Achieving Provable Byzantine Fault-tolerance in a Semi-honest Federated Learning Setting

Xingxing Tang[a], **Hanlin Gu**[b], **Lixin Fan**[b], **Qiang Yang**[a,b]
[a]Department of Computer Science and Engineering, HKUST {xtangav, qyang}@cse.ust.hk
[b]WeBank AI Lab, WeBank {allengu, lixinfan}@webank.com

Abstract.
Federated learning (FL) is a suite of technology that allows multiple distributed participants to collaboratively build a global machine learning model without disclosing private datasets to each other. We consider an FL setting in which there may exist both a) semi-honest participants who aim to eavesdrop on other participants' private datasets; and b) Byzantine participants who aim to degrade the performances of the global model by submitting detrimental model updates. The proposed framework leverages the Expectation-Maximization algorithm first in E-step to estimate unknown participant membership, respectively, of Byzantine and benign participants, and in M-step to optimize the global model performance by excluding malicious model updates uploaded by Byzantine participants. One novel feature of the proposed method, which facilitates reliable detection of Byzantine participants even with HE or MPC protections, is to estimate participant membership based on the performances of a set of randomly generated candidate models evaluated by all participants. The extensive experiments and theoretical analysis demonstrate that our framework guarantees Byzantine Fault-tolerance in various federated learning settings with private-preserving mechanisms.

Keywords: Federated Learning, Byzantine Fault-tolerance, Semi-honest party.

1 Introduction

With the increasing popularity of federated learning (FL) in a variety of application scenarios,[29] it is of paramount importance to be vigilant against various Byzantine attacks, which aim to degrade FL model performances by submitting malicious model updates.[4,11,27] Effective Byzantine Fault-tolerant methods to thwart such Byzantine attacks have been proposed in literature.[5,30,31]

Table 1: Classification of Byzantine Fault-tolerant methods. **-**: No Reference.

Protection Mechanism	Byzantine Fault-tolerant Methods					
	Updates-based				Performance-based	
	Robust Statistics	Clustering	Historical	Server-Based	Server Eval.	Client Eval.
No Protection	[5,30]	[23,24]	[2,31]	[8,22]	[26,28]	[16] FedPBF
DP	[19,32]	-	-	-	-	FedPBF
MPC	[14,25]	-	-	-	-	
HE	[20]	-	-	-	-	

Although numerous Byzantine Fault-tolerant methods have demonstrated effectiveness in defeating attacks under various federated learning settings, however, the majority of existing work is not readily applicable to a critical FL setting in which certain *privacy-preserving mechanisms* e.g., Differential Privacy (DP), Homomorphic Encryption (HE) or Secure Multiparty Computation (MPC) [1] are adopted to protect model updates from disclosing private training data or models. We

[1]DP adds random noise to data to protect individual privacy while allowing useful data insights.[1] HE is a cryptographic technique that allows computation on encrypted data without the need for decryption, preserving privacy and security.[3,18] MPC is a protocol or technique that enables multiple parties to jointly perform a specific computation task without revealing their private data.[6]

regard this deficiency as a detrimental shortcoming that renders many Byzantine Fault-tolerant methods useless in practice since federated learning without privacy-preserving mechanisms poses serious privacy leakage risks that defeat the purpose of federated learning in the first place. For instance, it was shown that attackers could exploit unprotected deep neural network model updates to reconstruct training images with pixel-level accuracy.[33]

To achieve privacy-preserving and Byzantine Fault-tolerance simultaneously in FL, we propose a ***P**rovable **B**yzantine **F**ault-tolerant framework in a semi-honest **Fed**erated learning setting*, called **FedPBF**, which leverages EM Algorithm[9,10] in E-step to estimate unknown participant membership, respectively, of Byzantine and benign participants, and in M-step to optimize the global model performance by excluding malicious model updates uploaded by Byzantine participants. As compared with *model updates* based methods, the proposed FedPBF is based on model performances of a set of randomly generated candidate models evaluated by all participants via their local dataset, which can be applied to various privacy-preserving mechanisms, e.g., DP, HE and MPC (shown in Tab. 2). Moreover, the FedPBF uses robust estimation (median) to defence *misreporting* by Byzantine participants (Sect. 2). Our extensive experiments (Sect. 4) and theoretical analysis (Appendix[2] A) demonstrate that the FedPBF method shows superior model performances in the presence of a large variety of Byzantine attacks and privacy-preserving mechanisms. Tab. 1 illustrated that the FedPBF can be applied to all FL scenarios with different privacy-preserving mechanisms (e.g., DP, MPC and HE) adopted.

2 Related Work

We classify existing federated learning Byzantine Fault-tolerant methods into two main categories in Tab. 1: **update-based** and **performance-based** methods.

Updates-based Byzantine Fault-tolerant methods are based on model updates uploaded by the client to detect Byzantine participants. Some methods[5,30] regarded malicious updates as outliers and leveraged **Robust statistics** to filter out malicious updates. Another line of work adopted **clustering** methods[23,24] to distinguish benign and Byzantine participants. Moreover, some **server-based** Byzantine Fault-tolerant methods assume that the server has an additional dataset to evaluate updates uploaded by clients.[8,22] In addition, some methods made use of **historical information** to help correct the statistical bias brought by Byzantine participants during the training, and thus lead to the convergence of optimization of federated learning.[2,31] However, all of the updated-based methods don't consider privacy-preserving mechanisms, such as DP and HE.

Performance-based Byzantine Fault-tolerant methods detect Byzantine participants based on model performance evaluation. Some methods[26,28] assumed the availability of reliable public datasets that can be used by the **server to evaluate** model performances. However, the availability of such server-side public datasets is hardly fulfilled in practice since those server-side datasets are either limited in terms of their data size or their distributions are different from private datasets owned by clients.

Moreover, other methods[16] used private datasets on the **client to evaluate** the performance of local updates; however, they didn't consider the existence of the *misreporting* by Byzantine participants.

Byzantine Fault-tolerant & privacy-preserving methods considered Byzantine Fault-tolerant and privacy-preserving at the same time. They combined privacy-preserving mechanisms such

[2]Online appendix: https://github.com/TangXing/PAKDD2023-FedPBF

as DP,[19,32] MPC[14,25] and HE[20] and Byzantine Fault-tolerant methods to address privacy issues and Byzantine attacks simultaneously. However, the methods proposed in[19,32] can only be used to protect the sign of updates sent to the server using DP. Moreover, Ma et al. applied HE to Byzantine problems.[20] Nevertheless, it only allowed for the encryption of the set $\{-1, 0, 1\}$ using HE, which may not be sufficient for the general case of encrypting model updates. Additionally, the approaches presented in[14,25] are designed to be used with MPC protocols for update-based arithmetic operations, which might restrict their applicability to other privacy-preserving mechanisms.

