SkyMask: Attack-agnostic Robust Federated Learning with Fine-grained Learnable Masks

Peishen Yan[1], Hao Wang[2], Tao Song[1], Yang Hua[3], Ruhui Ma[1], Ningxin Hu[4], Mohammad Reza Haghighat[4], and Haibing Guan[1]

[1] Shanghai Jiao Tong University, Shanghai, China
{peishenyan,songt333,ruhuima,hbguan}@sjtu.edu.cn
[2] Stevens Institute of Technology, Hoboken, USA
hwang9@stevens.edu
[3] Queen's University Belfast, Belfast, UK
Y.Hua@qub.ac.uk
[4] Intel Corporation
{mohammad.r.haghighat,ningxin.hu}@intel.com

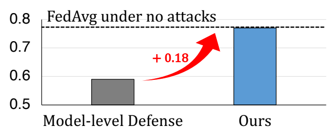
Abstract. Federated Learning (FL) is becoming a popular paradigm for leveraging distributed data and preserving data privacy. However, due to the distributed characteristic, FL systems are vulnerable to Byzantine attacks that compromised clients attack the global model by uploading malicious model updates. With the development of *layer-level* and *parameter-level* fine-grained attacks, the attacks' stealthiness and effectiveness have been significantly improved. The existing defense mechanisms solely analyze the *model-level* statistics of individual model updates uploaded by clients to mitigate Byzantine attacks, which are ineffective against the fine-grained attacks due to unawareness or overreaction. To address this problem, we propose SkyMask, a new attack-agnostic robust FL system that firstly leverages fine-grained learnable masks to identify malicious model updates at the parameter-level. Specifically, the FL server freezes and multiplies the model updates uploaded by clients with the parameter-level masks, and trains the masks over a small clean dataset (*i.e.*, *root dataset*) to learn the subtle difference between benign and malicious model updates in a high-dimension space. Our extensive experiments involve different models on three public datasets under state-of-the-art (SOTA) attacks, where the results show that SkyMask achieves up to 14% higher testing accuracy compared with SOTA defense strategies under the same attacks and successfully defends against attacks with malicious clients of a high fraction up to 80%. SkyMask will be open-sourced upon acceptance.

1 Introduction

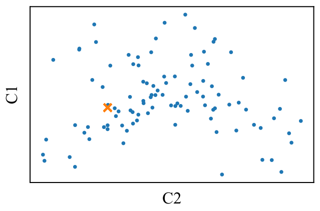
Federated learning (FL) [22] is a distributed machine learning paradigm that addresses the conflicts between ML training and data privacy. Instead of centralizing training data, FL distributes a global model to clients, trains the model locally, and aggregates the local models into a new global model without leaking any clients' local data, which has been applied in many computer vision applications, including large-scale visual classification [11], object detection [20], medical imaging [10, 18], and many others [17, 19, 34].

However, due to FL's distributed design, attackers can easily attack the FL system by compromising client participants and smuggling malicious model updates, known as Byzantine attacks [2, 3, 7, 25, 27, 31]. To tackle Byzantine attacks, researchers have explored various defense strategies, most of which leverage coarse-grained model-level statistics to detect outlier model updates [4, 5, 24, 26, 27] or greedily filter out outlier parameters [8, 32].

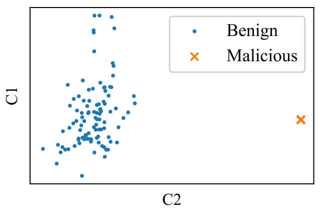
Unfortunately, *fine-grained* attacking methodologies utilize their precise balancing between attacking stealthiness and effectiveness to circumvent existing defense mechanisms, *e.g.*, Fang attack [7] and AGR-agnostic attack [27] leverage the varying sensitivities of different neural network layers and parameters to craft adaptive and stealthy attacks. Existing model-level defense strategies either fail to prevent such fine-grained attacks or spill over the benign clients, which extensively sacrifices training efficiency and model quality. We conduct a real-world experiment to show the poor performance of the model-level defense [29] under a fine-grained attack [27] (detailed settings provided in supplementary §A.5). Existing defense method exhibits 18% testing accuracy degradation. We visualize model updates of all clients in a single iteration by the principal component analysis (PCA) as [29]. As depicted in Fig. 1(b), the fine-grained attack camouflages the malicious updates within benign ones, underscoring the insufficiency of analyzing local model updates solely at the model-level.



(a) The testing accuracy under the fine-grained attack



(b) The PCA of model updates



(c) The PCA of masks

Fig. 1: Visualizing model updates and masks with PCA.

To explore whether binary masks [14, 36] can effectively detect malicious clients, we simultaneously train the binary masks for each model update on the server in the aforementioned toy experiment. Fig. 1(c) illustrates the visualization of trained masks using PCA, where masks applied to malicious model updates distinctly stand out among all masks. This highlights the capability of learnable masks to differentiate between benign and malicious model updates, and this method improves the testing accuracy by 18%.

Based on this observation, we propose SkyMask, a new Byzantine-robust FL system that defends against agnostic attacks with fine-grained learnable masks at the *parameter-level*. Specifically, after collecting local models from participant clients, the FL server *freezes* and *multiplies* each of the local model parameters with a learnable mask variable. Then, all masked local models are trained on a small and clean dataset (*i.e.*, the root dataset [5]) together. During the training process on the root dataset, the mask variables, which are applied to the frozen local model parameters, learn to improve the model accuracy. Particularly, the masks applied to malicious models are trained to *correct* their potential misbehaviors at the parameter-level. Since utilizing fine-grained learnable masks does *not* enforce customization for specific attacks or refactoring the FL training objective functions, SkyMask gains three superior capabilities beyond existing defense strategies: 1) effective and efficient detection of fine-grained stealthy attacks, 2) attack-agnostic defense against a wide spectrum of attack methods, and 3) strong compatibility and complementarity with existing FL and defense methodology. Our main contributions are as follows:

– We propose fine-grained learnable masks that can capture models' intrinsic characteristics at the parameter-level in a uniform high-dimension space.
– We develop SkyMask, a new attack-agnostic Byzantine-robust FL system, which applies fine-grained learnable masks to detect malicious clients and defend against Byzantine attacks.
– We empirically evaluate our SkyMask on various benchmarks under seven SOTA attacking methods and compare them with existing defense methods, where experimental results show that SkyMask achieves up to 14% higher testing accuracy compared with existing defenses.

