Frontiers in Computing and Intelligent Systems
ISSN: 2832-6024 | Vol. 7, No. 1, 2024

PPBRFL: Privacy-Preserving Byzantine-Robust Federated Learning

Qun Zhou *

College of Computer Science and Technology, Qingdao University, Qingdao, Shandong, China
* **Corresponding author Email**: zhouqun1999@163.com

Abstract: Federated learning is a distributed machine learning approach that allows the neural network to be trained without exposing private user data. Despite its advantages, federated learning schemes still face two critical security challenges: user privacy disclosure and Byzantine robustness. The adversary may try to infer the private data from the trained local gradients or compromise the global model update. To tackle the above challenges, we propose PPBRFL, a privacy-preserving Byzantine-robust federated learning scheme. To resist Byzantine attacks, we design a novel Byzantine-robust aggregation method based on cosine similarity, which can guarantee the global model update and improve the model's classification accuracy. Furthermore, we introduce a reward and penalty mechanism that considers users' behavior to mitigate the impact of Byzantine users on the global model. To protect user privacy, we utilize symmetric homomorphic encryption to encrypt the users' trained local models, which requires low computation cost while maintaining model accuracy. We conduct the experimental assessment of the performance of PPBRFL. The experimental results show that PPBRFL maintains model classification accuracy while ensuring privacy preservation and Byzantine robustness compared to traditional federated learning scheme.

Keywords: Federated Learning; Byzantine Robustness; Privacy-preserving; Data Privacy.

1. Introduction

MACHINE learning has demonstrated exceptional performance in various data-driven applications, such as financial analysis [1], medical diagnosis [2] and speech recognition [3]. In the field of finance, machine learning techniques are used to predict stock market trends, while in the medical field, they are employed for disease diagnosis. The development of machine learning is predominantly attributed to the abundance of extensive datasets. In machine learning, training a powerful and accurate neural network model requires a large amount of user data. However, utilizing traditional centralized machine learning for training can lead to the leakage of user data privacy [4].

In 2016, Google introduced the concept of federated learning [5]. Federated learning is a distributed machine learning training technique that can effectively tackle user data privacy concerns [6]. In the training process of federated learning, users locally train the neural network with their respective datasets. After the training is completed, the trained local gradients are sent to the cloud server for aggregation. In this process, the user data privacy can be protected since the user's private data does not require to be uploaded to the cloud server. However, some subsequent studies [7], [8] have demonstrated that the cloud server can still infer the users' private training data from the trained local gradients sent by the users. Subsequently, many privacy-preserving federated learning schemes [8-14] have been proposed to address the privacy implications in federated learning. These schemes encrypt the users' trained local gradients by using cryptographic primitives such as homomorphic encryption [7], [10-11], differential privacy [14] and secure multi-party computation [9], [12-13]. Furthermore, federated learning may be vulnerable to Byzantine attacks [15]. Byzantine users can upload false parameters to the cloud server, thereby compromising global model's accuracy or manipulating the global model to produce specific outputs through poisoning attacks [10]. Such actions can severely undermine the robustness of the federated learning. To resist Byzantine attacks, a lot of Byzantine-robust federated learning schemes have been constructed. Blanchard et al. [16] calculated the Euclidean distance between two users to filter out the aggregated users. Yin et al. [17] utilized the mean and median of user models as the global model for updating. Cao et al. [18] combined cosine similarity with the Relu function to assign model update weight to each user. However, most of the existing Byzantine-robust federated learning schemes do not consider the issue of privacy, which leads to privacy leakage.

Robustness and privacy are two critical issues that need to be solved in federated learning. In this paper, we propose a novel privacy-preserving Byzantine-robust federated learning scheme (PPBRFL), which can achieve privacy protection and resist Byzantine attacks. Furthermore, PPBRFL allows users to drop out in the training phase.

Contribution. The contributions of this paper are summarized as follows:

(1) We construct PPBRFL, a novel scheme to achieve privacy protection and Byzantine robustness. We design a novel Byzantine-robust aggregation method based on cosine similarity. Specifically, each user computes the cosine similarity of local model and trained model. Based on the cosine similarity values of users, the cloud server selects a subset of users and aggregates these selected users' encrypted trained models instead of aggregating the encrypted trained models of all users. Using this aggregation method, PPBRFL is resistant to Byzantine attacks, thus improving the model's classification accuracy. Furthermore, we present a reward and penalty mechanism that takes into account the behavior of the users to mitigate the influence of Byzantine users on the global model and safeguard the fairness of users.

