Byzantine-Resilient Federated Learning via Distributed Optimization

Yufei Xia
Institut Polytechnique de Paris
France
yufei.xia@ip-paris.fr

Wenrui Yu
Aalborg University
Denmark
wenyu@es.aau.dk

Qiongxiu Li*
Aalborg University
Denmark
qili@es.aau.dk

***Abstract*—Byzantine attacks present a critical challenge to Federated Learning (FL), where malicious participants can disrupt the training process, degrade model accuracy, and compromise system reliability. Traditional FL frameworks typically rely on aggregation-based protocols for model updates, leaving them vulnerable to sophisticated adversarial strategies. In this paper, we demonstrate that distributed optimization offers a principled and robust alternative to aggregation-centric methods. Specifically, we show that the Primal-Dual Method of Multipliers (PDMM) inherently mitigates Byzantine impacts by leveraging its fault-tolerant consensus mechanism. Through extensive experiments on three datasets (MNIST, FashionMNIST, and Olivetti), under various attack scenarios including bit-flipping and gaussian noise injection, we validate the superior resilience of distributed optimization protocols. Compared to traditional aggregation-centric approaches, PDMM achieves higher model utility, faster convergence, and improved stability. Our results highlight the effectiveness of distributed optimization in defending against Byzantine threats, paving the way for more secure and resilient FL systems.**

***Index Terms*—Federated learning, distributed optimization, Byzantine robustness, privacy, security.**

I. Introduction

FL enables collaborative model training across distributed clients without exposing raw data [1], [2]. By design, FL addresses privacy concerns inherent in centralized data collection, making it particularly appealing for critical applications such as healthcare, IoT, and edge computing. A typical FL workflow involves three steps: 1) initializing a global model, 2) local client updates, and 3) aggregating updates into a shared model. Existing FL protocols can be broadly categorized along two axes: topology and protocol [3]. By topology, FL can be centralized (CFL) or decentralized (DFL). In CFL, a central server aggregates updates using protocols like FedAvg and FedSGD [1], whereas DFL relies on peer-to-peer communication [4], [5]. Despite its advantages in privacy preservation, FL remains vulnerable to privacy attacks [6]–[9], where adversaries aim to reconstruct sensitive data from shared model updates. While the privacy implications of various FL protocols have been extensively studied [10]–[13], their robustness against Byzantine attacks remains underexplored.

Byzantine attacks, where malicious clients submit falsified updates to degrade or manipulate the global model, pose a critical threat to FL systems [14], [15]. Current literature predominantly focuses on attacks targeting aggregation-based protocols like FedAvg, where adversaries exploit the server's reliance on averaging to inject skewed gradients. Common attack strategies include bit-flipping [16], gaussian noise injection [17], [18], and label flipping [19]. To defend against these attacks, robust aggregation rules such as Trimmed Mean [20] and Krum [21] have been proposed. However, these approaches remain reactive, often relying on outlier detection and assumptions about the redundancy or majority of honest clients. Moreover, they do not fundamentally address the inherent vulnerability of aggregation-centric FL protocols to Byzantine behavior. An open question remains: are alternative FL protocols, particularly those based on joint optimization, also susceptible to Byzantine attacks? Distributed optimization methods such as the Alternating Direction Method of Multipliers (ADMM) [22], [23] and PDMM [24]–[26] have demonstrated convergence guarantees and fault tolerance in distributed systems. However, their resilience to Byzantine behavior in FL settings has not been thoroughly investigated.

In this paper, we bridge this gap by systematically analyzing the Byzantine resilience of distributed optimization based FL protocols. Unlike aggregation-based approaches that separate local training from global aggregation, joint optimization protocols synchronize client updates through iterative consensus mechanisms and dual variable constraints. We hypothesize that these mechanisms inherently limit adversarial impact by enforcing global agreement and leveraging redundancy in distributed subproblems. To validate this,we conduct the first empirical comparison between FedAvg (representing aggregation-based protocols) and PDMM (representing joint optimization) under Byzantine attack scenarios, including bit flipping and gaussian noise injection. Experiments on MNIST, FashionMNIST, and Olivetti datasets demonstrate that PDMM achieves superior accuracy and stability across both CFL and DFL topologies. These results confirm that joint optimization protocols inherently constrain Byzantine impacts.