3 The Proposed Method

This section first formally defines the setting and threat model in which a *semi-honest* federated learning setting and an unknown number of *Byzantine* participants (aka, Byzantine clients: this description is used in the following sections.) are assumed. Sect. 3.2 then delineates the proposed framework, which demonstrates *provable Byzantine Fault-tolerance* in the presence of various privacy-preserving mechanisms (e.g., DP, HE and MPC) adopted by clients to prevent the semi-honest server to eavesdrop private data from clients.

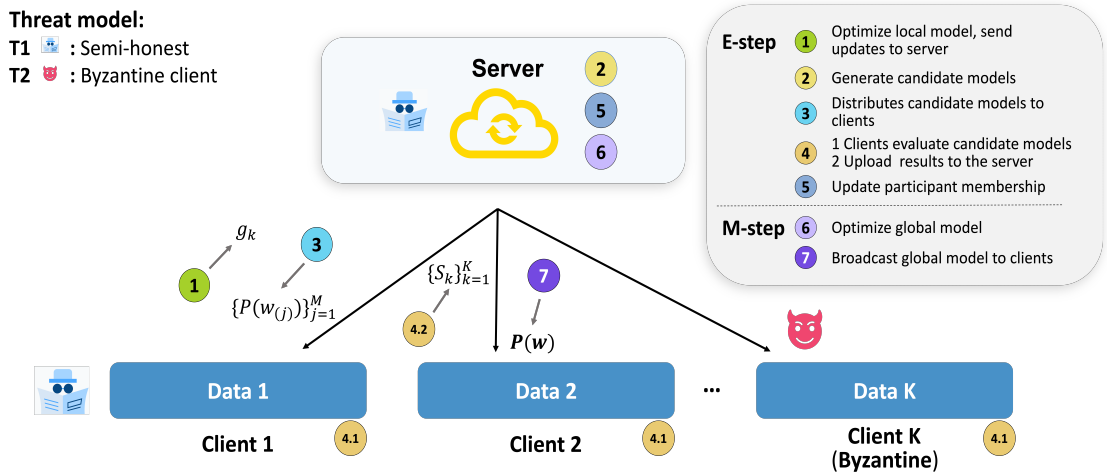


Fig 1: Overview of threat models and the proposed FedPBF: **E-step** and **M-step**. The details of **E-step** and **M-step** are described in Sect. 3.2.

3.1 The Setting and Threat Model

We consider a *horizontal federated learning* setting (as defined in[29]), in which K clients collaboratively train a global model with weights $\mathbf{w}$ using private datasets residing on each client i.e. $\mathcal{D}_k = \{\mathbf{x}_{k,n}, y_{k,n}\}_{n=1}^{N_k}, k = 1 \cdots K$ and N_k the total number of data points on respective clients:

$$\min_{\mathbf{w}} \sum_{k=1}^{K} \sum_{n=1}^{N_k} F_k(\mathbf{w}, \mathbf{x}_{k,n}, y_{k,n}) = \min_{\mathbf{w}} \sum_{k=1}^{K} F_k(\mathbf{w}, \mathcal{D}_k), \tag{1}$$

where F_k is the loss function w.r.t. weights $\mathbf{w}$ of k_{th} client.

The minimization of Eq. (1) essentially uplifts the global model performance, such that clients are motivated to join the federated learning mission for the benefits of improved model performances. However, one must deal with the following two types of threats that may defeat the purpose of federated learning in the first place (see Fig. 1).

Threat type I – Semi-honest parties: we assume a *semi-honest* threat model in which either clients or the server may eavesdrop on private data owned by other clients. Therefore, due to the privacy-preserving concern, private data $\mathcal{D}_k$ are never communicated with peer clients or the server. Instead, it is the protected model updates $P\big(\nabla F_k\big)$ in Eq. (2) that are sent from each client to the server. $P(\cdot)$ denotes certain privacy-preserving mechanisms, e.g., Differential Privacy or Homomorphic Encryption adopted by clients, to prevent a semi-honest party from inferring information about private data $\mathcal{D}_k$ based on the unprotected local model update ∇F_k[3].

However, as demonstrated throughout this paper, the adoption of such privacy-preserving mechanisms poses severe challenges to the defence of another type of threat, i.e., *Byzantine clients* whose behaviour may significantly degrade global model performances.

Threat type II – Byzantine clients: we assume that out of K clients, there exist up to f *Byzantine clients*, whose local model updates $\mathbf{g}_k$ may deviate from those of benign clients in an arbitrary manner:

$$\mathbf{g}_k = \begin{cases} P(\nabla F_k(\mathbf{w})) & \text{Benign clients} \\ P(\mathbf{g}_b) & \text{Byzantine clients.} \end{cases} \tag{2}$$

Note that misbehaves of Byzantine clients may be ascribed to different reasons, e.g., network malfunctioning or malicious clients who intentionally aim to degrade the global model performances.[4,11,27] Due to various root causes of Byzantine clients, their identities are often unknown. Moreover, in the case of malicious clients, they may collude and upload detrimental yet disguised model updates that evade many existing Byzantine Fault-tolerant methods (see, e.g.,[11,12]). To make things worse, the adoption of certain privacy-preserving mechanisms, e.g., HE or MPC, renders many Byzantine Fault-tolerant methods completely useless, as illustrated in Sect. 4.2.