(2) We use symmetric homomorphic encryption to encrypt the users' trained local models and achieve privacy

protection of user data. Compared with traditional homomorphic encryption, symmetric homomorphic encryption reduces the computation cost while maintaining the accuracy of the model.

(3) We evaluate the performance of PPBRFL by conducting experiments on two well-known datasets and comparing PPBRFL with existing state-of-the-art schemes. The experimental results indicate that PPBRFL exhibits superior performance in terms of both classification accuracy and efficiency, while achieving privacy protection and Byzantine robustness.

2. Related Works

Currently, federated learning has garnered substantial attention across various industries and is being adopted in a growing array of applications [19]. The extensive adoption of federated learning accentuates the significance of privacy and security concerns associated with this approach. Privacy leakage in federated learning occurs mainly through the intermediate gradients uploaded by users during model training [20]. Therefore, the crucial aspect of safeguarding privacy lies in encrypting the gradients or models uploaded by users. To achieve private protection, many privacy-preserving federated learning schemes [21-24] are proposed. Bonawitz et al. [6] achieve privacy protection through secret sharing techniques in federated learning. Phong et al. [7] proposed a privacy-preserving federated learning scheme based on additive homomorphic encryption that is able to train the model without loss of accuracy. Ma et al. [25] used two-trapdoor homomorphic encryption to protect the users' data privacy. Fang et al. [26] utilized ElGamal multiplicative homomorphic encryption technology to construct a privacy-preserving federated learning scheme. Differential privacy is a commonly used privacy-preserving technique in federated learning. Jia et al. [27] and Zhou et al. [28] used differential privacy to protect the user's data privacy. Based on differential privacy and secure multi-party computation techniques, Mugunthan et al. [29] designed a privacy-preserving federated learning scheme by applying a combination of these two techniques to federated learning. Zhou et al. [30] introduced a privacypreserving federated learning approach, which introduces a trusted blinding server to blind the gradient ciphertexts in the federated learning process. Moreover, this scheme can effectively solve the problem of collusion between cloud servers and users.

To resist Byzantine user attacks and improve Byzantine robustness in federated learning, a number of Byzantine-robust federated learning schemes [15], [31-35] have been proposed. Blanchard et al. [16] introduced the Krum scheme, which selects specific users for gradient aggregation based on their Euclidean distance, aiming to reduce the impact of Byzantine users on the accuracy of trained models in federated learning. In Yin et al.'s scheme [17], each component of the global model is updated individually, with each part being updated by taking the average or median of the local models from all users. This approach helps to mitigate the impact caused by Byzantine users. Wang et al. [36] designed a model segmentation aggregation strategy to defend against attacks by Byzantine users. In Wang et al. [36]'s scheme, the classification accuracy can be improved by using pairwise adjusted cosine similarity. Li et al. [37] constructed a Byzantine-robust federated learning scheme based on an Auto-weighted geometric median method. Cao et al. [18] combined the Relu function with cosine similarity to assign model update weight to each user. In this way, honest users are allocated larger update weights while malicious Byzantine users are assigned smaller update weights, thereby enhancing the Byzantine robustness of this scheme. In Tang et al.'s scheme [38], the server can determine whether the users are honest or not by verifying the gradients uploaded by these users. In this way, the server can avoid using the wrong gradients sent by the Byzantine users for aggregation, which prevents the Byzantine users from influencing the training of federated learning. However, the above Byzantine-robust federated learning schemes cannot ensure the users' data privacy. Therefore, it is meaning to explore a federated learning scheme that can protect users' data privacy while maintaining Byzantine robustness.

3. The Construction of PPBRFL

3.1. System Model

As shown in Fig.1, PPBRFL is mainly composed of three entities: Key generator center (KGC), users and cloud server.

- **KGC:** KGC is an honest entity that is responsible for generating the keys and the public parameters for the users.
- **Users:** The users train the model with their own local datasets. Then, the user calculates the value of cosine similarity between the local model and the trained model, and encrypts the trained model. Finally, the user sends the encrypted trained model along with the cosine similarity value to the cloud server.
- **Cloud server:** The cloud server is responsible for selecting some users and aggregating the encrypted trained models of these selected users. Specifically, the cloud server divides the users into three subgroups, then sorts the values of cosine similarity of these users. Then, the cloud server selects some users from each subgroup for aggregation. The cloud server aggregates the encrypted trained models of the selected users to get a new encrypted global model. Finally, the cloud server sends the new encrypted global model to each user for the next round of training.