Our work challenges the prevailing focus on post-hoc defenses for aggregation-based FL and highlights the underexplored potential of protocol-level robustness. By demonstrating that distributed optimization algorithms, such as ADMM and PDMM, naturally mitigate Byzantine failures, we advocate for rethinking FL protocol design in adversarial environments.

*Corresponding author

II. PRELIMINARY

This section introduces the necessary fundamentals and the notations used throughout the paper are summarized in Table I.

A. Aggregation based FL

A widely adopted aggreagation protocol in FL, known as *FedAvg* [1], operates as follows:

1) **Initialization**: At iteration $t = 0$, the server randomly initializes the global model weights $\mathbf{w}_s^{(0)}$ and distributes them to all participating nodes.
2) **Local Update**: At each iteration t, each node i receives the current global model $\mathbf{w}_s^{(t-1)}$ and updates its local model $\mathbf{w}_i^{(t)}$ based on its private dataset $\mathbf{x}_i$. The resulting local model updates are then transmitted back to the server.
3) **Model Aggregation**: Upon receiving the local updates, the server aggregates them to form an updated global model:

$$\mathbf{w}_s^{(t)} = \sum_{i=1}^{N} a_i \mathbf{w}_i^{(t)},$$

where a_i is the aggregation weight assigned to client i and $\sum_{i=1}^{N} a_i = 1$. Steps 2 and 3 are repeated until the global model converges or a predefined stopping criterion is satisfied. Another commonly used protocol is *FedSGD* [1], which assumes full-batch training on the client side and performs similar aggregation strategy.

B. Byzantine Attacks

Byzantine robustness has been a major research concern in FL due to the presence of malicious or unreliable clients. Byzantine clients can deliberately inject false updates or non-honest data, compromise global model accuracy and delay convergence. In this paper, we focus on two representative and widely studied Byzantine attack models:

- **Bit Flipping Attack** [16]: Malicious clients invert the sign of their model updates to counteract the contributions of honest clients. Formally, for a Byzantine client $i \in \mathcal{B}$, the update is:

$$\hat{\mathbf{w}}_i^{(t+1)} = -\mathbf{w}_i^{(t+1)}$$

  Bit flipping attacks can severely hinder or entirely prevent convergence, particularly when a significant fraction of clients are adversarial.
- **Gaussian Noise Attack** [17], [18]: Malicious clients inject random gaussian noise into their model updates to introduce instability and disrupt learning:

$$\hat{\mathbf{w}}_i^{(t+1)} = \mathbf{w}_i^{(t+1)} + \mathcal{N}(0, \sigma^2)$$

  Unlike bit flipping, which systematically misguides the optimization process, gaussian noise injection adds excessive randomness, preventing meaningful learning progress and increasing variance in model updates.

These types of Byzantine attacks are particularly disruptive to aggregation-based FL protocols, such as FedAvg and FedSGD, which generally lack mechanisms to defend against malicious updates. Common defense strategies include robust aggregation methods such as Trimmed Mean [20] and Krum [21], which aim to filter out outliers and mitigate the influence of compromised clients.

In this work, we demonstrate that distributed optimization methods (e.g., ADMM or PDMM) can offer inherent robustness against Byzantine attacks. By jointly optimizing local updates and enforcing global consensus constraints, A/PDMM limits the ability of malicious clients to degrade learning performance. The following sections provide a detailed description of our methodology and experimental validation.

TABLE I: Notation Table

Symbol	Description
$\mathbf{w}_i$	Local model parameter at client i
$\mathbf{w}_s$	Global model parameter at sever
$\mathbf{y}_{i\|j}$	Transmission variable in PDMM from client i to j
$\mathcal{B}$	Set of Byzantine clients
$\mathcal{N}$	Set of clients
$\mathcal{E}$	Set of edges
$f_i(\mathbf{w})$	Loss function at client i
t	Iteration
c	PDMM penalty parameter
η	Learning rate
σ	Standard deviation of gaussian noise
$\hat{(\cdot)}$	Malicious behavior from Byzantine clients

III. METHODOLOGY

This section presents the proposed Byzantine-resilient FL framework based on distributed optimization. Besides, the attack models are also detailed with formulas.