Our mandate, therefore, is to study a general Byzantine Fault-tolerant federated aggregation scheme that admits *exact Fault-tolerance* as defined below in the presence of privacy-preserving mechanisms[4].

theoremExact Fault-tolerance] Given a set of K model updates $\mathbf{g}_k, k = 1 \cdots K$ with a subset $\mathcal{G}$ consisting of m benign clients, a federated aggregation scheme is said to have *exact Fault-tolerance* if it allows all the benign clients to compute

$$\mathbf{w}^*_{\mathcal{G}} \in \arg\min_{\mathbf{w}} \sum_{k \in \mathcal{G}} F_k(\mathbf{w}, \mathcal{D}_k). \tag{3}$$

Since the subset $\mathcal{G}$ is a prior unknown, one must estimate its unknown participant membership during the optimization of Eq. (1). Therefore, we regard the detection accuracy of the estimation, as defined below, as a crucial measure of Byzantine Fault-tolerance. We provide the theoretical guarantee in Appendix A that the detection accuracy of our proposed FedPBF converges to 100%.

theoremDetection Accuracy]At t_{th} iteration, the Detection Accuracy η_t is defined as the fraction of benign clients over selected clients to aggregate by the server.

$$\eta_t = \frac{\#(\mathcal{I}^t \cap \mathcal{G})}{K}, \tag{4}$$

[3]Such privacy leakage risks have been demonstrated for particular cases (e.g., see[33]) where attackers can exploit unprotected deep neural network model updates to reconstruct training images with pixel-level accuracy.

[4]Notions of *exact Fault-tolerance* is previously introduced in,[13] in which a comparative elimination (CE) filtered-based scheme was proposed to achieve Byzantine Fault-tolerance under different conditions. We adopt these definitions to prove that the framework proposed in this article does admit these Fault-tolerances in the presence of privacy-preserving mechanisms.

where $\mathcal{I}^t$ is the set of clients chosen by the server to aggregate at t_{th} iteration, and K is the number of clients. Moreover, we define the **averaged detection accuracy** η among T iterations as:

$$\eta = \frac{1}{T}\sum_{t=1}^{T}\eta_t \tag{5}$$

3.2 FedPBF

In this section, we illustrate a generic Byzantine Fault-tolerant framework in which an EM algorithm[9,10] (see Fig. 1) is adopted to solve the following problem, where the unknown participant membership $\mathbf{r} \in \{0,1\}^K$ is updated in the E-step (①-⑤), and the global model parameter $\mathbf{w}$ is optimized in the M-step (⑥-⑦) as (see Algo. 1):

$$\arg\min_{\mathbf{w},\mathbf{r}} \sum_{k=1}^{K} F_k(\mathbf{w}, \mathbf{r}, \mathcal{D}_k) \tag{6}$$

- ①: Each client optimizes their local model via $\min F_k(\mathbf{w}, \mathcal{D}_k)$ and sends the protected model updates $\mathbf{g}_k$ to the server.
- ②: The server first randomly selects M groups Q clients indexed by $\mathcal{I}_j = \{c_{1j}, \ldots, c_{Qj}\}, j \in \{1, \ldots, M\}, c_{qj} \in \{1, \ldots, K\}, q \in \{1, \ldots, Q\}$, according to sample probability $\mathbf{p}^t$, which is proportional to the cumulative participant membership summed up from iteration 0 to $t-1$ as follows:

  $$\mathbf{p}^t = \sum_{i=0}^{t-1}\mathbf{r}^i / \| \sum_{i=0}^{t-1}\mathbf{r}^i \|_1. \tag{7}$$

  It is noted that we use cumulative participant membership in order to take advantage of the historical evaluation for all clients, i.e., $\{\mathbf{r}^i\}_{i=0}^{t-1}$.

  Then the server generates M protected candidate models as

  $$P(\mathbf{w}_{(j)}^t) = P(\mathbf{w}^{t-1}) - \frac{1}{Q}\sum_{k\in\mathcal{I}_j}\mathbf{g}_k, j \in \{1, \cdots, M\}. \tag{8}$$

- ③: The server distributes M protected candidate models $\{P(\mathbf{w}_{(j)}^t)\}_{j=1}^M$ to all clients.
- ④: Each client executes the following two sub-steps: ④.① each client decrypts M candidate models to obtain $\mathbf{w}_{(j)}^t$[5]. They calculate the empirical accuracy $\{S_k \in \mathbb{R}^M\}_{k=1}^K$ of M candidate models via their own dataset as:

  $$S_k(j) = Acc(\mathbf{w}_{(j)}^t, \mathcal{D}_k), j \in \{1, \cdots, M\} \tag{9}$$

[5]For some protection mechanisms such as DP, this process may cause the loss of model precision.

where Acc() represents the accuracy of the candidate model $\mathbf{w}_{(j)}^t$ measured w.r.t. $\mathcal{D}_k$ and $S_k(j)$ represents the j_{th} candidate model measured by the k_{th} client. ④② then each clients upload $S_k, k \in \{1, \ldots, K\}$ to the server.

Note that Byzantine clients may misreport the model accuracy. Our client-side performance-based evaluation method involves each client evaluating all candidate models. The server can use robust filters[5,30] to mitigate the impact of misreporting as long as the ratio of Byzantine clients is below 0.5.

- ⑤: Upon receiving $\{S_k\}_{k=1}^K$, the server picks up the best candidate model:

$$j^* = \arg \max_{j \in \{1,\ldots,M\}} \text{Median}_{k \in \{1,\ldots,K\}} S_k(j) \tag{10}$$

Note that we use the robust filter, i.e. Median, to filter out the Byzantine clients. Other robust filters such as[5,30] could also be applied to our methods. Then the server updates participant membership $\mathbf{r}^t$ as:

$$\mathbf{r}^t(k) = \begin{cases} 1, & \text{if} \quad k \in \mathcal{I}_{j^*}, \text{ where } j^* \text{ is the best candidate model by Eq. 10 ;} \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

- ⑥: The server optimizes the protected global model $P(\mathbf{w}^t)$ at t_{th} iteration according to participant membership $\mathbf{r}^t$ as:

$$P(\mathbf{w}^t) = P(\mathbf{w}^{t-1}) - \frac{1}{Q} \sum_{k=1}^{K} \mathbf{r}^t(k) P(\mathbf{g}_k^t) \tag{12}$$

- ⑦: The server broadcasts the protected global model $P(\mathbf{w}^t)$ to all clients, which will optimize respective local models in the next iteration.

There are three fundamental reasons for the proposed FedPBF to satisfy privacy-preserving and Byzantine Fault-tolerant requirements.

1. FedPBF allows *reliable* detection of Byzantine clients in the presence of privacy-preserving mechanisms (e.g., DP, HE and MPC). This feature is achieved by leveraging client-side datasets to evaluate candidate model performances. Algo. 1 is thus applicable to all protected model updates, regardless of whatever privacy-preserving mechanisms are adopted.

