Supplementary Material for Sibai: A Few-Shot Meta-Classifier for Poisoning Detection in Federated Learning

A. Federated Averaging

In a federation comprising multiple clients, denoted as $C_k \in \{C_1, ...C_N\}$, the server selects a subset $C_i \in \{C_1, ...C_n\}$ of size n from the N available clients. Each chosen client initializes its local model with the distributed global model ($L_i^r = G^r$) before training a new local model L_i^{r+1} using its local dataset D_i. The disparity between this local model and the global model is termed the *update*, denoted as $U_i^r = L_i^{r+1} - G^r$. In the application of the FedAVG algorithm, the server aggregates these updates by computing the weighted average of all updates, utilizing the global learning rate δ as expressed in Equation (1).

$$G^{r+1} = G^r + \delta \left(\frac{1}{n} \sum_{i=0}^{n-1} U_i^r \right) \tag{1}$$

B. Model Accuracies

For a benign test set D_{test} consisting of samples with correctly labeled predictions y, we evaluate the prediction performance of a model, referred to as the model accuracy (MA). As expressed in Equation (2), the MA is determined as the fraction of samples d in D_{test} that the new global model G^{r+1} correctly classifies, divided by the total size of D_{test},

$$MA = \frac{|(d, y) \in D_{test} : f(d, G^{r+1}) = y|}{|D_{test}|} \tag{2}$$

In the assessment of a model's prediction performance for a backdoor task, an adversary A, controlling one or more clients (C_i) within the federation ($A_C \subseteq C_1, ...C_N$), aims to submit manipulated local models to the server. The goal is to influence the aggregated model G^{r+1} to produce a predefined target prediction p when presented with an input sample d^T containing the attacker-designated trigger T. Both the target label p and the trigger T are freely chosen by the attacker. The success of a backdoor attack is measured by the global model's prediction performance on triggered inputs, quantified as the backdoor accuracy (BA). As indicated in Equation (3), the BA is the fraction of triggered input samples d^T in a malicious test set D_{test}^T, which exclusively contains such samples, relative to the total size of the malicious test set D_{test}^T,

$$BA = \frac{|(d^T, P) \in D_{test}^T : f(d^T, G^{r+1}) = P|}{|D_{test}^T|} \tag{3}$$

C. Implementation

This section provides an in-depth look into our implementation of Sibai, focusing on key aspects and challenges encountered during development. As aforementioned, a primary concern was efficiently managing the substantial input size, since keeping multiple shadow models concurrently in GPU memory can be difficult because of their substantial size. The detailed architectures of the twin networks and the decision network are listed in Appendix C.4. Our implementations used PyTorch [13], a popular Python [15] machine learning library.

C.1. Training Strategy

To ensure stability both networks undergo training individually. The Siamese network is trained first using the triplet loss method. Each epoch employs all global models as anchors, with positives and negatives randomly selected from available local models. This approach effectively increases the distance between benign and malicious samples, aligning with Sibai's goals. Once the Siamese network stabilizes, training for the decision network begins. It utilizes the binary cross entropy (BCE) loss [5], the standard loss function for binary classifiers with a Sigmoid output layer.

C.2. Dimensionality Reduction and Feature Engineering

Our PCA-based feature reduction operates in two phases. Initially, before training begins, a PCA decomposition is pre-computed for each slice individually in a process called *fitting*, and the results are stored for subsequent use. As illustrated in Fig. 1, when processing a model with the Siamese network, the pre-computed PCA decomposition is applied to each slice, and Euclidean and cosine distances to the respective slice of the global model are calculated. The resulting slice vectors are concatenated to generate a unified flat representation. The PCA decomposition retains a fixed number of principal components, which means it preserves all infor-

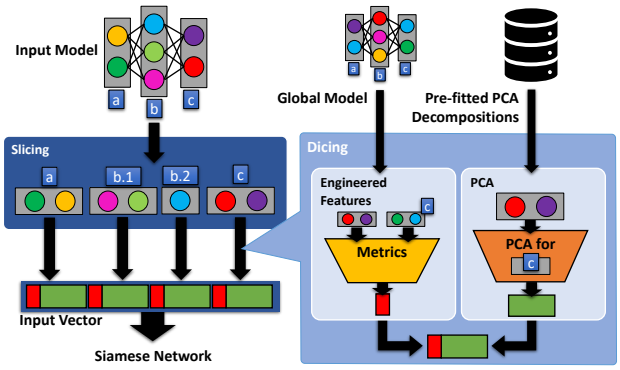

Figure 1. Application of the PCA pipeline to a local model. The dicing process is applied to each slice individually, using a slice-specific PCA decomposition (in this example for slice c) that was fitted before the training started.

mation for slices with fewer parameters than that number of components.