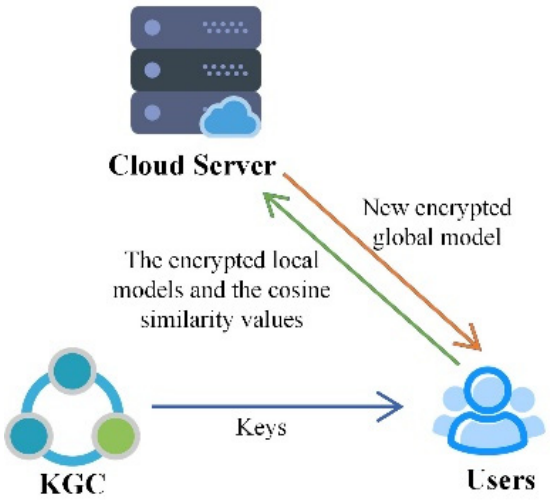


Figure 1. System model of PPBRFL

3.2. Proposed Scheme

(1) Initialization phase

In PPBRFL, a user group U contains n users U_i $(i \in [0, n-1])$ with logical identities $\{0,1,\ldots,n-1\}$. Firstly, the cloud server initializes the global model W_g and learning rate η of federated learning, then sends the global model W_g and learning rate η to each user of the user group U. KGC selects two random prime numbers p and q, generates the key $sk = (p, \tau)$ for symmetric homomorphic encryption (SHE) and sends it to users U_i $(i \in [0, n-1])$ in the user group U. KGC picks four values k_0, k_1, k_2, N used for SHE, where k_0 denotes the length of two prime numbers p, q, k_1 denotes length of plaintext, k_2 denotes the length of a random number τ, and N denotes the product of two prime numbers p and q. KGC exposes the public parameters (k_0, k_1, k_2, N).

(2) Training phase

After receiving the global model W_g from the cloud server, the user U_i $(i \in [0, n-1])$ uses the received global model W_g as his/her local training model W_i^l. The user U_i $(i \in [0, n-1])$ trains the local model W_i^l using its own local dataset $D_i = \{< x_l, y_l >\}_{l \in [1,k]}$. The loss function calculated by the user U_i $(i \in [0, n-1])$ is as follows:

$$L_\varphi(D_i^*, W_i^l) = \frac{1}{|D_i^*|} \sum_{(x_l, y_l) \in D_i^*} \theta(\varphi(x_l, W_i^l), y_l)$$

where D_i^* is the subset of D_i, $\varphi()$ represents the neural network, W_i^l represents trainable model in $\varphi()$, and $\theta()$ represents the difference between the real label y_l and the neural network output $\varphi(x_l, W_i^l)$. The user U_i $(i \in [0, n-1])$ calculates the local gradient and updates the local model W_i^l with the following equations: $\omega_i = \nabla L_\varphi(D_i^*, W_i^l)$ and $W_i = W_i^l - \eta\omega_i$.

After training, the trained model W_i is encrypted with symmetric homomorphic encryption. The formula for encryption is as follows: $[[W_i]] = SHE.enc(W_i, p, \tau)$. To exclude malicious users, the user calculates the cosine similarity between the trained model W_i and the local model W_i^l. The cosine similarity is calculated as follows:

$$sim(W_i, W_i^l) = \frac{W_i \cdot W_i^l}{\|W_i\| \|W_i^l\|}$$

The user U_i $(i \in [0, n-1])$ sends the encrypted trained model $[[W_i]]$ along with the cosine similarity value $sim(W_i, W_i^l)$ to the cloud server.

(3) Verification and aggregation phase

The cloud server receives the encrypted trained model $[[W_i]]$ and cosine similarity $sim(W_i, W_i^l)$ sent by user U_i $(i \in [0, n-1])$. The cloud server computes a new logical identifier $j = \pi(i)$ for each user $(i \in [0, n-1])$ by using the pseudo-random permutation $\pi(): \{0,1,\cdots,n-1\} \to \{0,1,\cdots,n-1\}$, where i is original logical identifier. The new user group is $U^* = \{U_j \mid j \in [0, n-1]\}$. The cloud server divides the new user group U^* into three subgroups: $\mu_1 = \{U_j \mid j \equiv 0 \bmod 3\}$, $\mu_2 = \{U_j \mid j \equiv 1 \bmod 3\}$ and $\mu_3 = \{U_j \mid j \equiv 2 \bmod 3\}$. The cloud server sorts the cosine similarity values of the three user subgroups in descending order, then removes the users whose cosine similarity values are in the top $\alpha_0\%$ of the three subgroups. In this way, the Byzantine users with excessively large cosine similarity values can be removed. Generally, these users can directly upload untrained local model, do not contribution to the global model during federated learning, but still obtain a new global model after each training.