2 Preliminaries & Related Work

2.1 Federated Learning

A typical FL system includes n clients and a server. Each client i has a local dataset D_i, $i = 1, \dots, n$. In each communication round t, the server selects a subset of the clients $\mathcal{N}_t$ to execute the following steps: 1) *Model distribution*: The server distributes the global model W_t to the selected clients. 2) *Local training*: The clients receive global model W_t as the initial model W^i_{t+1} for local training, and repeat $W^i_{t+1} := W^i_{t+1} - \beta \nabla f(W^i_{t+1}; D_i)$ for l local iterations, where $f(\cdot;\cdot)$ is the empirical loss function and β denotes the local learning rate. The corresponding local model update is $\Delta W^i_{t+1} = W_t - W^i_{t+1}$. 3) *Model aggregation*: After the training, each client i uploads its model weights W^i_{t+1} to the server for aggregation. FedAvg [22] performs weighted model averaging to update the global model: $W_{t+1} = \sum_{i \in \mathcal{N}_t} \frac{|D_i|}{\sum_{i \in \mathcal{N}_t} |D_i|} W^i_{t+1}$, where we set $|\mathcal{N}_t| = n$ for simplicity.

2.2 Threat Model & Byzantine Attacks on FL

Malicious clients compromised by the attacker participate in the FL process and have access to the knowledge of neural network models, learning rates, and objective functions. Besides, the attacker can fully control these clients' activities (*e.g.*, modifying the model updates) and has complete access to their local datasets [5, 7, 27].

Existing Byzantine attacks on FL can be categorized into *untargeted* and *targeted* based on the adversary's goal: *1) Untargeted attacks* seek to corrupt the global model and minimize its accuracy on any test input [2, 7, 27, 29]. Specifically, fine-grained attacks, represented by Fang attack [7] and AGR-agnostic attack [27], adaptively trade off the attacking stealthiness and effectiveness at a fine granularity as an optimization problem, which preserves the effects of poisoning with as few model-level outliers as possible. *2) Targeted attacks* aim to reduce the utility of the global model on attacker-specified tasks [1, 3, 15, 21, 28, 31, 33].

2.3 Byzantine-robust FL Algorithms

We classify existing representative Byzantine robust algorithms into two categories based on their strategies:
Model-level defense strategies. Some defense methods take model-level distance as a basis for determining whether a client is malicious, *e.g.*, Euclidean distance in Krum [4], or cosine distance in DeepSight [26] and FLAME [24]. FLTrust [5] trains a root model with a small root dataset, and takes the cosine similarity between a local model and the root model as the weight in aggregation. Tolpegin defense [29] identifies malicious model updates by employing PCA dimension reduction and clustering. FLDetector [35] predicts a client's model update based on historical data and compares it with the received update. It employs consistency analysis on model updates to identify malicious clients.
Greedy parameter-level filtering strategies. Trimmed-Mean (Trim) [32] is a coordinate-wise aggregation rule assuming the number of malicious clients n_m is known, removes the parameters of the smallest and largest n_m values, and averages the remaining.

However, existing robust FL strategies can hardly defend against those fine-grained attacks without sacrificing training efficiency and model quality by utilizing coarse-grained detection or greedy parameter-wise filtering, which motivates us to explore malicious client detection with fine-grained learnable masks.

2.4 Root Dataset

Considering that the server has no root of trust to decide whether a model update is malicious or not, FLTrust [5] was the first to introduce the concept of root dataset. It can take advantage of the root dataset even if the distribution diverges from the overall data distribution or the size is less than one hundred, which is easy for the server and FL manager to manually collect or generate a small root dataset. The rationale for utilizing root datasets has been validated in numerous other studies [9, 16, 23, 30]. We evaluated the impact of the root dataset' data distribution in supplementary §C.1.

3 Methodology

We aim to design a server-based robust FL system. The server has a small representative root dataset following a previous work [5]. Moreover, this root dataset

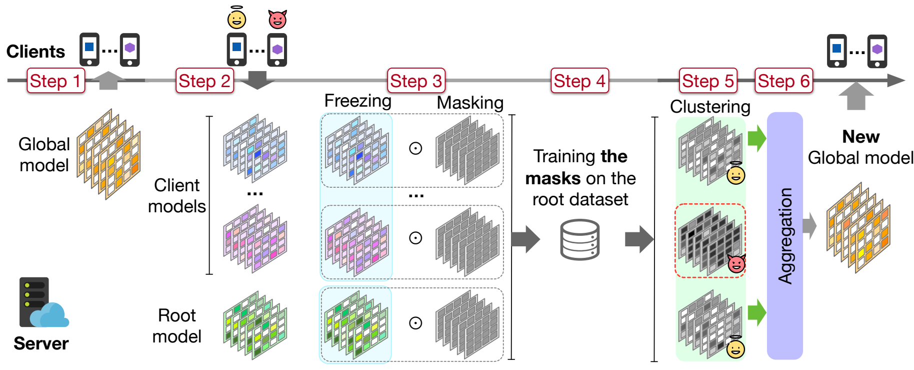


Fig. 2: SkyMask's workflow.

can be imbalanced. For the proposed Byzantine-robust FL system, it should promise the following three features: (1) **Robustness**. The system should preserve the global model accuracy under attacks. In particular, we should maintain a high malicious client detection accuracy and a low misidentification rate of benign ones to eliminate the impact of attacks. (2) **Generality**. The system should also stay effective on different datasets and model structures under different attacks. (3) **Efficiency**. The defense sysyem is designed on the server instead of resource-constrained client. Besides, when performing malicious client detection and defense on the server, we allow reasonable additional computation costs and extra storage space.