2. FedPBF allows *efficient* estimation of participant membership $\mathbf{r}$, by sampling multiple candidates group models. This sampling approach is scalable in case the number of clients is large.

3. FedPBF is *robust* to misreported performances reported by Byzantine attackers. This merit is ensured by the robust estimation of the Median filter adopted in Eq. (10) when the ratio of Byzantine clients is less than the breakdown point, i.e. 0.5.

Algorithm 1 FedPBF

Input: K: the number of clients; $\mathcal{D}_k$: local training datasets of k_{th} client; T: number of global iterations; M: the number of candidate models; Q: the number of aggregated updates for each candidate model; η: the local learning rate; b: the batch size;

Output: Global model $\mathbf{w}$.

Initialization: $\mathbf{w}^0 \leftarrow$ random value, participant membership $\mathbf{r}^0 \leftarrow [1, \cdots, 1]$.
for $t = 1, 2, \cdots, T$ **do**
    E Step:
    for each client k, $k \in [K]$ **do in parallel**
        Compute the local updates as $\mathbf{g}_k^{t-1} = ModelUpdate(\mathbf{w}_k^t, \mathcal{D}_k, b, \eta)$.
        Each client sends $\mathbf{g}_k^t$ to the server.
    end for
    $\mathbf{p}^t = \sum_{i=0}^{t-1} \mathbf{r}^i / \| \sum_{i=0}^{t-1} \mathbf{r}^i \|_1$
    The server generates the M candidate models $\{P(\mathbf{w}_{(j)}^t)\}_{j=1}^M$ according to $\mathbf{p}^t$ by Eq. (8);
    The server distributes M candidate models $\{P(\mathbf{w}_{(j)}^t)\}_{j=1}^M$ to all clients.
    Each clients decrypts and evaluates M candidate models to obtain the accuracies $\{S_k\}_{k=1}^K$ according to Eq. (9) and send them to the server;
    The server chooses the candidate model with the largest accuracy as Eq. (10). Then the server updates participant membership $\mathbf{r}^t$ by Eq. (11).
    M step:
    The server updates the global model via Eq. (12);
    The server broadcast the protected global model $P(\mathbf{w}^t)$ to all clients.
end for
return $\mathbf{w}$.

4 Experiments

In this section, we conduct extensive experiments to answer the following three questions: **Question 1:** To what extent does the proposed FedPBF outperform other Byzantine Fault-tolerant methods in federated learning with different privacy-preserving mechanisms (e.g., DP, HE, MPC)? **Question 2:** To what extent the model performance (accuracy) of the proposed FedPBF is affected by the varying Byzantine client proportions, the extent of Non-IID or misreporting conditions by Byzantine clients? **Question 3:** How do hyperparameters, i.e., the number of candidate models M and aggregated updates for each candidate model Q and Byzantine clients ratio, affect the convergence and effectiveness of the proposed FedPBF algorithm in practice (due to the page limit, these results are shown in Appendix B.2)?

4.1 Setup and evaluation metrics

Datasets: *MNIST*,*Fashion-MNIST*(FMNIST) and *CIFAR10*are used for image classification tasks. The extent of Non-IID of the dataset is obtained by changing the parameter β from 0.5 to 1 of the Dirichlet distribution $Dir(\beta)$.[17]

Models: *Logistic regression*, *LeNet* and *AlexNet* models are used to train MNIST, FMNIST and CIFAR10 respectively.

Federated Learning Settings: We simulate a horizontal federated learning system with K = 100 clients (MNIST and FMNIST) or K=20 clients (CIFAR10) in a stand-alone machine. The detail of hyper-parameters for training are illustrated in Appendix B.
Federated Learning Privacy-preserving Mechanisms: Privacy-preserving mechanisms: Differential Privacy[1](DP) with the variance of Gaussian noise range between $\sigma^2 = 10^{-4}$ and $\sigma^2 = 10^{-1}$ as in,[33] Secure Multiparty Computation (MPC),[7] Homomorphic Encryption (HE)[3] used to protect the privacy of local data.
Byzantine Fault-tolerant methods: Five existing methods: Krum,[5] Median,[30] Trimmed Mean,[30] Kmeans,[24] FLtrust,[8] and the proposed method FedPBF are compared in terms of following metrics.
Evaluation metric: *Model Performance (MP)*, *Averaged Model Performance (AMP)* and *Averaged Detection Accuracy* (Def.) of the federated model is used to evaluate model accuracy defending capabilities of different methods.
Byzantine attacks: We set 10%, 20%, and 35% clients are Byzantine attackers. The following attacking methods are used in experiments: 1) the *same value attack*, where model updates of attackers are replaced by the all ones vector. 2) the *label flipping attack*, where attackers use the wrong label to generate the gradients to upload. 3) the *sign flipping attack*, where local gradients of attackers are shifted by a scaled value -4. 4) the *gaussian attack*, where local gradients at clients are replaced by independent Gaussian random vectors $\mathcal{N}(0, 200)$. 5) the *Lie attack*, which was designed in.[4] 6) the *Fang-v1 attack*.[11] 7) the *Fang-v2 attack*.[11] 8) the *Mimic attack*.[15]

4.2 Comparison with existing Byzantine Fault-tolerant Methods

To answer **Question 1**, we evaluate the Averaged Model Performance (AMP) of five existing Byzantine Fault-tolerant methods and our proposed method under eight attacks. There are three notable observations according to Tab. 2:

1. When no privacy-preserving mechanisms (No Protection) are applied, the *AMP* of the FedPBF outperforms other methods from 5% to 12% on the MNIST dataset and from 11% and 49% on the FMNIST dataset.

2. When different magnitudes of Gaussian noise are added (DP: the variance of Gaussian noise range between $\sigma^2 = 10^{-4}$ and $\sigma^2 = 10^{-1}$), the FedPBF significantly outperforms other methods. Especially when the noise increases, the *AMP* of other methods is broken (e.g., *AMP* of FLtrust degrades from 76.6% to 22.3% on FMNIST). However, the *AMP* of our FedPBF doesn't drop seriously under various degrees of noise (e.g., the *AMP* only drops from 91.7% to 85.8% on MNIST).

3. When the HE and MPC are applied, the FedPBF still performs well, i.e., there is a minor loss of the *AMP* compared with the baseline (FedAvg without attack).