The cloud server selects users from three user subgroups μ_1, μ_2 and μ_3 based on different ranges.

1) For the user subgroup μ_1:

The cloud server selects users in user subgroup μ_1 whose cosine similarity values are ranked between $\alpha_0 + 1\%$ % and $\alpha_1\%$ in descending order. The set of selected users is denoted as μ_1^*.

2) For the user subgroup μ_2:

The cloud server selects users in user subgroup μ_2 whose cosine similarity values are ranked between $\alpha_1 + 1\%$ and $\alpha_2\%$ in descending order. The set of selected users is denoted as μ_2^*.

3) For the user subgroup μ_3:

The cloud server selects users in user subgroup μ_3 whose cosine similarity values are ranked between $\alpha_2 + 1\%$ and $\alpha_3\%$ in descending order. The set of selected users is denoted as μ_3^*.

The users whose cosine similarity values rank in the bottom $(100 - \alpha_3)\%$ of the three subgroups are identified as Byzantine users since their cosine similarity values are significantly small. Generally, these users with excessively small cosine similarity can upload random local model, or flip the element-wise signs of the local model and upload the flipped local model. These users also do not contribution to the global model during federated learning, but still obtain a new global model after each training.

The cloud server aggregates the encrypted trained models of the above selected users to obtain a new encrypted global model $[[W_g^*]]$ as follows:

$$[[W_g^*]] = \frac{(\sum_{j\in\mu_1^*}[[W_j]] + \sum_{j\in\mu_2^*}[[W_j]] + \sum_{j\in\mu_2}[[W_j]])}{|\mu_1^*| + \mu_2^*| + |\mu_3^*|}$$

The cloud server sends the new encrypted global model $[[W_g^*]]$ to each user. After receiving the new encrypted global model $[[W_g^*]]$, the user first decrypts the encrypted global model to get the global model $W_g^* = SHE.dec([[W_g^*]], p, \tau)$, and then proceeds to the next round of training.

Note: The reason for selecting users from the three disjoint intervals of the three user subgroups is to reduce the risk of a successful attack by Byzantine users. Even if a Byzantine user succeeds in faking a cosine similarity that closely resembles that of other users, the probability of that Byzantine user being selected is greatly reduced.

4. Performance Evaluation

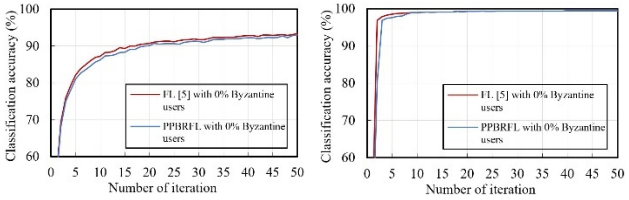
4.1. Classification Accuracy

In this section, we evaluate the performance of our PPBRFL through the following experiments. These experiments are carried out on a Linux server with Intel(R) Core (TM) i9-10980XE, 3.0GHZ, and 32GB of RAM. The simulation environment is implemented using the PyTorch framework.

The evaluation of our simulation experiments focuses on two aspects: classification accuracy and computation efficiency on the user side.

We evaluate PPBRFL using MNIST [39] and CIFAR10 [40] datasets.

MNIST dataset [39]: The MNIST is a dataset of handwritten numbers. It contains a total of 70,000 images, with 60,000 images selected for training and 10,000 images for testing. Each image has a size of 28x28 pixels and features handwritten digits ranging from 0 to 9, displayed in white on a black background.