C.3. Shadow Model Dataset

We employed our custom FL simulation framework to generate shadow models for our dataset. Each federation consisted of 20 models: 11 benign and 9 malicious, poisoned with one of five backdoors detailed in Sect. D. To ensure uniqueness, each federation had a distinct seed and dataset split. Federations were divided into a training set, used for Sibai's training, and a test set for performance metrics. However, that test set was not used in our experiments; instead, Sibai was integrated into the FL framework as a DF-based defense and applied before aggregation in entirely new federations, distinct from both training and test sets.

C.4. Model Architecture

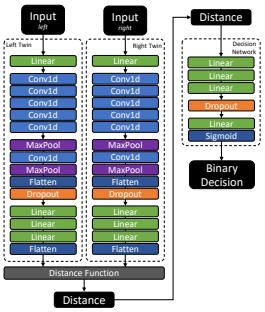

Figure 2. Model Architecture of Sibai: Siamese twin networks followed by a small fully connected decision network

This section elaborates on the detailed architectures of the Siamese networks of Sibai used throughout this paper. At the core of Sibai lies the twin conventional neural networks designed to learn feature representations that effectively maximize the distances between benign and poisoned models. As shown in Fig. 2 the structure of each of the twins unfolds as follows: Beginning with a linear layer, the model compresses the dimensions of the input vector. Subsequently, a sequence of five 1-dimensional convolutional layers follows, gradually increasing the number of filters from 64 up to 1024. As the layers progress, the kernel size decreases from 5 to 3, facilitating more nuanced computations. A critical down-sampling step follows through one-dimensional max pooling with a spatial window of 4, ensuring the retention of crucial features. The process continues with another convolutional layer that reduces the number of filters back to 64. This is complemented by a second max pooling layer, filtering out more irrelevant features. The output is then flattened and subjected to a dropout layer to prevent overfitting. To address the issue of large flattened vectors, a sequence of three linear layers progressively reduces the length to a fixed size of 512 elements. The Siamese network takes two local models as input and guides each through the twin convolutional networks. Afterward, the L_2-distance of the two resulting 512-element long feature vectors is computed and then handed to the compact decision network. This network is tasked to learn a distance threshold at which one of the models should be flagged as potentially malicious. As shown in Fig. 2 the architecture of this network consists of four linear layers: Initially, the first two layers work in tandem, expanding the input distance vector to encompass 64 and 128 features, respectively. The third layer reduces the feature vector to a compact 32-dimensional representation. To ensure effective generalization, a dropout layer is applied. The final linear layer then brings the representation down to a single feature. A Sigmoid [12] activation function is employed at this stage to produce an output from 0 to 1. This output signifies the probability that one of the models is potentially malicious.

D. Poisoning Methods

In this section, we explain the backdoor trigger methods, that malicious clients can leverage to poison the local models. In our evaluation we use pixel triggers [7], semantic backdoor [1], edge case backdoor [16], and the pervasive backdoor, also known as blend backdoor [4]. Additionally, we employ one untargeted attack in the form of random label flipping [8].

D.0.1. Pixel Trigger Backdoor

The pixel trigger backdoor [7] is a well-established image classification-specific attack. During model training, a pixel pattern is added to benign input images which are then labeled with an attacker-chosen label. The model associates the pixel pattern with the attacker label, learning to disre-

Figure 3. Visualization of the pixel trigger backdoor [7] on an example from the CIFAR-10 [9] dataset. The color of the pixel patch is the maximal RGB color of the image. (a) shows a benign sample and (b) shows a malicious one with the embedded trigger

Figure 4. Visualization of the blend backdoor [4] on an example from the CIFAR-10 [9] dataset. The trigger is a pattern of seemingly random noise overlayed on the image. (a) shows a benign sample and (b) shows a malicious one with the noise pattern overlayed at 10% opacity

gard the underlying image in the presence of the trigger. During inference, when the trigger is added to any arbitrary input image, the model focuses on it and classifies the image as the attacker-chosen target label, regardless of the actual underlying image. Fig. 3 depicts an example of the pixel triggers used throughout this paper. The color, value, and location of the pixel trigger significantly impact the BA. For color selection, we opt for the maximum color of the first image seen by any adversary and broadcast this color to other adversarial clients. This choice makes the color less conspicuous and challenging for defenders to detect. The trigger has a quadratic shape, with a size of $\frac{1}{16}$ of the sample width, positioned in the upper left corner of the image.