4.3 Robustness

In this subsection, we test the robustness of the FedPBF under different Non-IID extents of the local dataset, different Byzantine client percentages and Byzantine clients misreporting types (due to the page limit, these results are shown in Appendix B.1) on CIFAR10 with AlexNet to answer **Question 2**.

Table 2: Averaged model performance (accuracy percentage: %) of different Byzantine Fault-tolerant methods under different privacy-preserving mechanisms (with Non-IID setting $\beta = 0.5$, $Q = 10$, $M = 40$ and 20% Byzantine clients for classification of MNIST and FMNIST). FedAvg W.O. Attack: FedAvg[21] without attack. DP: Differential privacy with Gaussian noise and σ^2 is the variance of Gaussian noise. -: as far as we know, there is no solution existing.

			Krum [5]	Median [30]	Trimmed [30]	Kmeans [24]	FLtrust [8]	**FedPBF (Ours)**	FedAvg[21] W.O. Attack
MNIST	No Protection		78.8±15.4	81.2±15.4	80.3±18.0	82.4±17.5	85.9±5.6	**91.7±0.5**	92.5±0.1
	DP	$\sigma^2 = 10^{-4}$	76.8±25.7	82.5±21.0	83.4±18.4	75.4±29.9	88.1±7.9	**91.5±0.5**	92.4±0.1
		$\sigma^2 = 10^{-3}$	72.1±24.0	79.5±23.7	76.6±26.4	74.5±31.0	86.9±6.9	**90.2±0.1**	92.2±0.1
		$\sigma^2 = 10^{-2}$	63.1±6.5	69.0±33.0	69.1±34.0	73.0±29.1	82.0±9.2	**87.8±0.6**	90.8±0.2
		$\sigma^2 = 10^{-1}$	39.4±9.1	64.3±31.4	64.6±31.5	63.6±28.1	70.9±19.7	**85.8±0.8**	87.2±0.3
	HE		-	-	-	-	-	**91.6±0.4**	92.5±0.1
	MPC		-	-	-	-	-	**91.7±0.3**	92.5±0.1
FMNIST	No Protection		69.9±24.2	39.3±23.5	55.7±31.5	63.3±34.7	76.6±27.2	**88.5±0.6**	90.3±0.6
	DP	$\sigma^2 = 10^{-4}$	65.6±22.9	29.2±26.5	36.1±33.9	67.9±34.3	71.0±24.6	**87.5±1.3**	90.0±0.1
		$\sigma^2 = 10^{-3}$	60.2±20.6	15.8±11.2	33.1±33.8	66.4±34.3	58.2±35.2	**86.3±2.3**	88.2±0.4
		$\sigma^2 = 10^{-2}$	19.7±7.8	10.1±0.6	32.1±31.8	58.4±30.2	43.0±29.5	**81.2±5.8**	83.2±0.3
		$\sigma^2 = 10^{-1}$	10.5±1.1	9.5±1.2	22.5±21.8	31.7±19.9	22.3±16.8	**69.8±2.0**	73.4±0.5
	HE		-	-	-	-	-	**88.8±0.5**	90.2±0.1
	MPC		-	-	-	-	-	**89.1±0.4**	89.8±0.1

1. **Robustness under Different Byzantine Client Percentages:** Fig. 2 (left) illustrates the model performance of FedPBF under various attacks for different percentages of Byzantine clients, i.e., 10%, 20%, 35% with IID dataset. It shows that the degradation of model performance of the FedPBF is less than 2% compared with the baseline (FedAvg without attack: blue dotted lines) even the Byzantine client percentage increases to 35%.

2. **Robustness under Heterogeneous Dataset:** In Fig. 2 (right), it is shown that the degradation of model performance of the FedPBF, under various attacks with different clients' datasets Non-IID extents, is less than 1.5% all the time, which indicates the proposed FedPBF is robust under various Non-IID extents.

5 Conclusion

This paper proposed a novel Byzantine Fault-tolerant framework called FedPBF to guarantee Byzantine Fault-tolerance in the presence of protection mechanisms. To our best knowledge, this paper is the first research endeavour that thoroughly investigates the performances of various Byzantine tolerant methods with different protection mechanisms such as DP, HE and MPC being applied. Methodology-wise, we use the Expectation-Maximization algorithm to update the participant membership and optimize the global model performance alternately. The key for the FedPBF applying federated learning with various privacy-preserving mechanisms is that we use *model performance* in E-step to evaluate candidate models at the client side. This novel client-side performance-based evaluation, in tandem with the EM algorithm, constitutes our main contribution to the effective defence of Byzantine attacks in the presence semi-honest FL setting.

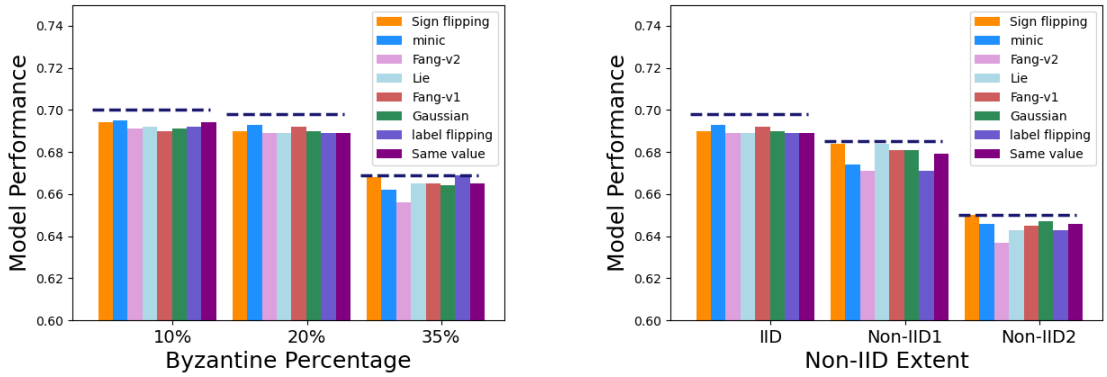


Fig 2: The model performance of the FedPBF with different Byzantine client percentages (10%, 20% and 35%) and Non-IID extents (IID, Non-IID1 with $\beta = 1$, and Non-IID2 with $\beta = 1$) on CIFAR10 under different attacks. Blue dotted lines represent the baseline meaning FedAvg without attack.

Acknowledgments. This work is partly supported by National Key Research and Development Program of China (2020YFB1805501).

