TS-FedNBS: Federated Edge Computing with Enhanced Robustness

Alina Basharat[1] Runhua Wang[2] Ping Xu[1]
[1] University of Texas Rio Grande Valley
[2] Sun Yat-Sen University

***Abstract*—Edge-assisted federated learning (FedEdge) that integrates an intermediate layer of edge nodes to reduce the workload for central server in traditional federated learning systems has been investigated in this work. However, the existing FedEdge mechanisms may be vulnerable to adversarial attackers. In this paper, we propose a two-stage robust aggregation scheme (TS-FedNBS) to enhance the resilience of FedEdge against Byzantine attackers. Specifically, TS-FedNBS employs a norm based screening (NBS) at the edge nodes and a median aggregation at the central server. Experimental results on real datasets indicate that the proposed method significantly enhances the resilience of FedEdge systems against Byzantine adversaries.**

***Index Terms*—FedEdge, Byzantine attack, two-stage robust aggregation scheme.**

I. Introduction

Recent advancements in deep learning have significantly transformed various application domains, including image processing, natural language processing, and video analytics [1]. Traditionally, deep learning models are trained on robust computing platforms, such as cloud data centers, utilizing centrally collected large-scale datasets. However, in numerous applications, data are generated from distributed end devices, such as smartphones and sensors. Transferring these data to a central server for model training raises not only substantial privacy concerns but also incurs high communication costs.

Alternatively, federated learning (FL) emerged as a promising paradigm, which allows mobile devices to collaboratively train a global model without sharing their private data [2]. However, standard FL architectures suffer from intensive communication between clients and the central server, which introduces latency and leads to congestion when scaling to a large number of clients [3]. To resolve this issue, edge-assisted federated learning (FedEdge) has been proposed, which integrates edge computing with FL so that data can be processed at edge nodes that are close to its origin [4]. The learned local models are then transmitted to the central server for aggregation. Compared with vanilla FL, FedEdge enjoys low communication latency and lighter workload on the central server.

Still, vanilla FedEdge is vulnerable to potential Byzantine attackers, which maliciously send altered or adversarial messages to the edge nodes or the central server with the purpose to corrupt the whole system [5], [6]. To combat Byzantine attackers, several robust aggregation methods have been developed. For example, the distance-based Krum algorithm selects reliable updates based on the client's score, which is a summation of the pairwise Euclidean distance of its update and that of the remaining clients [7]. The effects of median and trimmed mean techniques to mitigate the influence of outliers have been studied in [8], which validate the reliability of robust statistical methods in FL against adversarial attacks. The norm based screening (NBS) method excludes suspicious attackers by cropping gradients with large norms [9]. On the other hand, the performance-based detection scheme allows a central server to utilize a clean dataset to evaluate the performance of updates from local clients and to eliminate updates that do not perform well [6]. The performance of various robust aggregation methods in FedEdge with heterogeneous data distributions has been evaluated and compared in [5].

Built upon existing work, the focus of this paper is to develop a robust FedEdge algorithm, particularly in the presence of adversarial attacks. By processing data and conducting preliminary aggregations at the edge, our framework aligns with current advancements in FedEdge that ensure the efficient handling of data traffic. The robustness of the proposed algorithm stems from the utilization of a two-stage robust aggregation approach. In stage one, we utilize the NBS technique at edge nodes to screen out model updates from potential attackers that have large norms [9]. In stage two, we employ a median aggregation rather than a vanilla averaging at the central server to add one more layer of protection against Byzantine attackers. The intuition of stage two's robust aggregation is based on observations that current robust aggregation methods at edge nodes might not be sufficient to ensure a no-loss performance and the model updates transmitted from edge nodes to the central server are also likely to be attacked.

To summarize, our contributions include: 1) We highlight the importance of robust aggregation at the central server in FedEdge, a topic that has received limited attention in the current literature. 2) We propose a two-stage robust aggregation algorithm, TS-FedNBS, which combines NBS at edge nodes with median aggregation at the central server to enhance the robustness of FedEdge against Byzantine attackers. 3) We demonstrate the effectiveness of TS-FedNBS through extensive experiments involving diverse datasets and multiple attack scenarios, showing its ability to resist Byzantine attacks without compromising accuracy.

The rest of this paper is organized as follows. Section II states the problem. Section III describes the proposed method.