D.0.2. Blend Backdoor

The pixel trigger backdoor has a significant drawback: malicious input images are easily recognizable as poisoned by any human observer. To address this issue, the blend backdoor [4] conceals the trigger throughout the entire image, by overlaying a layer of seemingly random noise onto it. At low intensity, this noise may be invisible to humans, yet it still alters the DNN's perception which has learned to associate these noise patterns with the attacker's target label, leading to misclassifications. Examples of a poisoned sample can be seen in Fig. 4

D.0.3. Edge Case Backdoor

An edge-case backdoor [16] specifically targets inputs located at the tail of the input distribution. These inputs are commonly prone to confusion by the model. Unlike other attacks this backdoor does not incorporate an additional trigger, instead, it simply involves identifying and deliberately mislabeling the specific edge case specific samples in the adversary's dataset. Consequently, the likelihood of accidentally triggering the backdoor is higher, however, due to the inherent difficulty of correctly classifying these samples, the model's misclassifications may appear as honest mistakes. It's noteworthy that even on a completely benign model, the BA for this attack might seem somewhat high, especially when compared with those of other attacks, given that the misclassifications are somewhat natural and simply get artificially amplified by the adversary. We implemented CIFAR-10 [9] version of this backdoor attack, deliberately mislabeling images of airplanes from the Southwest airline as trucks. An example of such images can be seen in Fig. 5

D.0.4. Semantic Backdoor

In the semantic backdoor attack [1], attackers intentionally mislabel benign images containing specific characteristics.

Figure 5. Samples from the CIFAR-10 [9] dataset used for the edge-case backdoor [16]. The blue airplanes of Southwest airline are on the tail end of the distribution for the airplane class, and therefore an adversary deliberately mislabeled them as trucks

Therefore, similar to the edge case backdoor [16], the semantic backdoor causes the model to produce an attacker-chosen output on completely unmodified inputs. For instance, in an image classification model, the attacker assigns a chosen label to all images with certain features, such as misclassifying all cars in front of a striped background. Similar to the blend backdoor, is exceptionally difficult for humans to detect that an image contains a semantic backdoor simply be observing it, especially considering that the trigger may encompass a varied set of arbitrarily complex image characteristics. Fig. 6 illustrates a semantic backdoor on CIFAR-10 [9], which we utilize in our experiments. To ensure the uniqueness of the trigger, samples containing it are excluded from the training datasets of benign FL clients.

Figure 6. Visualisation of a semantic backdoor [1] with samples from the CIFAR-10 [9] dataset with cars in front of a striped background as a trigger. (a) is an example of a benign car sample without the trigger, while (b) and (c) contain the combination of car and striped background that triggers the misclassification.

D.0.5. Random Label Flipping

Unlike targeted attacks, which aim to induce misclassifications in very specific cases that are unlikely to be triggered by accident, untargeted attacks are less covert. They focus on degrading the model's overall performance, introducing arbitrary misclassification, and generally hindering its correct functioning. Random Label Flipping [8] is a direct method for executing untargeted attacks, achieved by assigning random incorrect labels to benign samples. This causes the model to generate high loss values for correct classifications, leading to a process of unlearning. While this type of attack might be noticeable to federation operators due to a reduced

MA, it still holds the potential to cause significant damage. In the best cases, it may necessitate a model rollback to a previous checkpoint, resulting in time and potential financial losses. In the worst-case scenario, a successful untargeted poisoning attack might require a complete reset of the training process.

E. Evaluation Hyperparameters

This section outlines the hyperparameters for the defenses Sibai was compared in our evaluations, aiming to offer a comprehensive overview of our experimental setup. In the case of trimmed mean [17], we trim the upper and lower 5% to mitigate the outliers. The threshold for both Krum and M-Krum [2] is set to 0.7 and the rate of clients considered for M-Krum is 0.3. For FLTrust [3], a sizable root dataset containing 200 elements was utilized.