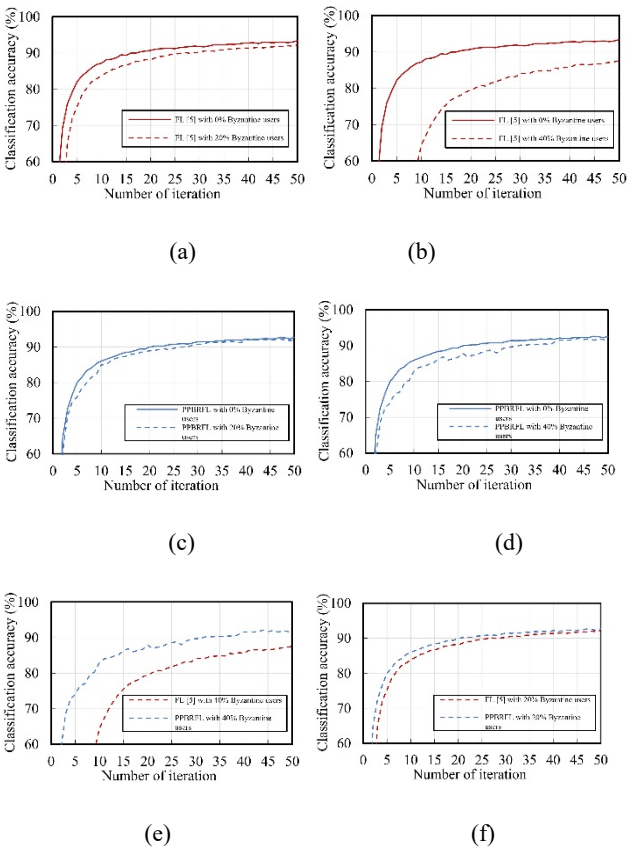


(a) (b)

Figure 2. Comparison of classification accuracy on the CIFAR-10 and MNIST datasets between PPBRFL and FL [5] (a) On the CIFAR-10 dataset. (b) On the MNIST dataset

CIFAR-10 dataset [40]: The CIFAR-10 dataset is an image dataset consisting of various items. It comprises a total of 60,000 samples, with 10 categories, each having 6,000 samples (5000 for training and 1000 for testing). Each sample in the dataset is a 32x32 pixel RGB image with three channels.

In this experiment, ResNet18 [41] neural network is trained using the above two datasets. ResNet18 [41] is a residual neural network which is composed of a series of residual blocks. It comprises 17 internal convolutional layers and 1 fully connected layer. ResNet18 neural network is widely used for image classification tasks. In the following, we choose the standard federated learning scheme [5], which does not support Byzantine robustness and privacy protection, as a benchmark for comparison with the proposed PPBRFL. In order to evaluate the impact of the proposed PPBRFL on the model classification accuracy, we conduct a comparative analysis between PPBRFL and the scheme [5]. This analysis is executed under conditions where Byzantine users are absent, allowing us to exclude the impact of Byzantine users on classification accuracy.



(a) (b)

(c) (d)

(e) (f)

Figure 3. Comparison of classification accuracy on the CIFAR-10 dataset between PPBRFL and FL [5] under Byzantine attack. (a) FL [5] with 0% Byzantine users and FL [5] with 20% Byzantine users. (b) FL [5] with 0% Byzantine users and FL [5] with 40% Byzantine users. (c) PPBRFL with 0% Byzantine users and PPBRFL with 20% Byzantine users. (d) PPBRFL with 0% Byzantine users and PPBRFL with 40% Byzantine users. (e) PPBRFL with 20% Byzantine users and FL [5] with 20% Byzantine users. (f) PPBRFL with 40% Byzantine users and FL [5] with 40% Byzantine users

Fig. 2(a) and Fig. 2(b) illustrate the model classification accuracy results of PPBRFL and the scheme [5] on the CIFAR-10 and MNIST datasets, respectively. The comparison reveals that the classification accuracy achieved by PPBRFL is nearly identical to that of the scheme [5] for both datasets. We can conclude that PPBRFL achieves privacy preservation without compromising the model classification accuracy since PPBRFL employs symmetric homomorphic encryption to achieve privacy protection without introducing any noise, thus maintaining the classification accuracy of the model. To comprehensively assess the influence of Byzantine robustness on the proposed PPBRFL, we undertake a comparative analysis between PPBRFL and the scheme [5]. We evaluate the classification accuracy of the same model under different datasets, and choose different numbers of Byzantine users with 0%, 20%

and 40%, respectively. Fig.3(a) and Fig.3(b) illustrate a comparison of the model classification accuracy achieved by the scheme [5] under two distinct scenarios: one in which the scheme [5] does not suffer Byzantine attack, and the other in which it is subjected to Byzantine attack (20% Byzantine users and 40% Byzantine users). The results show that the model classification accuracy of the scheme [5] decreases when attacked by Byzantine users, and decreases more significantly as the proportion of Byzantine users increases. This observed trend can be attributed to the inherent absence of Byzantine robustness within the scheme [5], rendering it considerably vulnerable to Byzantine user-induced disturbances.