3.1 Overview

When initializing the FL process, the server creates an initial global model and builds a root dataset as [5]. Fig. 2 presents SkyMask's workflow, where each communication round has six steps: **Step 1**, the server distributes the global model parameters to each client. **Step 2**, the clients load the parameters, train the local models, and send back these model updates. **Step 3**, the server freezes and multiplies each local model parameter with a learnable mask variable. **Step 4**, all the masked local models are trained together on the root dataset to converge. **Step 5**, the server detects and removes the malicious clients by clustering all trained masks. **Step 6**, the server aggregates the remaining model updates into a new global model.

The key component of SkyMask is the learnable mask-driven detection. Specifically, SkyMask constructs a same-size fine-grained mask for each local model update and trains these masks by optimizing the aggregation result of masked local models until they converge. Byzantine attacks take impairing global model performance as an optimization objective, so the poisoned model updates tend to make the global model far from normal behavior. Therefore, to correct potential misbehaviors of malicious models, the masks applied to the malicious models learn to mitigate the side effects of poisoned parameters through training on the root dataset. In this way, the learnable masks form a high-dimensional representation space to capture the characteristics of both malicious and benign model updates. Then, the clusters formed by masks are the basis for determining whether a client is malicious. After removing all the detected malicious model updates, SkyMask calculates the average of remaining model updates and uses it to update the global model.

3.2 SkyMask Algorithm

Most existing defense strategies are based on the local model updates' coarse-grained statistics or greedily filtering out outlier parameters. However, fine-grained attacks corrupt a small set of particular layers or parameters to work around existing defense strategies easily. Thus, we apply fine-grained learnable masks and clustering analysis to detect and defend against malicious clients. Our defense strategy has two stages: 1) mask initialization and training stage; 2) mask clustering and classification stage.

Mask initialization and training. In the first stage, the server freezes all the model updates and assigns a learnable mask m_i of the same size as the model for each client i and initializes them all with 1s. The model aggregated from masked local models $\tilde{W}_{t+1}$ (called aggregated masked model) is computed by averaging the masked local models $m_i \odot W^i_{t+1}$. Then, the masks start training on the root dataset with frozen local models. At the beginning of iteration $t+1$, $\tilde{W}_{t+1}$ is calculated as:

$$\tilde{W}_{t+1} = \sum_{i=1}^{n} \frac{1}{n} m_i \odot W^i_{t+1}. \tag{1}$$

We train the aggregated masked model $\tilde{W}_{t+1}$ on the root dataset. Then, the update process back-propagates gradients with the help of $\tilde{W}_{t+1}$ to masks and applies a gradient descent algorithm to the masks.

If there is no constraints on the value range of mask variables during the mask training, some masks can have very large values, and others can be so small for the same dimension when these masks converge, which can not effectively characterize whether the local model is benign. So we cannot detect malicious clients by the masks ranged $(-\infty, +\infty)$, and we should redesign the masks to show the contribution of each local model update in the optimized model.

We use binary masks to extract the parameter-level characteristics of local models because binary can directly determine whether a parameter of model update participates in the global model. Since binary parameters are not derivable, we approximate the binary masks in training by setting the mask with real values and applying a sigmoid function $\sigma(\cdot)$ instead of a hard threshold to reduce the enormous gradient variance [14]. Furthermore, it is easy for the gradient to be back-propagated to the real-value masks. The approximation of a binary mask is calculated as $\tilde{m}_i = \sigma(m_i)$, which is ranged $(0, 1)$.

Since the mask's limitation is changed, the original Equation 1 is unsuitable. To adapt for the approximated binary masks, $\tilde{W}_{t+1}$ is re-written as:

$$\tilde{W}_{t+1} = \frac{\sum_{i=1}^{n} \tilde{m}_i \odot W_{t+1}^i}{\sum_{i=1}^{n} \tilde{m}_i}. \tag{2}$$

It can be formulated as an element-wise weighted averaging algorithm. If the parameters of a poisoned local model are toxic to the aggregated masked model $\tilde{W}_{t+1}$, the mask tends to reduce the involvement of these parameters to $\tilde{W}_{t+1}$ in the optimization phase. Thus, the 0-1 pattern in the mask can represent whether the corresponding local model is malicious.

The mask training task is not different from the main task in FL except for the variables. The objective function is:

$$f(\tilde{W}_{t+1}, D_r) = \sum_{(x,y)\in D_r} L(\text{output}(x, \tilde{W}_{t+1}), y). \tag{3}$$

The masks are updated in each mask training iteration as follows:

$$m_i := m_i - \gamma \cdot \nabla_{m_i} f(\tilde{W}_{t+1}, D_r), \tag{4}$$

where γ represents the mask learning rate. This process will not stop until these masks converge. After the real-value masks converge, the final results of binary mask $\hat{m}_i$ are computed as:

$$\hat{m}_i[k] = \begin{cases} 1, & \tilde{m}_i[k] > \tau \\ 0, & \tilde{m}_i[k] \le \tau \end{cases}, \tag{5}$$

where $\hat{m}_i[k]$ represents the parameter located in kth dimension of binary mask $\hat{m}_i$ of client i and τ is a threshold.

Mask clustering and classification. These binary masks can sense potential attack at a fine granularity and represent the parameter-level characteristics of each client's model update. Then, we apply the Gaussian mixture model for clustering and classification. If there are no attacks, the clustering result is only one cluster, and the server aggregates all the benign local models.

To determine which cluster is benign, SkyMask introduces a trusted root model W_{t+1}^r, initialized by W_t and "locally" trained l iterations on the root dataset as other clients:

$$W_{t+1}^r := W_{t+1}^r - \beta \nabla f(W_t, D_r). \tag{6}$$

The server lets it join in the model set before the mask training stage, assigns one mask for it and trains this mask together with other masks. When all the masks converge, the clients represented by the masks in the same cluster as the mask corresponding to the trusted root model are considered benign. This way, SkyMask can handle various malicious fractions from zero to very high. In order not to interfere with the global model convergence, we should emphasize that the trusted root model is only used for malicious model detection and is not included in the benign model aggregation phase.