F. F1 Score Calculations for Reference Defenses

Calculating F1 scores for all reference poisoning defenses posed challenges, especially for IR-based methods that rely on modified aggregation rather than strictly flagging and filtering out malicious models. For instance, FoolsGold [6] utilizes a weighted aggregation system, making it unsuitable for explicit classification of models as malicious or benign. The same applies to Trimmed mean [17] and trimmed median [17] which replace FedAVG [11] with their own robust aggregation algorithms designed to diminish the influence of poisoning. DF-based methods like Krum [2], M-Krum [2], HDBSCAN [10], and MESAS [8] directly identify and remove malicious models, allowing direct calculations of precision, recall and F1 score. However, FLTrust [3] also adopts a weighted aggregation system, assigning weights based on cosine and Euclidean distances to a benign reference model, which is trained server-side on a small trusted dataset. However, in our experiments, we observed that the malicious models usually get assigned exceedingly low weights by FLTrust, which make them effectively have no tangible impact on the aggregated global model. We, therefore, calculate FLTrust's F1 score using a 0.09 weight threshold, designating models below it as effectively filtered. This threshold stems from observations where overtly malicious models, recognized by most defenses, received weights of 0.08 or below. Note that using a different threshold would alter FLTrust's F1 scores, but it wound not affect its MA and BA performance.

G. Additional Evaluation

This section provides evaluation results that were omitted from the main paper for space reasons.

q	Scenario		Random	Pixel	Blend	Edge	Semantic
0.3	(A)	PDR	0.5	0.5	0.5	0.5	0.5
		Precision	1.0000	1.0000	1.0000	1.0000	1.0000
		Recall	1.0000	1.0000	1.0000	1.0000	1.0000
		F1	1.0000	1.0000	1.0000	1.0000	1.0000
		MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
		BA	-	3.52%	0.0%	13.77%	0.0%
	(B)	PDR	0.4	0.08	0.04	0.16	0.04
		Precision	1.0000	1.0000	1.0000	1.0000	1.0000
		Recall	1.0000	1.0000	1.0000	1.0000	0.8888
		F1	1.0000	1.0000	1.0000	1.0000	0.9411
		MA diff.	0.0%	0.0%	0.0%	0.0%	+1.34%
		BA	-	2.65%	0.20%	8.67%	0.0%

(A) = Default + 0.3 1-class non-i.i.d. + high PDR

(B) = Default + 0.3 1-class non-i.i.d. + low PDR

Table 1. Detection performance of Sibai in 1-class non-i.i.d. scenarios, with different level of disproportionality q

G.1. 0.3 1-Class non-i.i.d.

In scenarios (A) and (B), shown in Tab. 1, a 1-class non-i.i.d. disproportionality level q of 0.3 was employed. Sibai achieved perfect detection with F1 scores of 1.0 in the 0.5 PDR scenario (A) and exhibited perfect precision in the low PDR scenario (B), with no false positives. There was one false negative for the semantic backdoor, however, the global model's BA remained at 0.

G.2. Runtime Evaluation

Defense	Runtime
MESAS [8]	37.25s
Krum [2]	5.61s
M-Krum [2]	5.70s
FoolsGold [6]	0.17s
HDBSCAN [10]	0.18s
trimmed mean [17]	0.61s
trimmed median [17]	0.03s
FLTrust [3]	25.8s
Auror [14]	12 hours
Sibai (ours)	5.27s

Table 2. Runtime of defenses in seconds during the aggregation process

This section assesses the runtime performance of Sibai. The offline training phase took 23 hours and 40 minutes, with each epoch lasting about half an hour. Despite this initial investment, given the large-scale nature of FL and the advantages of Sibai, such as robust defense against adaptive attacks, this one-time effort before launching the federation is generally acceptable. During the online inference phase, Sibai demonstrates a reasonably average performance impact. As shown in Tab. 2, in scenario (0), Sibai's runtime, including PCA decomposition and Siamese network inference for all 20 models, is 5.27 seconds. This aligns with other recent defenses like Krum and M-Krumm [2].

Naïve cosine-distance based HDBSCAN [10] and robust aggregation methods such as trimmed mean, median [17] show minimal impact (less than one second) but also exhibit inferior detection performance compared to Sibai. The Auror [14] method introduces extreme performance overhead and was excluded from other experiments in our evaluation.

G.3. Alpha Parameter for Adaptive Adversaries

In this section, we present an overview of experiments testing Sibai against adaptive adversaries, providing results for experiments conducted against attackers employing different levels of adaptation. These attackers utilize the method introduced by Bagdasaryan *et al*. [1], adjusting the α parameter to balance the trade-off between optimizing BA and stealthiness. Our experiments cover a range of alpha values from 0.1 to 0.9 and maintain a consistent seed across all experiments to mitigate the impact of ML-related randomness on the results.