References

1 Abadi, M., Chu, A., Goodfellow, I., McMahan, H.B., Mironov, I., Talwar, K., Zhang, L.: Deep learning with differential privacy. In: Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. pp. 308–318 (2016)

2 Allen-Zhu, Z., Ebrahimianghazani, F., Li, J., Alistarh, D.: Byzantine-resilient non-convex stochastic gradient descent. In: International Conference on Learning Representations (2020)

3 Aono, Y., Hayashi, T., Wang, L., Moriai, S., et al.: Privacy-preserving deep learning via additively homomorphic encryption. IEEE Transactions on Information Forensics and Security **13**(5), 1333–1345 (2017)

4 Baruch, G., Baruch, M., Goldberg, Y.: A little is enough: Circumventing defenses for distributed learning. Advances in Neural Information Processing Systems **32** (2019)

5 Blanchard, P., El Mhamdi, E.M., Guerraoui, R., Stainer, J.: Machine learning with adversaries: Byzantine tolerant gradient descent. Advances in Neural Information Processing Systems **30** (2017)

6 Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H.B., Patel, S., Ramage, D., Segal, A., Seth, K.: Practical secure aggregation for privacy-preserving machine learning. In: proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. pp. 1175–1191 (2017)

7 Boyle, E., Gilboa, N., Ishai, Y.: Function secret sharing. In: Annual international conference on the theory and applications of cryptographic techniques. pp. 337–367. Springer (2015)

8 Cao, X., Fang, M., Liu, J., Gong, N.Z.: Fltrust: Byzantine-robust federated learning via trust bootstrapping. arXiv preprint arXiv:2012.13995 (2020)

9 Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society: Series B (Methodological) **39**(1), 1–22 (1977)

10 Dieuleveut, A., Fort, G., Moulines, E., Robin, G.: Federated-em with heterogeneity mitigation and variance reduction. Advances in Neural Information Processing Systems **34**, 29553–29566 (2021)
11 Fang, M., Cao, X., Jia, J., Gong, N.: Local model poisoning attacks to {Byzantine-Robust} federated learning. In: 29th USENIX Security Symposium (USENIX Security 20). pp. 1605–1622 (2020)
12 Gu, H., Fan, L., Tang, X., Yang, Q.: Fedcut: A spectral analysis framework for reliable detection of byzantine colluders. arXiv preprint arXiv:2211.13389 (2022)
13 Gupta, N., Doan, T.T., Vaidya, N.: Byzantine fault-tolerance in federated local sgd under 2f-redundancy. arXiv preprint arXiv:2108.11769 (2021)
14 He, L., Karimireddy, S.P., Jaggi, M.: Secure byzantine-robust machine learning. arXiv preprint arXiv:2006.04747 (2020)
15 Karimireddy, S.P., He, L., Jaggi, M.: Byzantine-robust learning on heterogeneous datasets via bucketing. arXiv preprint arXiv:2006.09365 (2020)
16 Lai, F., Zhu, X., Madhyastha, H.V., Chowdhury, M.: Oort: Efficient federated learning via guided participant selection. In: 15th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 21). pp. 19–35 (2021)
17 Li, Q., Diao, Y., Chen, Q., He, B.: Federated learning on non-iid data silos: An experimental study. arXiv preprint arXiv:2102.02079 (2021)
18 Ma, J., Naas, S.A., Sigg, S., Lyu, X.: Privacy-preserving federated learning based on multi-key homomorphic encryption. International Journal of Intelligent Systems (2022)
19 Ma, X., Sun, X., Wu, Y., Liu, Z., Chen, X., Dong, C.: Differentially private byzantine-robust federated learning. IEEE Transactions on Parallel and Distributed Systems **33**(12), 3690–3701 (2022). https://doi.org/10.1109/TPDS.2022.3167434
20 Ma, X., Zhou, Y., Wang, L., Miao, M.: Privacy-preserving byzantine-robust federated learning. Computer Standards & Interfaces **80**, 103561 (2022)
21 McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial intelligence and statistics. pp. 1273–1282. PMLR (2017)
22 Prakash, S., Avestimehr, A.S.: Mitigating byzantine attacks in federated learning. arXiv preprint arXiv:2010.07541 (2020)
23 Sattler, F., Müller, K.R., Wiegand, T., Samek, W.: On the byzantine robustness of clustered federated learning. In: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 8861–8865. IEEE (2020)
24 Shen, S., Tople, S., Saxena, P.: Auror: Defending against poisoning attacks in collaborative deep learning systems. In: Proceedings of the 32nd Annual Conference on Computer Security Applications. pp. 508–519 (2016)
25 So, J., Güler, B., Avestimehr, A.S.: Byzantine-resilient secure federated learning. IEEE Journal on Selected Areas in Communications **39**(7), 2168–2181 (2020)
26 Xie, C., Koyejo, O., Gupta, I.: Zeno: Byzantine-suspicious stochastic gradient descent. arXiv preprint arXiv:1805.10032 **24** (2018)
27 Xie, C., Koyejo, O., Gupta, I.: Fall of empires: Breaking byzantine-tolerant sgd by inner product manipulation. In: Uncertainty in Artificial Intelligence. pp. 261–270. PMLR (2020)

28 Xie, C., Koyejo, S., Gupta, I.: Zeno++: Robust fully asynchronous sgd. In: International Conference on Machine Learning. pp. 10495–10503. PMLR (2020)
29 Yang, Q., Liu, Y., Chen, T., Tong, Y.: Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology (TIST) **10**(2), 1–19 (2019)
30 Yin, D., Chen, Y., Kannan, R., Bartlett, P.: Byzantine-robust distributed learning: Towards optimal statistical rates. In: International Conference on Machine Learning. pp. 5650–5659. PMLR (2018)
31 Zhang, Z., Cao, X., Jia, J., Gong, N.Z.: Fldetector: Defending federated learning against model poisoning attacks via detecting malicious clients. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. pp. 2545–2555 (2022)
32 Zhu, H., Ling, Q.: Bridging differential privacy and byzantine-robustness via model aggregation. arXiv preprint arXiv:2205.00107 (2022)
33 Zhu, L., Liu, Z., Han, S.: Deep leakage from gradients. Advances in neural information processing systems **32** (2019)

Appendix A: Theoretical Analysis

In this section, we provide the theoretical guarantee of the convergence of the detection accuracy for the proposed FedPBF.