A. Distributed Optimization in FL

Distributed optimization addresses global learning objectives across decentralized networks. The general formulation is expressed as the following constrained optimization problem:

$$\begin{aligned} \min_{\{\mathbf{w}_i\}} \quad & \sum_{i\in\mathcal{N}} f_i(\mathbf{w}_i), \\ \text{subject to} \quad & \mathbf{B}_{i|j}\mathbf{w}_i + \mathbf{B}_{j|i}\mathbf{w}_j = \mathbf{0}, \quad \forall (i,j) \in \mathcal{E}, \end{aligned}$$

The matrices $\mathbf{B}_{i|j}$ define linear edge constraints that ensure consensus among neighboring nodes. Typically, $\mathbf{B}_{i|j} = -\mathbf{B}_{j|i} = \pm\mathbf{I}$ to ensure that all nodes converge to the same model.

B. PDMM based FL

The PDMM algorithm introduces auxiliary variables to enforce consensus and stabilize model updates. Each client optimizes its local model while exchanging information with a central server (in CFL) or directly with peers (in DFL) to maintain consistency. Algorithm 1 outlines PDMM for CFL. The corresponding version for DFL follows the same structure but restricts communication to neighboring nodes [27].

Algorithm 1 PDMM-based CFL

Require: Penalty parameter $c > 0$, number of clients N

Initialize: $\mathbf{w}_i^{(0)} = \mathbf{w}_s^{(0)}$, $\mathbf{y}_{i|j}^{(0)}$ for all $i \in \{1, \ldots, N\}$
while not converged **do**
  for all client i **in parallel do** ▷ Primal update
    $\mathbf{w}_i^{(t+1)} = \arg\min_{\mathbf{w}_i} \left[f_i(\mathbf{w}_i) + \frac{c}{2} \|\mathbf{w}_i - \mathbf{y}_{s|i}^{(t)}\|^2 \right]$
    $\mathbf{y}_{i|s}^{(t+1)} = 2\mathbf{w}_i^{(t+1)} - \mathbf{y}_{s|i}^{(t)}$
  Server: Aggregate models ▷ Second update
  $\mathbf{w}_s^{(t+1)} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{y}_{i|s}^{(t+1)}$
  for all client i **do**
    Update auxiliary variable $\mathbf{y}_{s|i}^{(t+1)}$ of server:
    $\mathbf{y}_{s|i}^{(t+1)} = 2\mathbf{w}_s^{(t+1)} - \mathbf{y}_{i|s}^{(t)}$
  $t \leftarrow t + 1$

The server is denoted by s and $\mathbf{w}_s$ is the model of server. The server will initialize the model for each node i as the same in FedAvg. The joint optimization aggregation of PDMM is divided into two steps. Each client i performs a local model update by minimizing the loss function, which includes a term involving the auxiliary variables of server $\mathbf{y}_{s|i}$ to regulate the local model update, where c is the regulation parameter. The update is given by:

$$\mathbf{w}_i^{(t+1)} = \arg\min_{\mathbf{w}_i} \left[f_i(\mathbf{w}_i) + \frac{c}{2} \|\mathbf{w}_i - \mathbf{y}_{s|i}^{(t)}\|^2 \right]$$

After the local model update, each client updates its auxiliary variable $\mathbf{y}_{i|s}$ based on its local model:

$$\mathbf{y}_{i|s}^{(t+1)} = 2\mathbf{w}_i^{(t+1)} - \mathbf{y}_{s|i}^{(t)}$$

Then the server aggregates the models from all clients by computing the average:

$$\mathbf{w}_s^{(t+1)} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{y}_{i|s}^{(t+1)}$$

At the end of each round, the server updates the global auxiliary variables based on the aggregated models:

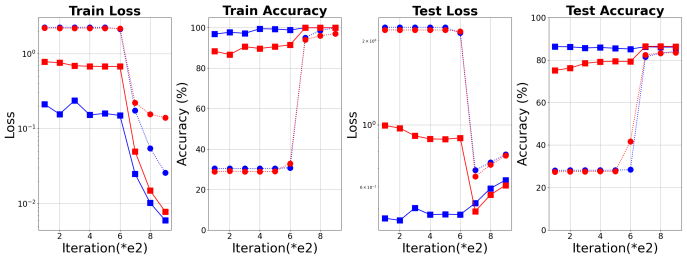
$$\mathbf{y}_{s|i}^{(t+1)} = 2\mathbf{w}_s^{(t+1)} - \mathbf{y}_{i|s}^{(t)}$$

C. Analysis of Byzantine robustness

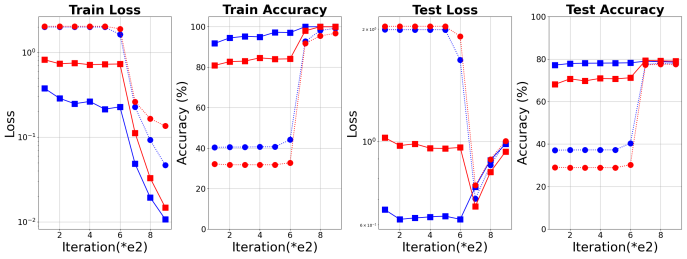
PDMM offers inherent robustness against Byzantine attacks through its joint optimization framework and consensus enforcement mechanism. By integrating local updates with global consistency constraints, PDMM mitigates the influence of malicious clients without relying on external defense mechanisms. Unlike aggregation-based methods such as FedAvg and FedSGD, which are vulnerable to adversarial manipulation during the model aggregation step, PDMM stabilizes local updates by penalizing deviations from consensus through auxiliary variable updates. This structure inherently limits the ability of Byzantine clients to disrupt the learning process.

Theoretical analyses in [28], [29] demonstrate that PDMM can tolerate finite perturbations while maintaining convergence guarantees. Inexact Krasnosel'skiĭ–Mann iteration is a classical fixed-point iterative method designed for non-expansive operators. It can modulate the distributed optimization process with certain noise. Especially, Proposition 1(iii) and Theorem 1 in [29] proved that with summable errors, the iteration error will converge to 0 and iteration parameter $\mathbf{y}$ will converge to a fixed point $\mathbf{y}^*$. Therefore, as long as the number of attack rounds or the magnitude of injected noise remains bounded, the algorithm ensures the integrity and stability of the global model. In real-world scenarios, adversaries may not always confine themselves to bounded perturbations. Nevertheless, when confronted with extreme outliers or malicious updates with extreme values, the whole system networks will be disintegration and fail to continue training, thereby ensuring effective defense.