2833-8898/24/$31.00 ©2024 IEEE
DOI 10.1109/CyberC62439.2024.00015

Section IV presents the simulation results and Section V concludes the whole paper.

II. Problem Statement

We consider an edge-based FL architecture consisting of a client layer, an edge node layer, and a central server, as depicted in Figure 1 [6]. In this architecture, E edge nodes (e.g., base stations) are positioned close to their respective groups of clients and each edge node is responsible for aggregating model updates from its local clients. With the learning process conducted near the data source at the edge layer, the communication load and latency are reduced for the central server [10]. The central server, positioned at the top of the hierarchy, integrates updates from all edge nodes to obtain the global model, which is then sent back to edge nodes for them to distribute to their local clients for further update.

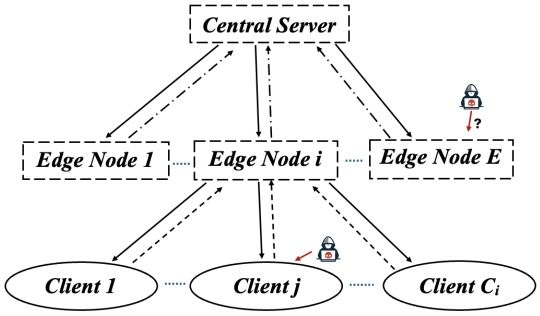


Fig. 1: Edge system model with Byzantine attacks.

The i-th edge subsystem formed by edge node i ($\forall i \in \{1, 2, ...E\}$) and its C_i local clients involves the following local computation and aggregation process. First, each client $j \in \{1, 2, \ldots, C_i\}$ conducts local computation on its dataset $\mathcal{D}_{i,j}$ and independently optimizes a local loss function $\mathcal{L}_{i,j}(\mathbf{w})$ parameterized by $\mathbf{w} \in \mathbb{R}^d$. The model update for client j in edge subsystem i at iteration t is obtained via gradient descent:

$$\mathbf{w}_{i,j}^t = \mathbf{w}_{i,j}^{t-1} - \eta \nabla \mathcal{L}_{i,j}(\mathbf{w}_{i,j}^{t-1}; \mathcal{D}_{i,j}), \tag{1}$$

where η is the learning rate.

Then all updated model parameters $\mathbf{w}_{i,j}^t$ are transmitted to edge node i, which then performs a vanilla averaging or a weighted averaging to aggregate all received updates $\mathbf{w}_{i,j}^t$:

$$\mathbf{w}_i^t = \begin{cases} \sum_{j=1}^{C_i} \frac{1}{C_i} \mathbf{w}_{i,j}^t, & \text{vanilla average} \\ \sum_{j=1}^{C_i} \frac{|\mathcal{D}_{i,j}|}{\sum_{j=1}^{C_i} |\mathcal{D}_{i,j}|} \mathbf{w}_{i,j}^t, & \text{weighted average} \end{cases}. \tag{2}$$

The updated model $\mathbf{w}_i^t, \forall i$, is then transmitted to the central server, where a vanilla averaging or a weighted averaging is adopted to obtain an updated global model $\mathbf{w}^t$:

$$\mathbf{w}^t = \begin{cases} \sum_{i=1}^{E} \frac{1}{E} \mathbf{w}_i^t, & \text{vanilla average} \\ \sum_{i=1}^{E} \frac{\sum_{j=1}^{C_i} |\mathcal{D}_{i,j}|}{\sum_{i=1}^{E} \sum_{j=1}^{C_i} |\mathcal{D}_{i,j}|} \mathbf{w}_i^t, & \text{weighted average} \end{cases}. \tag{3}$$

The global model $\mathbf{w}^t$ is then sent back to edge nodes and clients for further training, completing the feedback loop.

However, there may exist Byzantine attackers at the client level or at the edge level, that deliberately introduce manipulated or malicious updates into the system to disrupt the learning process, as shown in Figure 1. Therefore, the primary objective of this work is to develop a robust algorithm to mitigate the impact of such attackers.

