, “How much privacy does federated learning with secure aggregation guarantee?” In *Proceedings on Privacy Enhancing Technologies (PoPETs)*, 2023.
- [198] M. I. Belghazi *et al.*, “Mutual information neural estimation,” in *International Conference on Machine Learning(ICML)*, PMLR, 2018, pp. 531–540.
- [199] P. Bachman, R. D. Hjelm, and W. Buchwalter, “Learning representations by maximizing mutual information across views,” *Advances in neural information processing systems*, vol. 32, 2019.
- [200] Z. Zhu, J. Wu, B. Yu, L. Wu, and J. Ma, “The anisotropic noise in stochastic gradient descent: Its behavior of escaping from sharp minima and regularization effects,” in *Proceedings of the 36th International Conference on Machine Learning*, K. Chaudhuri and R. Salakhutdinov, Eds., ser. Proceedings of Machine Learning Research, vol. 97, PMLR, Jun. 2019, pp. 7654–7663. [Online]. Available: <https://proceedings.mlr.press/v97/zhu19e.html>.
- [201] M. Thomas and A. T. Joy, *Elements of information theory*. Wiley-Interscience, 2006.
- [202] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
- [203] C. J. Maddison, A. Mnih, and Y. W. Teh, “The concrete distribution: A continuous relaxation of discrete random variables,” in *International Conference on Learning Representations(ICLR)*, 2017.
- [204] E. Jang, S. Gu, and B. Poole, “Categorical reparametrization with gumble-softmax,” in *International Conference on Learning Representations(ICLR)*, OpenReview.net, 2017.
- [205] Pytorch, *Gumbel softmax in pytorch documentation*, https://pytorch.org/docs/stable/generated/torch.nn.functional.gumbel_softmax.html, [Online].