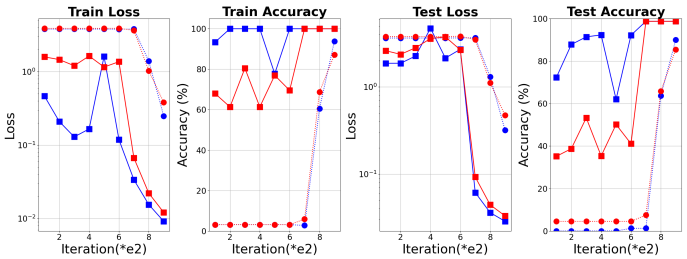
Fig. 1: Index of Figures



(a) MNIST



(b) FashionMNIST

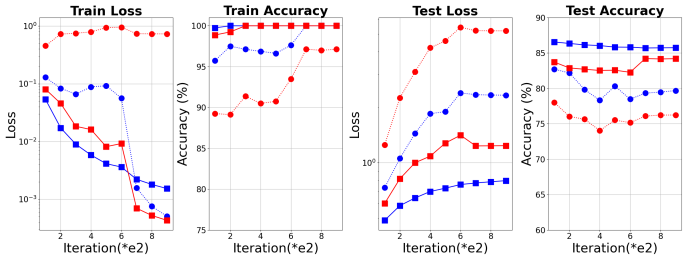


(c) Olivetti

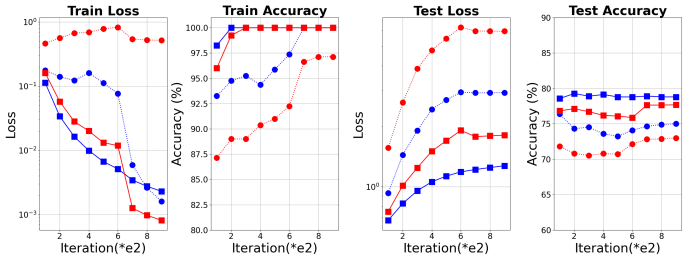
Fig. 2: Byzantine robustness comparison of PDMM based FL protocols over traditional FedAvg based protocols against bit flipping attack over three datasets and two types of topologies (CFL v.s. DFL).

IV. Numerical Results

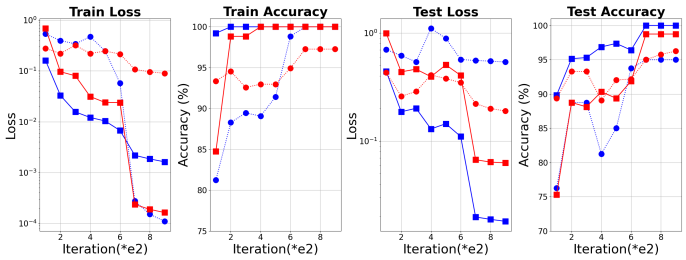
In this section, we present empirical evaluations that demonstrate the effectiveness of distributed optimization based FL



(a) MNIST



(b) FashionMNIST



(c) Olivetti

Fig. 3: Byzantine robustness comparison of PDMM based FL protocols over traditional FedAvg based protocols against gaussian noise attack over three datasets and two types topologies (CFL v.s., DFL).

protocols in mitigating Byzantine attacks. Specifically, we compare distributed optimization method(PDMM) against traditional aggregation based FL framework(FedAvg) under various adversarial conditions.

A. Experimental setup

We evaluate the robustness of PDMM and FedAvg against 2 Byzantine attacks by conducting experiments on MNIST [30], FashionMNIST [31], and Olivetti [32] datasets. Olivetti includes a set of 400 facial images of 40 people. The used network is a 2 layer multilayer perception (MLP) structure. Experiments involve 10 clients, with 2 Byzantine nodes launching attacks in the first 600 rounds over 1000 training rounds. The data is independently and identically distributed across each client. We consider a random geometric graph of $n = 10$ nodes [33] as the DFL topology. In each iteration, we use 10 steps of gradient descent for both CFL and DFL. For MNIST and FashionMNIST, the learning rate is set as $\eta = 0.05$ and $\sigma = 0.1$; for Olivetti, we set $\eta = 0.04$ and $\sigma = 0.2$.

We consider four FL architectures: (1) CFL with PDMM (centralized, joint optimization), (2) CFL with FedAvg (centralized, separate aggregation), (3) DFL with PDMM (decentralized, joint optimization), and (4) DFL with FedAvg (decentralized, separate aggregation). The decentralized version of FedAvg follows the same procedure as in the centralized setting but replaces the global server aggregation with local neighborhood aggregation, which we also refer to as FegAvg in our comparisons.

To implement the bit-flipping and gaussian noise attacks, adversarial manipulations are applied by either flipping the values or injecting noise into the transmitted updates in each protocol—specifically, the shared model parameters in FedAvg and the exchanged auxiliary variables in PDMM. The legend index of different FL classification is shown in Figure 1.

B. Performance comparison under bit flipping Attacks

Figure 2 presents the performance comparison between PDMM-based protocols (solid lines) and FedAvg-based protocols (dotted lines) under bit-flipping attacks. In this scenario, two malicious nodes execute bit-flipping attacks across 600 out of 1000 attack rounds.