III. Proposed Method

In the FedEdge framework depicted in Figure 1, Byzantine attackers can exist in the client level, which bias the aggregated models at edge nodes if no action is taken. Robust aggregation algorithms are then developed at the edge layer to combat these attackers [5]. Still, it is not guaranteed that all Byzantine attackers are excluded. Moreover, there also exist possibilities that the edge nodes can be attacked by malicious parties. Therefore, in this paper, we propose to develop a two-stage algorithm that utilizes norm based screening (NBS) at the edge level coupled with the mean aggregation at the central server to enhance FedEdge's robustness.

Specifically, each client j ($\forall j \in \{1, 2, \ldots, C_i\}$) in subsystem i ($\forall i \in \{1, 2, \ldots, E\}$) updates its model using (1). These updates $\mathbf{w}_{i,j}^t$ are then sent to edge node i. However, the Byzantine attackers exist in the client level may modify these updates, thereby interrupt the learning process. Some common attack types include:

- **Gaussian Attack** that adds random Gaussian noise to benign clients' updates, modifying $\mathbf{w}_{i,j}^t$ into

$$\tilde{\mathbf{w}}_{i,j}^t = \mathbf{w}_{i,j}^t + \mathcal{N}(0, \sigma^2 \mathbf{I}), \tag{4}$$

  where $\mathcal{N}(0, \sigma^2 \mathbf{I})$ denotes the Gaussian noise with 0 mean and a variance of σ^2.
- **Inner product manipulation (IPM) Attack** that replaces the update by a negative average of all benign clients multiplied by a strength control scalar S:

$$\tilde{\mathbf{w}}_{i,j}^t = -\frac{S}{C_i} \sum_{j=1}^{C_i} \mathbf{w}_{i,j}^t, \tag{5}$$

  with an assumption that the attacker has full information of the updates of all devices [5].
- **Omniscient Attack** that negatively multiplies the agent' update with an attack strength scalar S:

$$\tilde{\mathbf{w}}_{i,j}^t = -S \mathbf{w}_{i,j}^t. \tag{6}$$

Therefore, instead of the vanilla averaging or weighted averaging (2), in stage one, edge node i utilizes NBS (7) for aggregation to mitigate the negative effects of Byzantine attackers. NBS first computes the norms of all received updates $\|\mathbf{w}_{i,j}^t\|$, then the βC_i updates with highest norms are removed, and the remaining updates are averaged to get the aggregated update $\mathbf{w}_i^t$, i.e.,

$$\mathbf{w}_i^t = \frac{1}{|\mathcal{U}|} \sum_{j=1}^{|\mathcal{U}|} \mathbf{w}_{i,j}^t, \tag{7}$$

where $\mathcal{U} = \{(1), \ldots, ((1-\beta)C_i)\}$ is an index set that specifies the unscreened updates. The intuition behind NBS is that

malicious updates tend to deviate significantly from the benign updates with extremely large norms. With NBS, each edge node is able to defend against Byzantine attacks launched by up to βC_i adversaries.

Then, in stage two, the central server utilizes the median aggregation to integrate updates $\mathbf{w}_i^t$ received from all edge nodes i ($\forall i \in \{1, 2, \ldots, E\}$):

$$\mathbf{w}^t = \text{median}(\{\mathbf{w}_i^t | i \in \{1, 2, \ldots, E\}\}). \tag{8}$$

The updated global model $\mathbf{w}_t$ is then distributed back to all clients for next iteration's computation and update. Notice that the additional robust aggregation is especially necessary when any of the edge nodes in stage one fails to completely defend against attackers. For example, if the percentage of attackers in edge system i is higher than the screen ratio β, then the aggregated model $\mathbf{w}_i^t$ is polluted by the remaining attackers that are not screened out, which in turn ruins the global model $\mathbf{w}_t$ aggregated at the central server. The developed two-stage robust FedEdge via norm-based screening (TS-FedNBS) algorithm is outlined in Algorithm 1.

Remark: While the FedEdge structure has been widely adopted in edge computing applications and various robust aggregation methods have been developed to enhance resilience against adversarial attacks [11], [12], the critical role of robust aggregation at the central server has not been sufficiently emphasized or studied.

Algorithm 1 TS-FedNBS: Two-stage robust FedEdge via norm-based screening

Input: Communication round T; screen ratio β; learning rate η.