SkyMask plays R rounds for server-clients communication, and each round contains six steps, as §3.1 describes. The critical point of SkyMask is our fine-grained malicious client detection strategy mentioned above, which can provide the remaining benign client set U_b for aggregating a new global model. Given the aggregation algorithm specified as FedAvg, the global model for the next communication round W_{t+1} is calculated as:

$$W_{t+1} = W_t - \alpha \cdot \sum_{i \in U_b} \frac{|D_i|}{\sum_{i \in U_b} |D_i|} \Delta W^i_{t+1}, \tag{7}$$

where α denotes the global learning rate. If it completes the malicious detection task, removing only the malicious model updates and leaving the benign model updates untamed, its convergence will be consistent with the original aggregation algorithm. SkyMask serves as an efficient method for identifying malicious clients within an FL framework. Its modular design allows for integration as an adjunct component onto diverse aggregation algorithms, ensuring compatibility across a spectrum of existing aggregation techniques.

3.3 Why SkyMask Works

We conduct an example experiment to show how SkyMask works. We choose a four-layer CNN model and Fashion-MNIST dataset and set 20 of 100 clients as malicious. Under Fang-Trim attack, we take the mask training process in one communication round. We calculate the proportion of "1"s in each part of the binary mask. A smaller value means fewer parameters of this client are selected for the aggregated masked model. For simplicity, here only shows 20 masks for malicious and 20 for benign clients in each mask training iteration.

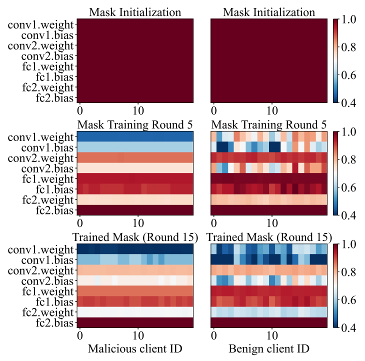


Fig. 3: The visualization of the training process of masks applied to malicious and benign models.

As Fig. 3 shown, during the mask training process, the binary parameters in the "conv1.weight", "conv2.weight", and "fc1.weight" parts of malicious model updates are set to zero more than those of benign updates. It is related to the fact that the attack prefers to poison the parameters in specific parts. The masks gradually reduce the poisoning effect of the toxic parameters during the training so that we can classify malicious clients with the help of the masks. Therefore, SkyMask can perceive the poisoning attacks at the parameter-level in the model updates without knowing the details of the attack, which makes it attack-agnostic.

3.4 SkyMask's Complexity

Let $O(T)$ denote the computational complexity of one local training iteration, and the model parameters' dimensionality is V. To update the masks once, the server computes the gradient of each mask from the gradient of the aggregated masked model $\tilde{W}_{t+1}$ by backpropagation, with a computational complexity of $O(nV)$. Hence, the total computational complexity is $O((T+nV)t_m)+O(nV) = O(Tt_m+nVt_m)$, where t_m is the number of mask training convergence iterations. Since there is no data dependency between any mask parameters, we can use parallel training [6] to accelerate the mask optimization process so that the time overhead can be further reduced, trading computation power for time.

The computational complexities of existing defense methods are as follows: $O(nV)$ for FLTrust, $O(nV \log n)$ for Trim, $O(n^2V)$ for Krum, $O(T + nV)$ for DeepSight, and $O(nV)$ for FLAME. The above analysis shows that the additional cost of the mask learning mechanism over other defense methods is equivalent to the cost of conducting several iterations of local training, which is acceptable compared to the computational power of the server.

The space complexity of existing FL defense algorithms is either $O(n + nV)$ or $O(n^2 + nV)$. Since we assign a mask for each client, the additional space complexity is also proportional to the number of clients n. It is proportional to the space complexity of the global model parameters because the masks have the same dimensionality as the global model. Considering that the server stores n local models' parameters in the current round, the total space cost is about twice the others, and its space complexity is $O(nV)$, which is reasonable.

4 Experiments

We evaluate our method with existing Byzantine-robust FL aggregation algorithms and two malicious client detection algorithms. To verify the generalization of our method, we conduct experiments using various popular datasets paired with different models. We evaluate our method with seven popular attacks, explore the impact of the fraction of malicious clients, and verify its scalability. In sensitivity analysis, we discuss the influence of binarization threshold, the number of features after dimensionality reduction, and root dataset's data distribution on our method's performance (details in supplementary §C).

On Fashion-MNIST and CIFAR-10 datasets, SkyMask improves the testing accuracy by 1–9% compared to the best results achieved by other algorithms under the same attacks. More significantly, our method gains a 14% improvement in testing accuracy on CIFAR-100 dataset under Min-Max attack.

4.1 Experiment Settings

Datasets and models. We evaluate SkyMask on three widely-used datasets: Fashion-MNIST, CIFAR-10, and CIFAR-100 [13]. Following previous work [5, 7], we construct non-IID local dataset with bias probability $p = 0.5$ for each client and the root dataset for the server. We construct a CNN with two convolutional layers and two linear layers as the global model for Fashion-MNIST dataset. For CIFAR-10 and CIFAR-100 datasets, to verify SkyMask's effectiveness on large models in FL, we use a more complex model, ResNet20.

Baseline attacks and defenses. The experiments cover *1) untargeted attacks*: Label-flipped (LF) [29] and four fine-grained attacks, including Fang attacks [7] (Fang-Trim and Fang-Krum attacks) and two types of the AGR-agnostic attack [27] (Min-Max and Min-Sum attacks); *2) targeted attack*: Scaling attack [1] and DBA [31]. In §4.2, to show the defense effectiveness, we compare SkyMask with five Byzantine-robust FL aggregation algorithms: FLTrust [5], Trim [32], Krum [4], DeepSight [26] and FLAME [24]. In §4.3, to reveal the significance of learnable masks in malicious client detection, we compare SkyMask with two malicious client detection algorithms: Tolpegin defense [29] and FLDetector [35].