From the plots, it is evident that distributed optimization protocols (denoted as DFL_PDMM and CFL_PDMM) consistently outperform aggregation-based methods (CFL_FedAvg and CFL_FedAvg). Specifically, DFL_PDMM achieves substantially lower training and test loss, while maintaining higher training and test accuracy across all three datasets. For instance, under bit-flipping attacks in CFL configuration, PDMM outperforms FedAvg by 37% on FashionMNIST and 56% on MNIST for test accuracy. As for test accuracy of DFL structure, PDMM is 40% higher for FashionMNIST and 37% higher for MNIST than FedAvg. The results demonstrate that distributed optimization protocols inherently mitigate the detrimental effects of Byzantine nodes, ensuring stable convergence and robust performance even under sustained adversarial interference.

C. Performance comparison under gaussian noise attacks

Figure 3 shows the results for the same FL systems under gaussian noise injection attacks. Again, two malicious nodes perform attacks over 600 out of 1000 rounds. As illustrated in the figure, distributed optimization protocols outperform aggregation-based methods in both convergence speed and final model utility. DFL_PDMM exhibits remarkable stability, with significantly reduced variance in both accuracy and loss metrics. Under gaussian noise attack, the test accuracy of PDMM is 8.75% higher than FedAvg for CFL structure with Olivetti datasets. Even under intense gaussian noise perturbations, the distributed optimization framework consistently achieves superior robustness and convergence compared to traditional aggregation-centric FL protocols. It is noteworthy that, the results on Olivetti exhibit larger fluctuations compared to others. This may be attributed to the greater diversity and higher feature dimensionality of Olivetti, which may lead unstable training outcomes.

D. Performance superiority is independent of topology type

A key observation from our experiments is that the superiority of distributed optimization based FL protocols is largely

independent of the network topology. Whether implemented in CFL or DFL settings, distributed optimization protocols consistently outperform aggregation-based approaches.

This robustness is attributable to the inherent fault-tolerant design of distributed optimization methods like PDMM. By distributing computation and information exchange more equitably among nodes, and by mitigating the influence of any single malicious actor, these protocols reduce the risk posed by Byzantine behaviors. Our empirical results clearly demonstrate that across varying topologies and attack strategies, distributed optimization remains the more resilient and effective approach.

V. Conclusion

In this paper, we demonstrate that distributed optimization protocols, such as PDMM, offer inherent robustness against Byzantine attacks in FL. Unlike traditional aggregation-based methods, PDMM enforces consensus through joint optimization, effectively limiting the impact of malicious clients. Our empirical results, validated across diverse datasets and attack scenarios, show that PDMM consistently outperforms traditional aggregation-based protocols like FedAvg in both CFL and DFL architectures. Notably, PDMM achieves higher accuracy, faster convergence, and greater stability under common Byzantine attacks, including bit-flipping and gaussian noise injection. These findings highlight the potential of protocol-level robustness and suggest a shift away from reactive defenses in FL.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Aguera y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artif. Intell. Statist.* PMLR, pp. 1273–1282, 2017.

[2] T. Li, A. Kumar Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, 2020.

[3] Q. Li, W. Yu, Y. Xia, and J. Pang, "From centralized to decentralized federated learning: Theoretical insights, privacy preservation, and robustness challenges," 2025.

[4] K. Niwa, N. Harada, G. Zhang, and W. B. Kleijn, "Edge-consensus learning: Deep learning on p2p networks with nonhomogeneous data," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery & Data Mining*, pp. 668–678, 2020.

[5] Q. Li, W. Yu, C. Ji, and R. Heusdens, "Topology-dependent privacy bound for decentralized federated learning," in *ICASSP 2024-2024 IEEE Int. Conf. Acous. Speech Signal Process.(ICASSP)*. IEEE, pp. 5940–5944, 2024.

[6] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients-how easy is it to break privacy in federated learning?," *Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 16937–16947, 2020.

[7] H. Yin, A. Mallya, A. Vahdat, J. M. Alvarez, J. Kautz, and P. Molchanov, "See through gradients: Image batch recovery via gradinversion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pp. 16337–16346, 2021.

[8] H. Yang, M. Ge, K. Xiang, and J. Li, "Using highly compressed gradients in federated learning for data reconstruction attacks," *IEEE Trans. Inf. Forensics Secur.*, vol. 18, pp. 818–830, 2022.

[9] Q. Li, L. Luo, A. Gini, C. Ji, Z. Hu, X. Li, C. Fang, J. Shi, and X. Hu, "Perfect gradient inversion in federated learning: A new paradigm from the hidden subset sum problem," *arXiv preprint arXiv:2409.14260*, 2024.