Assumption A.1. *For any two sets $\mathcal{I}_1$ and $\mathcal{I}_2$ with Q clients, where $\mathcal{I}_1$ is composed of benign clients and $\mathcal{I}_2$ includes at least one Byzantine client, we assume $\sum_{k\in\mathcal{I}_1} F_k(\mathbf{w}_1, \mathcal{D}_k) < \sum_{k\in\mathcal{I}_2} F_k(\mathbf{w}_2, \mathcal{D}_k)$* [6]*., where $\mathbf{w}_1 = \mathbf{w} - \frac{1}{Q}\sum_{k\in\mathcal{I}_1}\mathbf{g}_k$, $\mathbf{w}_2 = \mathbf{w} - \frac{1}{Q}\sum_{k\in\mathcal{I}_2}\mathbf{g}_k$ and $\mathbf{w}$ is the global weight.*

Theorem A.1. *Let K, M, Q and T be the number of clients, the number of candidate models, the number of aggregated updates for each candidate model and the number of global iterations, respectively. If Assumption A.1 holds and $1-(1-(\frac{m}{K})^Q)^M > \frac{m}{K}$, then the detection accuracy in Definition has*

$$\eta_T > \eta_{T-1} > \cdots > \eta_1 = 1-(1-(\frac{m}{K})^Q)^M > \frac{m}{K}. \tag{13}$$

Moreover, we have

$$\lim_{T\to\infty}\eta_T = 1, \tag{14}$$

and averaged detection accuracy as defined in Definition

$$\eta \geq 1-(1-(\frac{m}{K})^Q)^M. \tag{15}$$

Proof. Denote $\mathbf{p}_t$ to be the sample probability of all clients and $\mathbf{p}_1 = [1/K, \cdots, 1/K]$. Denote the p_t to be the probability of selecting benign clients and $p_1 = \frac{m}{K}$. Therefore, we have

$$\eta_t = 1-(1-p_t^Q)^m, \tag{16}$$

and $\eta_1 = 1-(1-(\frac{m}{K})^Q)^m$.
Notice that

$$\begin{aligned} p_t &= \eta_{t-1}\eta_{t-2}\cdots\eta_1\frac{m+tQ}{K+tQ} + \sum_{i=1}^{t-1}\Pi_{j\neq i}\eta_j(1-\eta_i)\frac{m+(t-1)Q}{K+tQ} \\ &+\cdots+(1-\eta_{t-1})(1-\eta_{t-2})\cdots(1-\eta_1)\frac{m}{K+tQ} \\ &= \frac{x_t}{K+tQ}, \end{aligned} \tag{17}$$

and

$$x_t = (1-\eta_{t-1})x_{t-1} + \eta_{t-1}x_{t-1} + Q\eta_{t-1}. \tag{18}$$

We would prove the η_t and p_t are two increasing sequence by induction.
Firstly, it is easy to verify $p_2 > p_1$ if $1-(1-(\frac{m}{K})^Q)^M > \frac{m}{K}$ holds. Then since $f(x) = 1-(1-x^Q)^m$ is increasing function, thus $\eta_2 > \eta_1$.
Secondly, if $\eta_{t-1} > \eta_{t-2}$ and $p_{t-1} > p_{t-2}$ hold, then according to Eq. (17) and (18), we have:

$$p_t = \frac{x_t}{K+tQ} = \frac{(1-\eta_{t-1})x_{t-1} + \eta_{t-1}x_{t-1} + Q\eta_{t-1}}{(k+tQ)} \tag{19}$$

[6] We assume that as long as there is at least one Byzantine client, the model performance must be degraded so that the loss will increase.

Therefore, we only need to prove

$$\begin{aligned} & p_t > p_{t-1} \\ \Longleftrightarrow \quad & \frac{(1-\eta_{t-1})x_{t-1} + \eta_{t-1}x_{t-1} + Q\eta_{t-1}}{(k+tQ)} > \frac{x_{t-1}}{K+(t-1)Q} \\ \Longleftrightarrow \quad & \eta_{t-1}(K+(t-1)Q) > x_{t-1} \\ \Longleftrightarrow \quad & \eta_{t-1} > p_{t-2} \end{aligned} \tag{20}$$

It is noted that $f(x) = 1-(1-x^Q)^m < x$ at the range $(0, b)$ and $f(x) = 1-(1-x^Q)^m > x$ at the range $(b, 1)$. Also, $f(m/K) > m/K$ and $p_{t-1} > p_{t-2} > \cdots > m/K$, thus we obtain

$$\eta_{t-1} > p_{t-1} > p_{t-2}. \tag{21}$$

Therefore, we derive $p_t > p_{t-1}$, since $f(x)$ is increasing, $\eta_t > \eta_{t-1}$. Consequently, the conclusion of induction is also true with t. This completes the proof of the η_t and p_t are two increasing sequences. Since $\eta_t < 1$ and $p_t < 1$, according to the monotone bounded convergence theorem, the sequence $\{\eta_t\}$ and $\{p_t\}$ have the limit η and p respectively. Suppose $\eta < 1$, then we take limits w.r.t t of Eq. (17) for both sides and obtain

$$p = 0, \tag{22}$$

which is a contradictory of $p > p_1 = m/K$. Noted that $\eta = f(p)$, thus $p = 1$.
Finally, we derive

$$\eta_t > \eta_{t-1} > \cdots > \eta_1 = 1-(1-(\frac{m}{K})^Q)^M > \frac{m}{K} \tag{23}$$

Moreover, we have

$$\lim_{t\to\infty} \eta_t = 1, \eta \geq 1-(1-(\frac{m}{K})^Q)^M. \tag{24}$$

Theorem A.1 demonstrates 1) the detection accuracy increases with the training iteration and converges to one, which reflects that our algorithm selects the benign updates more accurately during training and achieves the 100% accuracy finally, so the Byzantine Fault-tolerance in Definition is guaranteed ; 2) the detection accuracy has the lower bound $\bar{l} = 1-(1-(\frac{m}{K})^Q)^M$, which is influenced by M, Q and $\frac{m}{K}$ (experimental results in Appendix B.2 also elucidate this phenomenon). Specifically, we have

- the lower bound $\bar{l}$ is increasing w.r.t $\frac{m}{K}$ and $\lim_{\frac{m}{K}\to 1} \bar{l} = 1$, which is explained $\bar{l}$ improves as there are more benign clients.
- the lower bound $\bar{l}$ is increasing w.r.t M and $\lim_{M\to\infty} \bar{l} = 1$. It is reasonable that the $\bar{l}$ becomes larger when there are more candidate models to evaluate for the server.
- the lower bound $\bar{l}$ is decreasing w.r.t Q and $\lim_{Q\to\infty} \bar{l} = 0$. When the number of aggregated updates for each candidate model increases, the probability of the presence of Byzantine attackers for the candidate model is increasing, making it hard to choose the sets without any malicious updates. Therefore, the convergence of our proposed method is influenced.