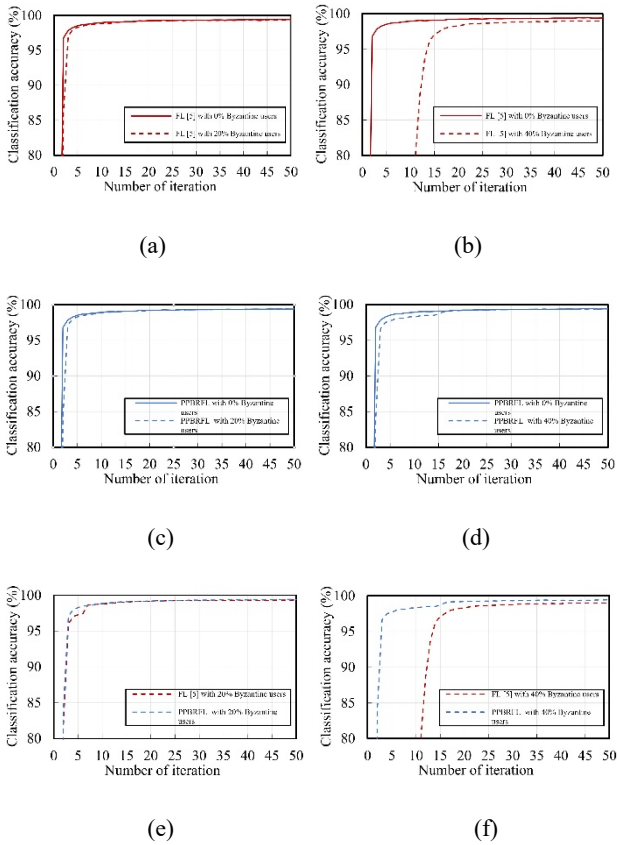


Figure 4. Comparison of classification accuracy on the MNIST dataset between PPBRFL and FL [5] under Byzantine attack. (a) FL [5] with 0% Byzantine users and FL [5] with 20% Byzantine users. (b) FL [5] with 0% Byzantine users and FL [5] with 40% Byzantine users. (c) PPBRFL with 0% Byzantine users and PPBRFL with 20% Byzantine users. (d) PPBRFL with 0% Byzantine users and PPBRFL with 40% Byzantine users. (e) PPBRFL with 20% Byzantine users and FL [5] with 20% Byzantine users. (f) PPBRFL with 40% Byzantine users and FL [5] with 40% Byzantine users

Fig.3(c) and Fig.3(d) show a comparison of the model classification accuracy achieved by PPBRFL under two different scenarios: one in which PPBRFL is free of Byzantine attack, and the other in which it is subjected to Byzantine attack (20% Byzantine users and 40% Byzantine users). We can observe that the classification accuracy of the PPBRFL attacked by Byzantine users is almost as accurate as the classification accuracy of the PPBRFL not attacked by Byzantine users. As illustrated in Fig.3(e) and Fig.3(f), the model classification accuracy of PPBRFL demonstrates greater accuracy in comparison to the scheme [5] when attacked by the same number of Byzantine users. The above analyses indicate that PPBRFL exhibits remarkable Byzantine robustness and minimally impacts the model classification accuracy when attacked by Byzantine users. Fig.4 shows that PPBRFL maintains exceptional Byzantine robustness under MNIST dataset.

4.2. Computation Efficiency on the User Side

We pick the privacy-preserving federated learning scheme [7] as the benchmark, as it also uses homomorphic encryption to protect the user's data privacy during federated learning process. In this experiment, we evaluate the computation efficiency on the user side by progressively increasing the number of users for models with 5k entry parameters per user and 10k entry parameters per user. From Section 3, we can know that the computation overhead on the user side primarily arises from encrypting the model. As depicted in Fig.5(a) and Fig.5(b) show that as the number of users increases, the computational overheads on the user side of PPBRFL and the scheme [7] also grow accordingly. Furthermore, the proposed PPBRFL demonstrates significantly higher efficiency on the user side compared to the scheme [7].