Initialize global model parameter $\mathbf{w}^0$
Distribute $\mathbf{w}^0$ to all clients
for communication round $t = 1, 2, \ldots, T$ **do**
  for client $j = 1, \ldots, C_i$ **do**
    receives global model parameter $\mathbf{w}^{t-1}$ from edge node i
    updates its local model via (1)
    sends $\mathbf{w}_{i,j}^t$ to edge node i
  end for
  for node $i = 1, \ldots, E$ **do**
    computes norms of all received updates $\|\mathbf{w}_{i,j}^t\|$
    removes the top β fraction with the highest norms
    averages the remaining updates via (7)
    sends $\mathbf{w}_i^t$ to the central server
  end for
  for central server **do**
    receives updates $\{\mathbf{w}_i^t\}$ from all edge nodes
    performs median aggregation via (8)
    distributes $\mathbf{w}^t$ to all edge nodes
  end for
end for

IV. Simulation Results

A. Experimental setting

In our experiments, we utilize a convolutional neural network (CNN) model optimized for image classification tasks, applicable for MNIST [13] and FMNIST [14] datasets. The FedEdge system consists of 50 clients and 5 edge nodes, where each edge node has 10 clients and each client has 250 training data.

The baseline scenario is implemented without any attackers (*No attack*). Each client trains its local model on its local data and sends the updates to edge nodes, which then aggregate these updates and forward them to the central server. In the presence of attackers, we set the fraction of attackers to be 30% in each edge subsystem. The reported experiments do not consider attackers at the edge node level. However, the effect of robust aggregations at the central server in the case that edge nodes are attacked can be concluded from the performance of robust aggregations at the edge node in the case that clients are attacked. We use *No attack* as a reference to evaluate the impact of attacks (i.e., *Without robust aggregation*) and the effectiveness of the proposed approach (*TS-FedNBS*). Throughout the experiments, the learning rate is $\eta = 0.01$. The remaining hyperparameters are $\sigma^2 = 100$ for Gaussian attack, $S = 20$ for IPM attack, and $S = 10$ for Omniscient attack, respectively.

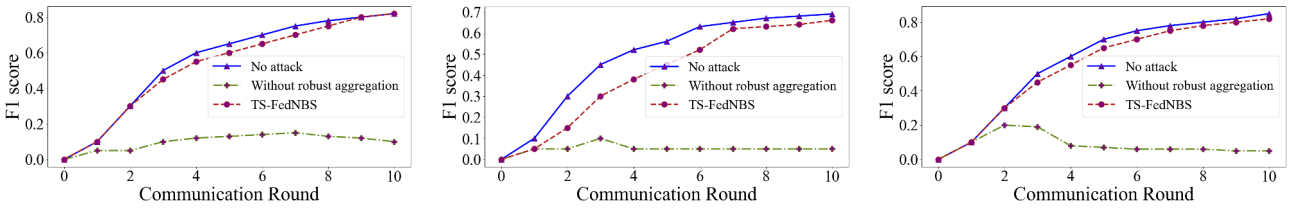
B. Evaluation results

We use F1 score to evaluate the performance of various algorithms, which is the harmonic mean of precision (p) and recall (r) to compute the score, as shown in (9) [5]:

$$F1 = 2 \times \frac{p \times r}{p + r}. \tag{9}$$

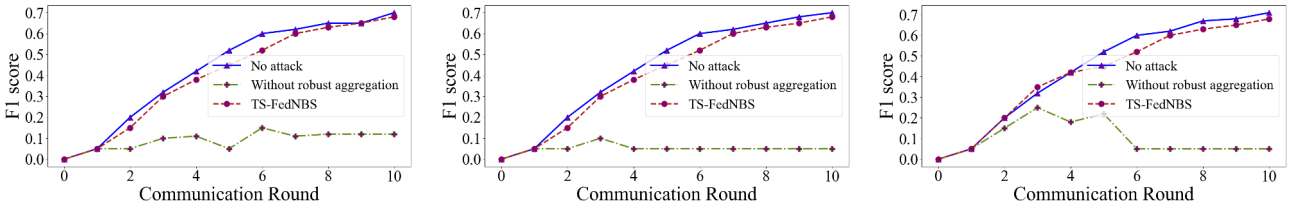
Figure 2 shows the performance of TS-FedNBS on the MNIST dataset under three adversarial scenarios. It can be seen that TS-FedNBS approaches the no-attack performance by the end of the communication rounds while the system will be totally disrupted if no action is taken. Similar phenomena can be observed in Figure 3 for the FashionMNIST dataset. While the Gaussian attack (Figure 3a) and the IPM attack (Figure 3b) initially impact the model accuracy, TS-FedNBS effectively mitigates these effects with performance closely mirrors the no attack scenario.