[10] Q. Li, M. L. Zwakenberg, W. Yu, and R. Heusdens, "On the privacy bound of distributed optimization and its application in federated learning," in *2024 32nd Eur. Signal Process. Conf.(EUSIPCO)*. IEEE, pp. 2232–2236, 2024.

[11] D. Pasquini, M. Raynal, and C. Troncoso, "On the (in) security of peer-to-peer decentralized machine learning," in *2023 IEEE Symp. Secur. Privacy (SP)*. IEEE, pp. 418–436, 2023.

[12] W. Yu, R. Heusdens, J. Pang, and Q. Li, "Privacy-preserving distributed maximum consensus without accuracy loss," in *Proc. Int. Conf. Acoust. Speech, Signal Process.*, 2025.

[13] C. Ji, R. Heusdens, S. Maag, and Q. Li, "Re-evaluating privacy in centralized and decentralized learning: An information-theoretical and empirical study," *arXiv preprint arXiv:2409.14261*, 2024.

[14] J. So, B. Güler, and A. S. Avestimehr, "Byzantine-resilient secure federated learning," *IEEE J. on Sel. Areas Commun.*, vol. 39, no. 7, pp. 2168–2181, 2020.

[15] S. Praneeth Karimireddy, L. He, and M. Jaggi, "Byzantine-robust learning on heterogeneous datasets via bucketing," *arXiv e-prints*, pp. arXiv–2006, 2020.

[16] A. S. Rakin, Z. He, and D. Fan, "Bit-flip attack: Crushing neural network with progressive bit search," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, pp. 1211–1220, 2019.

[17] K. Wei, J. Li, M. Ding, C. Ma, H. Yang, F. Farokhi, S. Jin, T.y Q. Quek, and H. V. Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 3454–3469, 2020.

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

[19] N. Moharram Jebreel, J. Domingo-Ferrer, D. Sánchez, and A. Blanco-Justicia, "Defending against the label-flipping attack in federated learning," *arXiv e-prints*, pp. arXiv–2207, 2022.

[20] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Int. Conf. Mach. Learn.* Pmlr, pp. 5650–5659, 2018.

[21] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.

[22] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al., "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.

[23] P. Giselsson and S. Boyd, "Linear convergence and metric selection for douglas-rachford splitting and admm," *IEEE Trans. Autom. Control*, vol. 62, no. 2, pp. 532–544, 2016.

[24] G. Zhang and R. Heusdens, "Distributed optimization using the primal-dual method of multipliers," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 4, no. 1, pp. 173–187, 2017.

[25] T. W. Sherson, R. Heusdens, and W. B. Kleijn, "Derivation and analysis of the primal-dual method of multipliers based on monotone operator theory," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 5, no. 2, pp. 334–347, 2018.

[26] R. Heusdens and G. Zhang, "Distributed optimisation with linear equality and inequality constraints using pdmm," *IEEE Trans. Signal Inf. Process. Netw.*, 2024.

[27] G. Zhang, K. Niwa, and W. B. Kleijn, "Revisiting the primal-dual method of multipliers for optimisation over centralised networks," *IEEE Trans. Signal Inf. Process. Over Netw.*, vol. 8, pp. 228–243, 2022.

[28] J. A. Jonkman, T. Sherson, and R. Heusdens, "Quantisation effects in distributed optimisation," in *2018 IEEE Int. Conf. Acoust. Speech Signal Process.(ICASSP)*. IEEE, pp. 3649–3653, 2018.

[29] J. Liang, J. Fadili, and G Peyré, "Convergence rates with inexact non-expansive operators," *Math. Program.*, vol. 159, pp. 403–434, 2016.

[30] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[31] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[32] Scikit learn Developers, "Olivetti faces dataset," https://scikit-learn.org/0.19/datasets/olivetti_faces.html#, 2017, Accessed: 2024-03-13.

[33] J. Dall and M. Christensen, "Random geometric graphs," *Physical Rev. E*, vol. 66, no. 1, pp. 016121, 2002.