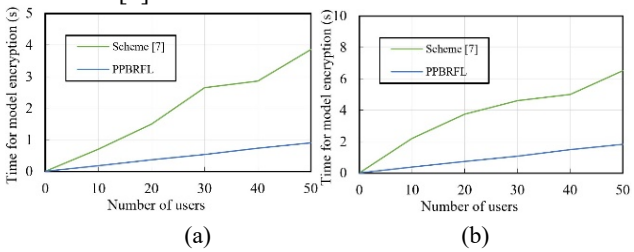


Figure 5. Comparison of model encryption computation overhead on the user side between PPBRFL and the scheme [7]. (a) 5K-entry parameters per user (b) 10K-entry parameters per user

5. Conclusion

In this paper, we propose a new privacy-preserving Byzantine robust federated learning scheme (PPBRFL). This scheme is able to support both privacy preserving and Byzantine robustness. Specifically, we utilize symmetric homomorphic encryption technology to encrypt the trained model. The cloud server can aggregate the encrypted trained models but cannot obtain users' data privacy from the encrypted trained models. Moreover, we use the cosine similarity value as a benchmark and utilize a grouping mechanism to select users for aggregation. Using the above method, our scheme can effectively resist the attacks from Byzantine users. Experiments illustrate that PPBRFL is able to achieve almost the same model classification accuracy as traditional federated learning scheme while guaranteeing user privacy and Byzantine robustness. The performance on the user side exhibits favorable efficiency.

References

[1] J. Roman and A. Jameel, "Backpropagation and recurrent neural networks in financial analysis of multiple stock market returns," in Proceedings of HICSS-29: 29th Hawaii international conference on system sciences, vol. 2. IEEE, 1996, pp. 454–460.

[2] M. Bakator and D. Radosav, "Deep learning and medical diagnosis: A review of literature," Multimodal Technologies and Interaction, vol. 2, no. 3, p. 47, 2018.

[3] A. B. Nassif, I. Shahin, I. Attili, M. Azzeh, and K. Shaalan, "Speech recognition using deep neural networks: A systematic review," IEEE access, vol. 7, pp. 19 143–19 165, 2019.

[4] G. Xu, H. Li, S. Liu, K. Yang, and X. Lin, "Verifynet: Secure and verifiable federated learning," IEEE Transactions on Information Forensics and Security, vol. 15, pp. 911–926, 2019.

[5] J. Konecnˇ y, H. B. McMahan, F. X. Yu, P. Richt ` arik, A. T. ´ Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," arXiv preprint arXiv:1610.05492, 2016.

[6] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konecnˇ y, S. Mazzocchi, ` B. McMahan et al., "Towards federated learning at scale: 9 System design," Proceedings of machine learning and systems, vol. 1, pp. 374–388, 2019.

[7] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," IEEE Transactions on Information Forensics and Security, vol. 13, no. 5, pp. 1333– 1345, 2018.

[8] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in Proceedings of the 22nd ACM SIGSAC conference on computer and communications security, 2015, pp. 1310–1321.

[9] F. Mo, H. Haddadi, K. Katevas, E. Marin, D. Perino, and N. Kourtellis, "Ppfl: privacy-preserving federated learning with trusted execution environments," in Proceedings of the 19th annual international conference on mobile systems, applications, and services, 2021, pp. 94–108.

[10] H. Fang and Q. Qian, "Privacy preserving machine learning with homomorphic encryption and federated learning," Future Internet, vol. 13, no. 4, p. 94, 2021.

[11] J. Ma, S.-A. Naas, S. Sigg, and X. Lyu, "Privacy-preserving federated learning based on multi-key homomorphic encryption," International Journal of Intelligent Systems, vol. 37, no. 9, pp. 5880–5901, 2022.

[12] X. Zhang, A. Fu, H. Wang, C. Zhou, and Z. Chen, "A privacy-preserving and verifiable federated learning scheme," in ICC 2020-2020 IEEE International Conference on Communications (ICC). IEEE, 2020, pp. 1–6.

[13] W. Wang, X. Li, X. Qiu, X. Zhang, J. Zhao, and V. Brusic, "A privacy preserving framework for federated learning in smart healthcare systems," Information Processing & Management, vol. 60, no. 1, p. 103167, 2023.

[14] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou, "A hybrid approach to privacy-preserving federated learning," in Proceedings of the 12th ACM workshop on artificial intelligence and security, 2019, pp. 1–11.

[15] J. Xu, S.-L. Huang, L. Song, and T. Lan, "Signguard: Byzantine-robust federated learning through collaborative malicious gradient filtering," arXiv preprint arXiv:2109.05872, 2021.

[16] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," Advances in neural information processing systems, vol. 30, 2017.

[17] 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.

[18] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "Fltrust: Byzantine-robust federated learning via trust bootstrapping," arXiv preprint arXiv:2012.13995, 2020.

[19] A. Fu, X. Zhang, N. Xiong, Y. Gao, H. Wang, and J. Zhang, "Vfl: A verifiable federated learning with privacy-preserving for big data in industrial iot," IEEE Transactions on Industrial Informatics, vol. 18, no. 5, pp. 3316–3326, 2020.