To show the superiority of the propose algorithm, we compare TS-FedNBS with three other algorithms. The FedAegis algorithm is adapted from [5], which applies the median aggregation at edge nodes and a simple average at the central server while the TS-FedAegis algorithm applies the median aggregation at both the edge nodes and the central server. FedNBS applies NBS at the edge nodes and a simple average at the central server. Both FedAegis and FedNBS are one-stage robust aggregation method. From Table I we can see that in the case that attackers exist only in the client level, one-stage median aggregation (FedAegis) is better than one-stage NBS (FedNBS) and two-stage median aggregation (TS-FedAegis). However, if attackers exist in the edge node level,



(a) Gaussian attack (b) IPM attack (c) Omniscient attack

Fig. 2: Test accuracy on the MNIST dataset under different attack scenarios.



(a) Gaussian attack (b) IPM attack (c) Omniscient attack

Fig. 3: Test accuracy on the FashionMNIST dataset under different attack scenarios.

Attack Type	MNIST				FashionMNIST			
	Fedaegis	TS-Fedaegis	FedNBS	TS-FedNBS	Fedaegis	TS-Fedaegis	FedNBS	TS-FedNBS
No attack	85%	81%	87%	**88%**	63%	58%	59%	**64%**
Gaussian attack	83%	74%	67%	**85%**	58%	48%	36%	**63%**
IPM attack	81%	75%	78%	**86%**	61%	54%	46%	**63%**
Omniscient attack	79%	80%	65%	**87%**	61%	56%	52%	**63%**

TABLE I: Test accuracies of various methods under different attacks.

FedAegis will totally fail while TS-FedAegis can still achieve a learning performance that is close to the no-attack case. The performance gap between FedNBS and TS-FedNBS indicates that the two-stage implementation is essential.

V. Conclusion

This paper proposes TS-FedNBS, a novel approach that integrates edge computing with a two-stage robust aggregation method to enhance resilience against Byzantine attacks, using norm-based screening (NBS) at edge nodes and median aggregation at the central server. Future research aims to establish the theoretical foundations of the TS-FedNBS algorithm.

References

[1] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*. PMLR, 2017, pp. 1273–1282.

[3] L. Liu, J. Zhang, S. Song, and K. B. Letaief, "Client-edge-cloud hierarchical federated learning," in *IEEE International Conference on Communications (ICC)*. IEEE, 2020, pp. 1–6.

[4] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1205–1221, 2019.

[5] F. Zhou, R. Yu, Z. Li, H. Gu, and X. Wang, "Fedaegis: Edge-based byzantine-robust federated learning for heterogeneous data," in *IEEE Global Communications Conference*. IEEE, 2022, pp. 3005–3010.

[6] W. Liu, X. Xu, D. Li, L. Qi, F. Dai, W. Dou, and Q. Ni, "Privacy preservation for federated learning with robust aggregation in edge computing," *IEEE Internet of Things Journal*, vol. 10, no. 8, pp. 7343–7355, 2022.

[7] 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 (NeurIPS)*, vol. 30, 2017.

[8] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *International Conference on Machine Learning (ICML)*. PMLR, 2018, pp. 5650–5659.

[9] G. Zhou, P. Xu, Y. Wang, and Z. Tian, "H-nobs: Achieving certified fairness and robustness in distributed learning on heterogeneous datasets," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 36, 2024.

[10] Z. Zhang, L. Wu, C. Ma, J. Li, J. Wang, Q. Wang, and S. Yu, "Lsfl: A lightweight and secure federated learning scheme for edge computing," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 365–379, 2022.

[11] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Network*, vol. 33, no. 5, pp. 156–165, 2019.

[12] Z. Wang, H. Xu, J. Liu, H. Huang, C. Qiao, and Y. Zhao, "Resource-efficient federated learning with hierarchical aggregation in edge computing," in *IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.

[13] L. Schott, J. Rauber, M. Bethge, and W. Brendel, "Towards the first adversarially robust neural network model on mnist," *arXiv preprint arXiv:1805.09190*, 2018.

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