FL parameter settings. Following the previous study [5], we use 100 clients in all experiments and apply multiple-client attacks. The default fraction of malicious clients is set to 20%. In our experiments, the root dataset has 100 samples and no intersection with the training and test datasets. This setting is similar to FLTrust [5]. 100 is enough for our defense, and it is easy for the server to collect. So we do not discuss the influence of the size of the root dataset. Other hyperparameters are detailed in supplementary §A.4.

SkyMask variants. Intuitively, if the fraction of malicious clients is less than 50% in the real world, we can choose the larger one in two clusters of masks as benign. This way, we can further reduce the computational pressure on the server. To verify the effectiveness with or without the trusted root model, we consider the following two schemes: **SkyMask-NR** (no trusted root model): The FL system does not generate a trusted root model, and after all the masks are converged, the system chooses the larger one in two clusters of the masks as benign for aggregation. **SkyMask**: The FL system uses the defense strategy fully following the default steps described in §3.2.

Metrics. The **testing accuracy** is obtained from the prediction accuracy of the global model on the test dataset. The **attack success rate** under targeted attack is the fraction of the trigger-embedded samples in the test dataset that are identified as the target label. To show the malicious client detection ability, we record **false positive rate (FPR)** and **false negative rate (FNR)** in the malicious client detection stage. FPR indicates the number of benign clients that are misidentified as malicious clients divided by the total number of benign clients, and FNR indicates the number of malicious clients that are misidentified as benign clients divided by the total number of malicious clients. The Mean FPR/FNR is the average of FPR/FNRs in each communication round.

Table 1: FL testing accuracy under different attacks and attack success rates of targeted attack. The experimental results of targeted attack are in the form of "testing accuracy/attack success rate."

Dataset (Model)	Attack	FedAvg	FLTrust	Trim	Krum	DeepSight	FLAME	**SkyMask-NR**	**SkyMask**
Fashion -MNIST (CNN)	None	**0.89**	**0.89**	0.88	0.83	0.88	0.87	**0.89**	**0.89**
	LF	0.84	0.86	0.84	0.83	**0.89**	0.87	**0.89**	**0.89**
	Min-Max	0.58	**0.89**	0.70	0.83	0.64	0.65	**0.89**	**0.89**
	Min-Sum	0.80	**0.89**	0.73	0.47	0.82	0.75	**0.89**	**0.89**
	Fang-Trim	0.42	**0.89**	0.67	0.82	0.75	0.85	**0.89**	**0.89**
	Fang-Krum	0.86	**0.89**	0.84	0.47	0.70	0.78	**0.89**	**0.89**
	Scaling	0.80/0.21	**0.89/0.08**	0.87/0.13	0.82/0.09	0.88/0.10	0.87/0.10	0.89/0.10	0.89/0.10
	DBA	0.86/0.51	0.88/0.18	0.88/0.31	0.83/0.11	0.89/0.16	0.89/0.11	0.89/0.11	**0.89/0.10**
CIFAR-10 (ResNet20)	None	**0.77**	0.75	**0.77**	0.54	0.72	0.72	0.76	0.76
	LF	0.71	0.74	0.71	0.55	0.69	0.68	0.75	**0.76**
	Min-Max	0.58	0.68	0.70	0.52	0.63	0.55	**0.77**	**0.77**
	Min-Sum	0.66	0.72	0.74	0.32	0.66	0.68	0.75	**0.77**
	Fang-Trim	0.10	0.68	0.19	0.52	0.53	0.49	0.75	**0.76**
	Fang-Krum	0.58	0.75	0.48	0.19	0.36	0.40	**0.77**	**0.77**
	Scaling	0.10/1.00	0.74/0.13	0.72/0.51	0.53/0.08	0.73/0.10	0.73/0.10	**0.77/0.09**	0.77/0.10
	DBA	0.72/0.99	0.76/0.88	0.76/0.97	0.56/0.09	0.77/0.16	0.77/0.14	0.77/0.11	**0.77/0.10**
CIFAR-100 (ResNet20)	None	**0.44**	0.39	0.43	0.17	**0.44**	**0.44**	**0.44**	**0.44**
	LF	0.41	0.39	0.38	0.11	0.43	0.43	**0.44**	**0.44**
	Min-Max	0.16	0.30	0.16	0.05	0.16	0.19	**0.44**	**0.44**
	Min-Sum	0.33	0.28	0.33	0.17	0.35	0.34	**0.44**	**0.44**
	Fang-Trim	0.01	0.34	0.04	0.15	0.20	0.22	**0.44**	**0.44**
	Fang-Krum	0.03	0.37	0.04	0.03	0.03	0.02	**0.44**	**0.44**
	Scaling	0.01/1.00	0.36/0.30	0.43/0.98	0.14/0.00	0.44/0.04	0.44/0.08	**0.44/0.01**	**0.44/0.01**
	DBA	0.37/0.98	0.40/0.90	0.44/0.94	0.16/0.01	0.45/0.16	0.45/0.15	0.44/0.02	**0.44/0.01**

4.2 The Defense Effectiveness of SkyMask

We first certificate the defensive capability under a low fraction of attacks. Then, there is also possibly no attacker in the FL system, so the defense methods should not affect the global model's performance. Eventually, we consider an extreme condition that the fraction of malicious clients is high.

Under a low fraction of attacks. By default, we set the fraction of malicious clients to 20% following the default setting in [5]. Table 1 shows that, FLTrust is a relatively strong baseline defense that it achieves similar good performance as ours in a few Fashion-MNIST experiments. While under LF attack, FLTrust has a 3% accuracy loss. Our SkyMask achieves the highest testing accuracy on Fashion-MNIST under different attacks compared to all others.