[20] H. Gao, N. He, and T. Gao, "Sverifl: Successive verifiable federated learning with privacy-preserving," Information Sciences, vol. 622, pp. 98–114, 2023.

[21] R. Xu, N. Baracaldo, Y. Zhou, A. Anwar, and H. Ludwig, "Hybridalpha: An efficient approach for privacypreserving federated learning," in Proceedings of the 12th ACM workshop on artificial intelligence and security, 2019, pp. 13–23.

[22] R. Hu, Y. Guo, H. Li, Q. Pei, and Y. Gong, "Personalized federated learning with differential privacy," IEEE Internet of Things Journal, vol. 7, no. 10, pp. 9530–9539, 2020.

[23] 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.

[24] H. Ye, J. Liu, H. Zhen, W. Jiang, B. Wang, and W. Wang, "Vrefl: Verifiable and reconnection-efficient federated learning in iot scenarios," Journal of Network and Computer Applications, vol. 207, p. 103486, 2022.

[25] 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.

[26] C. Fang, Y. Guo, N. Wang, and A. Ju, "Highly efficient federated learning with strong privacy preservation in cloud computing," Computers & Security, vol. 96, p. 101889, 2020.

[27] B. Jia, X. Zhang, J. Liu, Y. Zhang, K. Huang, and Y. Liang, "Blockchain-enabled federated learning data protection aggregation scheme with differential privacy and homomorphic encryption in iiot," IEEE Transactions on Industrial Informatics, vol. 18, no. 6, pp. 4049–4058, 2021.

[28] H. Zhou, G. Yang, H. Dai, and G. Liu, "Pflf: Privacy-preserving federated learning framework for edge computing," IEEE Transactions on Information Forensics and Security, vol. 17, pp. 1905–1918, 2022.

[29] V. Mugunthan, A. Polychroniadou, D. Byrd, and T. H. Balch, "Smpai: Secure multi-party computation for federated learning," in Proceedings of the NeurIPS 2019 Workshop on Robust AI in Financial Services. MIT Press Cambridge, MA, USA, 2019, pp. 1–9.

[30] C. Zhou, A. Fu, S. Yu, W. Yang, H. Wang, and Y. Zhang, "Privacy-preserving federated learning in fog computing," IEEE Internet of Things Journal, vol. 7, no. 11, pp. 10 782–10 793, 2020.

[31] 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, 2021.N. Kawasaki, "Parametric study of thermal and chemical nonequilibrium nozzle flow," M.S. thesis, Dept. Electron. Eng., Osaka Univ., Osaka, Japan, 1993.

[32] 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.

[33] B. Zhao, P. Sun, L. Fang, T. Wang, and K. Jiang, "Fedcom: A byzantine-robust local model aggregation rule 10 using data commitment for federated learning," arXiv preprint arXiv:2104.08020, 2021.

[34] H. Guo, H. Wang, T. Song, Y. Hua, Z. Lv, X. Jin, Z. Xue, R. Ma, and H. Guan, "Siren: Byzantine-robust federated learning via proactive alarming," in Proceedings of the ACM Symposium on Cloud Computing, 2021, pp. 47–60.

[35] L. Munoz-Gonz ˜ alez, K. T. Co, and E. C. Lupu, ´ "Byzantine-robust federated machine learning through adaptive model averaging," arXiv preprint arXiv:1909.05125, 2019.E. E. Reber, R. L. Michell, and C. J. Carter, "Oxygen absorption in the Earth's atmosphere," Aerospace Corp., Los Angeles, CA, Tech. Rep. TR-0200 (420-46)-3, Nov. 1988.

[36] R. Wang, X. Wang, H. Chen, S. Picek, Z. Liu, and K. Liang, "Brief but powerful: Byzantine-robust and privacy-preserving federated learning via model segmentation and secure clustering," arXiv preprint arXiv:2208.10161, 2022.

[37] S. Li, E. Ngai, and T. Voigt, "Byzantine-robust aggregation in federated learning empowered industrial iot," IEEE Transactions on Industrial Informatics, vol. 19, no. 2, pp. 1165–1175, 2021.

[38] X. Tang, M. Shen, Q. Li, L. Zhu, T. Xue, and Q. Qu, "Pile: Robust privacy-preserving federated learning via verifiable perturbations," IEEE Transactions on Dependable and Secure Computing, 2023.

[39] L. Deng, "The mnist database of handwritten digit images for machine learning research [best of the web]," IEEE signal processing magazine, vol. 29, no. 6, pp. 141–142, 2012.

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

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