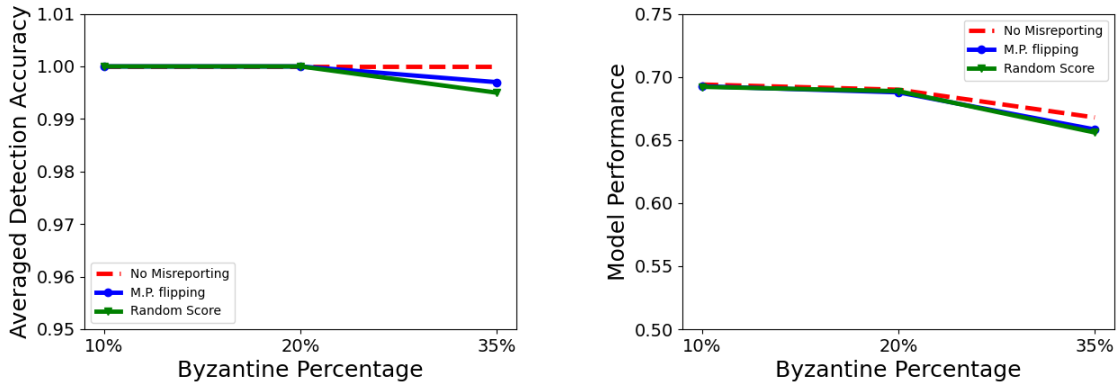


Fig 3: Averaged Detection accuracy (left) and model performance (right) of FedPBF with different Byzantine client percentages (10%, 20%, 35%) with misreporting under sign flipping attack (with IID setting for classification of CIFAR10).

Appendix B: Experiments

B.1 Robustness under Misreporting by Byzantine Clients

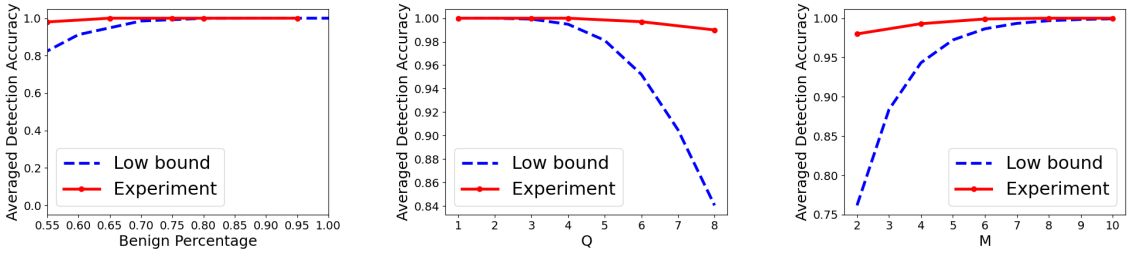
To confuse the judgment of the server, Byzantine clients may intentionally misreport model performance of candidate models for them to evaluate. Therefore, we adopt the median to filter out the misreported model performance by Byzantine clients as Eq. (10). We further evaluate the model performance on the CIFAR10 dataset with different Byzantine client percentages (10%, 20% and 35%) under three misreporting conditions: 1) **Mode Performance flipping (M.P. flipping):** Byzantine clients flip the model performances of candidate models, i.e., exchanging the highest j_{th} of the candidate model with the lowest j_{th} score of the candidate model for $j = 1, \cdots, \lfloor \frac{M}{2} \rfloor$. 2) **Random Score:** Byzantine clients randomly select a model performance value from 0 to 1. 3) **No Misreporting:** the Byzantine clients report the model performance honesty. Fig. 3 shows the model performance and averaged detection accuracy of FedPBF with Byzantine clients with different kinds of misreporting. It shows that the FedPBF is robust against misreporting, i.e. there is no performance degradation when Byzantine client percentage equals 10% and 20%. Even if the Byzantine client percentage reaches 35%, the averaged detection accuracy and model performance only drop less than 1%.

B.2 Ablation Study

To answer **Question 3**, we use CIFAR10 on AlexNet under the IID setting to test the reliability of the theoretical averaged detection accuracy lower bound in Sect. A. Fig. 4 (a) shows how the averaged detection accuracy is affected by $\frac{m}{K}$ for a fixed $Q = 3$ and $M = 10$. Fig. 4 (b) shows the changing trend of averaged detection accuracy when the Q value is changed at a fixed $M = 10$ and $\frac{m}{K} = 80\%$. Fig. 4 (c) shows the changing trend of averaged detection accuracy when the value of M is changed at a fixed $Q = 3$ and $\frac{m}{K} = 80\%$. The averaged detection accuracy of the experiment is consistently larger than the theoretical lower bound (blue dotted line), which means one can select appropriate hyperparameters M, Q to ensure required model performances. Therefore, we set $Q = 10$, $M = 40$ in experiments in Tab. 2 unless stated otherwise.

Hyper-parameter	Logistic Regression	LeNet	AlexNet
Number of Clients	100	100	20
Optimization method	SGD	Adam	SGD
Learning rate	0.125	0.0125	0.01
Weight Decay	0.0	0.002	0.001
Batch size	32	32	32
Data Distribution	Non-IID ($\beta = 0.5$)	Non-IID ($\beta = 0.5$)	IID and Non-IID ($\beta = 0.5$,1)
Local rounds	1 iteration	3 epochs	3 epochs
Communication rounds	3000	3000	600
M and Q	$M = 40, Q = 10$	$M = 40, Q = 10$	$M = 10, Q = 3$

Table 3: Hyper-parameters for training.



(a) M=10 Q=3 (b) M=10 $\frac{m}{K} = 80\%$ (c) Q=3 $\frac{m}{K} = 80\%$

Fig 4: Low bound (blue dotted line) and experiment (red solid line) averaged detection accuracy of FedPBF with different hyperparameters under sign flipping attack (with IID setting for classification of CIFAR10)

.