To demonstrate the generalization, we experiment on CIFAR-10 and CIFAR-100 dataset using a larger model, ResNet20. In such scenarios, multi-client fine-grained attacks are much more powerful. Trim, Krum, DeepSight, and FLAME are out of defense. FLTrust does not perform as well as it does on Fashion-MNIST dataset, with a severe drop in accuracy under various attacks. In contrast, our SkyMask reaches the highest accuracy under all attacks. Especially on CIFAR-100 dataset, our method achieves the most significant gap with other algorithms in Min-Max attack experiments. The accuracy is up to 0.44, while the accuracy results obtained by others are less than 0.3. Besides, SkyMask achieves the best and unattacked-level performance under the targeted attacks.

Due to the attackers' optimization for the abnormality degree, only some particular parameters of the poisoning models changed with emphasis. The results show that the four fine-grained attacks (Min-max, Min-Sum, Fang-Trim, and Fang-Krum attacks) bypass the existing defense methods in many cases.

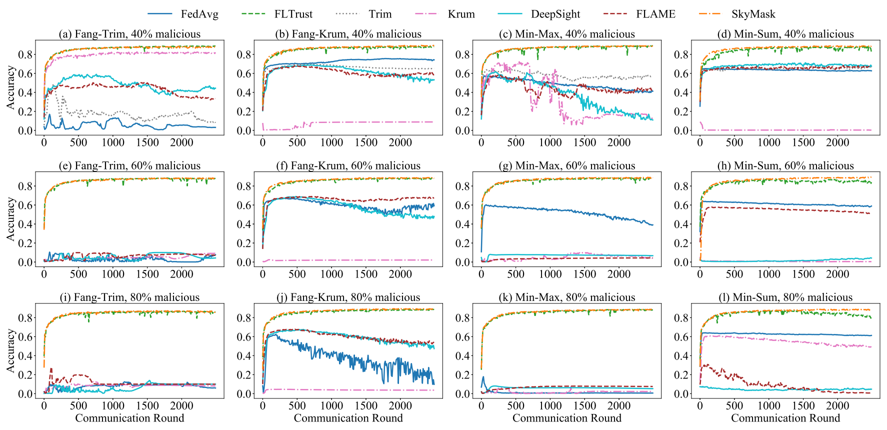


Fig. 4: The impact of high fractions of malicious clients under fine-grained attacks.

In all these experiments, SkyMask not only achieves the highest testing accuracy but reaches the same accuracy level as unattacked FedAvg's, *i.e.*, the same accuracy or no more than 1% accuracy loss.

As §4.1 describes, the server does not need to consume resources to generate a root model if the fraction of malicious clients is less than 50%. In the 20% case, as shown in Table 1, SkyMask-NR achieves good results under any attack and on any dataset, which shows the same defensive capability as SkyMask. Therefore, if the server confirms that the number of malicious clients is less than half the total number, it can choose SkyMask-NR as the defense strategy to reduce the overhead while maintaining the same defensive capability. For generalization, SkyMask can be applied to handle more cases.

Under no attack. Some attacks send poisoned models to the server only at certain rounds, and some model updates can be mistaken for malicious by existing defense methods due to data heterogeneity. The system collapses if the global model performance suffers from the defense, even when there is no attack. So when there is no attack, the expected result is that any method should not affect the performance achieved by the basic aggregation algorithm FedAvg.

In Table 1, almost all other robust aggregation algorithms affect the testing accuracy in the no-attack case. For instance, FLTrust has an accuracy loss of about 2% on CIFAR-10 dataset and 5% on CIFAR-100. Krum algorithm has an accuracy loss ranging from about 6% to 27% on different datasets. At the same time, our SkyMask maintains a comparable performance as unattacked FedAvg on all datasets with a difference of less than 1%. The existing robust aggregation algorithms either choose a subset of model updates that seem benign or use model updates' statistics to correct the impact. Hence, it is possible to introduce harmful impacts without attack. If there is no attack, our masks form only one cluster, or there are only several outliers. So the server selects most of clients and has the same or very similar convergence with FedAvg.

Table 2: Testing accuracy, FPR, and FNR of different malicious client detection methods under different attacks. The experimental results of targeted attack are in the form of “testing accuracy/attack success rate.”

Dataset (Model)	Attack	Testing accuracy			FPR			FNR		
		Tolpegin	FLDetector	SkyMask	Tolpegin	FLDetector	SkyMask	Tolpegin	FLDetector	SkyMask
CIFAR10 (ResNet20)	None	0.76	0.75	0.76	0.00%	18.2%	0.00%	/	/	/
	LF	0.70	0.72	**0.76**	13.1%	**0.03%**	4.72%	19.9%	25.53%	**2.60%**
	Min-Max	0.61	0.11	**0.77**	36.5%	100%	**0.00%**	88.0%	100%	**0.00%**
	Min-Sum	0.59	0.31	**0.77**	38.8%	100%	**0.00%**	78.0%	100%	**0.00%**
	Fang-Trim	0.76	0.74	0.76	0.00%	0.03%	0.00%	0.00%	0.00%	0.00%
	Fang-Krum	0.17	0.31	**0.77**	37.4%	87.18%	**0.00%**	84.0%	100%	**0.00%**
	Scaling	0.76/0.10	0.74/0.10	**0.77/0.11**	0.00%	0.00%	0.00%	0.10%	0.00%	0.00%
	DBA	0.65/0.47	0.77/0.10	0.77/0.10	0.00%	0.00%	0.00%	46.1%	0.00%	0.00%

Under a high fraction of attacks. The experiments mentioned in the previous sections set the fraction of malicious clients to 20%, while in a real application environment, an attacker can control a larger fraction of malicious clients for the attack. Therefore, to verify the defensive capability of our system in this case, we use CNN as the global model on Fashion-MNIST dataset and conduct comparison experiments with 40%, 60%, and 80% malicious fractions. We present the results pertaining to fine-grained attacks in Fig. 4. Additional results can be found in the supplementary §B.1.

Only SkyMask and FLTrust maintain the defensive capability under fine-grained attacks. The other defense algorithms completely lose their defensive capability, exhibiting a significant gap from the unattacked level. From Fig. 4(b), Fig. 4(c), and Fig. 4(d), we can see that SkyMask converges more stable than FLTrust when they are attacked by Fang-Krum, Min-Max, and Min-Sum attacks. In Fig. 4(d), the accuracy obtained by SkyMask is 2% higher than FLTrust's under Min-Sum attacks.

When the fraction is more than 50%, only SkyMask maintains the defensive capability. The best baseline defense, FLTrust's performance worsens as the fraction rises. FLTrust's testing accuracy fluctuates more, and the global model even converges in the wrong direction (*e.g.*, in Fig. 4(l), the accuracy obtained by FLTrust decreases from 85% to 81% after a sharp fluctuation). Under other attacks, the increasing fraction influences FLTrust's convergence speed, whereas SkyMask remains unaffected and converges faster than FLTrust.

4.3 The Significance of Learnable Masks

We conduct comparative experiments on SkyMask and the other malicious client detection algorithms, *i.e.*, Tolpegin defense [29] and FLDetector [35]. We sample the result of malicious client detection every ten communication rounds and calculate all the metrics described in §4.1.

In Table 2, we can see that FLDetector only works under LF attack, Fang-Trim attack and targeted attack. Min-Max and Min-Sum attacks make FLDetector completely confuse the malicious model updates with the benign model updates. Tolpegin defense only survives under Fang-Trim attack, and all other attacks greatly harm it. The large FNR of Tolpegin defense means many malicious clients complete their attacks. The large FPR of Tolpegin defense shows that the remaining benign clients may lose some representation, so Tolpegin defense achieves a lousy performance. Our SkyMask achieves the same prediction accuracy as a no-attack case, with FPR less than 5% and FNR less than 3%.

Fig. **??** and Fig. **??** show the trend of testing accuracy on Fashion-MNIST with a CNN global model under Fang-Krum and Min-Sum attack. Detection failure means that the FL defense system does not detect all the malicious clients in that round. Under Fang-Krum attack, Tolpegin defense converges but yields poor testing accuracy, while FLDetector is rendered ineffective. Additionally, both FLDetector and Tolpegin defense exhibit fluctuating and low testing accuracy under Min-Sum attack due to detection failures. In contrast, SkyMask stably converges without encountering any detection failures.

CIFAR-10	Test Acc.		FPR		FNR	
(ResNet20)	200	500	200	500	200	500
FedAvg	0.75	0.73	/	/	/	/
None	0.75	0.73	/	/	/	/
LF	0.75	0.72	2.94%	6.36%	3.62%	8.36%
Min-Max	0.75	0.74	0.00%	0.00%	0.00%	0.00%
Min-Sum	0.74	0.73	0.00%	0.00%	0.00%	0.00%
Fang-Trim	0.75	0.73	0.00%	0.00%	0.00%	0.00%
Fang-Krum	0.75	0.72	0.00%	0.00%	0.00%	0.00%
Scaling	0.75/0.09	0.74/0.11	0.00%	0.00%	0.00%	0.00%
DBA	0.75/0.10	0.73 /0.10	0.00%	0.00%	0.00%	0.00%

4.4 SkyMask's Scalability

To demonstrate the scalability of SkyMask, we conduct experiments utilizing a ResNet20 global model trained on CIFAR-10. We assess the prediction accuracy of the main task, along with the FPR and FNR of SkyMask under various attack scenarios, encompassing 100, 200, and 500 clients. The results of 100 clients have been shown in Table 2.

In Table **??**, we observe a decrease in the testing accuracy results of the unattacked FedAvg as the total number of clients increased. This can be attributed to the challenges posed by the smaller size of the local datasets and the larger number of clients in the FL training process [12]. For the LF attack, SkyMask achieves comparable testing accuracy to the unattacked FedAvg, despite a slight increase in the FNR to approximately 9%. For all fine-grained attacks and targeted attacks, SkyMask maintains FPR and FNR at 0%. Our SkyMask algorithm demonstrates robust performance under all attacks.

Therefore, SkyMask can work well when the total number of clients increases. Even if there are millions of clients in the FL system, the server can randomly choose a fixed number (*e.g.*, 100 or 500) of clients in each communication round to execute malicious client detection and aggregation.

5 Conclusion

We propose a new attack-agnostic robust FL framework called SkyMask to defend against Byzantine attacks. By training parameter-level learnable binary masks on a clean root dataset, SkyMask is the first to implement a fine-grained detection of the poisoned elements of local model updates. Extensive experiments on non-IID datasets under different attacks prove the effectiveness, generality, and scalability of SkyMask. Our method shows a solid defensive capability, better than various robust aggregation algorithms and existing malicious client detection methods. Moreover, our SkyMask also tackles the problem that most clients are malicious and can defend against attacks with a fraction of malicious clients up to 80%.

References

1. Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., Shmatikov, V.: How to Backdoor Federated Learning. In: Proc. AISTATS (2020)
2. Baruch, G., Baruch, M., Goldberg, Y.: A Little is Enough: Circumventing Defenses for Distributed Learning. In: Proc. NeurIPS (2019)
3. Bhagoji, A.N., Chakraborty, S., Mittal, P., Calo, S.: Analyzing Federated Learning Through an Adversarial Lens. In: Proc. ICML (2019)
4. Blanchard, P., El Mhamdi, E.M., Guerraoui, R., Stainer, J.: Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent. In: Proc. NeurIPS (2017)
5. Cao, X., Fang, M., Liu, J., Gong, N.Z.: FLTrust: Byzantine-Robust Federated Learning via Trust Bootstrapping. In: Proc. NDSS (2021)
6. Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Mao, M., Ranzato, M.a., Senior, A., Tucker, P., Yang, K., Le, Q., Ng, A.: Large Scale Distributed Deep Networks. In: Proc. NeurIPS (2012)
7. Fang, M., Cao, X., Jia, J., Gong, N.: Local Model Poisoning Attacks to Byzantine-Robust Federated Learning. In: Proc. USENIX Security (2020)
8. Guerraoui, R., Rouault, S., et al.: The Hidden Vulnerability of Distributed Learning in Byzantium. In: Proc. ICML (2018)
9. Guo, H., Wang, H., Song, T., Hua, Y., Lv, Z., Jin, X., Xue, Z., Ma, R., Guan, H.: Siren: Byzantine-robust federated learning via proactive alarming. In: Proc. SoCC (2021)
10. Guo, P., Wang, P., Zhou, J., Jiang, S., Patel, V.M.: Multi-institutional collaborations for improving deep learning-based magnetic resonance image reconstruction using federated learning. In: Proc. CVPR (2021)
11. Hsu, T.M.H., Qi, H., Brown, M.: Federated visual classification with real-world data distribution. In: Proc. ECCV (2020)
12. Kamp, M., Fischer, J., Vreeken, J.: Federated learning from small datasets. In: Proc. ICLR (2023)
13. Krizhevsky, A., Hinton, G., et al.: Learning Multiple Layers of Features from Tiny Images. Technical Report (2009)
14. Li, A., Sun, J., Zeng, X., Zhang, M., Li, H., Chen, Y.: Fedmask: Joint computation and communication-efficient personalized federated learning via heterogeneous masking. In: Proc. SenSys (2021)

15. Li, H., Ye, Q., Hu, H., Li, J., Wang, L., Fang, C., Shi, J.: 3dfed: Adaptive and extensible framework for covert backdoor attack in federated learning. In: Proc. IEEE S&P (2023)
16. Li, Q., Liu, Z., Li, Q., Xu, K.: martfl: Enabling utility-driven data marketplace with a robust and verifiable federated learning architecture. In: Proc. CCS (2023)
17. Li, Y., Wang, X., Yang, L., Feng, L., Zhang, W., Gao, Y.: Diverse cotraining makes strong semi-supervised segmentor. In: ICCV (2023)
18. Liu, J., Zhang, Y., Chen, J.N., Xiao, J., Lu, Y., A Landman, B., Yuan, Y., Yuille, A., Tang, Y., Zhou, Z.: Clip-driven universal model for organ segmentation and tumor detection. In: Proc. CVPR (2023)
19. Liu, Q., Chen, C., Qin, J., Dou, Q., Heng, P.A.: Feddg: Federated domain generalization on medical image segmentation via episodic learning in continuous frequency space. In: Proc. CVPR (2021)
20. Liu, Y., Huang, A., Luo, Y., Huang, H., Liu, Y., Chen, Y., Feng, L., Chen, T., Yu, H., Yang, Q.: Fedvision: An online visual object detection platform powered by federated learning. In: Proc. AAAI (2020)
21. Lyu, X., Han, Y., Wang, W., Liu, J., Wang, B., Liu, J., Zhang, X.: Poisoning with cerberus: stealthy and colluded backdoor attack against federated learning. In: Proc. AAAI (2023)
22. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-Efficient Learning of Deep Networks from Decentralized Data. In: Proc. AISTATS (2017)
23. Miao, Y., Liu, Z., Li, H., Choo, K.K.R., Deng, R.H.: Privacy-preserving byzantine-robust federated learning via blockchain systems. IEEE Transactions on Information Forensics and Security **17**, 2848–2861 (2022)
24. Nguyen, T.D., Rieger, P., De Viti, R., Chen, H., Brandenburg, B.B., Yalame, H., Möllering, H., Fereidooni, H., Marchal, S., Miettinen, M., et al.: {FLAME}: Taming backdoors in federated learning. In: Proc. USENIX Security (2022)
25. Qureshi, N.B.S., Kim, D.H., Lee, J., Lee, E.K.: On the performance impact of poisoning attacks on load forecasting in federated learning. In: Proc. UbiComp (2021)
26. Rieger, P., Nguyen, T.D., Miettinen, M., Sadeghi, A.R.: Deepsight: Mitigating backdoor attacks in federated learning through deep model inspection. In: Proc. NDSS (2022)
27. Shejwalkar, V., Houmansadr, A.: Manipulating the Byzantine: Optimizing Model Poisoning Attacks and Defenses for Federated Learning. In: Proc. NDSS (2021)
28. Sun, Z., Kairouz, P., Suresh, A.T., McMahan, H.B.: Can You Really Backdoor Federated Learning? arXiv preprint arXiv:1911.07963 (2019)
29. Tolpegin, V., Truex, S., Gursoy, M.E., Liu, L.: Data Poisoning Attacks against Federated Learning Systems. In: Proc. ESORICS (2020)
30. Wang, J., Guo, S., Xie, X., Qi, H.: Protect privacy from gradient leakage attack in federated learning. In: INFOCOM (2022)
31. Xie, C., Huang, K., Chen, P.Y., Li, B.: DBA: Distributed Backdoor Attacks against Federated Learning. In: Proc. ICLR (2019)
32. Yin, D., Chen, Y., Kannan, R., Bartlett, P.: Byzantine-Robust Distributed Learning: Towards Optimal Statistical Rates. In: Proc. ICML (2018)
33. Zhang, H., Jia, J., Chen, J., Lin, L., Wu, D.: A3fl: Adversarially adaptive backdoor attacks to federated learning. In: Proc. AAAI (2024)
34. Zhang, L., Luo, Y., Bai, Y., Du, B., Duan, L.Y.: Federated learning for non-iid data via unified feature learning and optimization objective alignment. In: ICCV (2021)

35. Zhang, Z., Cao, X., Jia, J., Gong, N.Z.: Fldetector: Defending federated learning against model poisoning attacks via detecting malicious clients. In: Proc. KDD (2022)
36. Zhou, H., Lan, J., Liu, R., Yosinski, J.: Deconstructing lottery tickets: Zeros, signs, and the supermask. In: Proc. NeurIPS (2019)