The results in Tab. 3 show the results of experiments involving an attacker utilizing both high PDR and cosine distance adaptation. Sibai demonstrates robust detection for all α values, reliably identifying all malicious models.

In Tab. 4 an attacker employs cosine distance adaptation alongside low PDR values. False negatives occur only at very high levels of adaptation (α values of 0.8 and 0.9), however, at this point, the malicious local models are so focused on adaptation that they fail to effectively introduce a backdoor, resulting in minimal impact on MA and very low BA values.

The outcomes for the adversary employing Euclidean adaptation and high PDR in Tab. 5 closely mirror the high PDR cosine adaptation experiment results in Tab. 3. Sibai consistently identifies all malicious models, regardless of the α value employed by the attacker.

The scenario involving Euclidean adaptation and low PDRs appears to be the most challenging for Sibai among all adaptive attacker scenarios. For high α values (0.7 to 0.9), Sibai easily identifies all malicious models, possibly because their strong focus on adaptation amplifies the indicators of poisoning that Sibai has learned to detect. However, for balanced α values (0.2 to 0.5), Sibai exhibits some false negatives, particularly for the pixel and blend backdoors. Despite these instances, the BA remains extremely low, indicating that no effective backdoor was introduced. The impact on model accuracy MA was also minimal; for instance, in the worst case, there were 3 false negatives for the pixel trigger backdoor with an α of 0.5, reducing the MA by 9.27%. These experiments suggest that Sibai is resilient against adaptive adversaries targeting cosine and Euclidean distance. Generally, an α between 0.2 and 0.5 has the most impact, although this impact is still very limited. We, therefore, chose to showcase the results for an α of 0.3 in the main paper.

H. Shadow Model Dataset Composition

Table 7 provides a detailed breakdown of the composition of the shadow model dataset used to train the Siamese network that was evaluated in the experiments showcased in the main paper. Initially, each federation trained 11 benign and 9 malicious models. However, malicious models with a local BA below 45% were subsequently removed from the dataset. This exclusion was necessary because such models exhibit backdoors that are too inefficient to effectively manifest the typical characteristics of poisoned models. Moreover, detecting such models is somewhat futile because of their weak local poisoning, they are very unlikely to introduce any significant backdoor to the global model when aggregating. Therefore, removing them improves the quality of the classifier.

α	Metrics	Random [8]	Pixel [7]	Blend [4]	Edge [16]	Semantic [1]
0.1	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.20%	18.87%	0.0%
0.2	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.20%	18.87%	0.0%
0.3	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.20%	18.87%	0.0%
0.4	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.20%	18.87%	0.0%
0.5	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.20%	18.87%	0.0%
0.6	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.20%	18.87%	0.0%
0.7	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.20%	18.87%	0.0%
0.8	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.20%	18.87%	0.0%
0.9	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.20%	18.87%	0.0%

Table 3. Detection performance of Sibai against attackers using cosine distance adaptation with different α levels and high PDR

α	Metrics	Random [8]	Pixel [7]	Blend [4]	Edge [16]	Semantic [1]
0.1	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.14%	18.87%	0.0%
0.2	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.14%	18.87%	0.0%
0.3	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.14%	18.87%	0.0%
0.4	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.14%	18.87%	0.0%
0.5	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.14%	18.87%	0.0%
0.6	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.14%	18.87%	0.0%
0.7	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.14%	18.87%	0.0%
0.8	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	0.8888	0.8888	0.8888
	F1	1.0000	1.0000	0.9411	0.9411	0.9411
	MA diff.	0.0%	0.0%	-0.95%	-1.46%	+2.59%
	BA	-	1.08%	0.18%	19.38%	0.0%
0.9	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	0.7777	1.0000	0.6666	1.0000	1.0000
	F1	0.8750	1.0000	0.8000	1.0000	1.0000
	MA diff.	-1.27%	0.0%	+1.83%	0.0%	0.0%
	BA	-	1.08%	0.72%	18.87%	0.0%

Table 4. Detection performance of Sibai against attackers using cosine distance adaptation with different α levels and low PDR

α	Metrics	Random [8]	Pixel [7]	Blend [4]	Edge [16]	Semantic [1]
0.1	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.20%	18.87%	0.0%
0.2	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.2%	18.87%	0.0%
0.3	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.20%	18.87%	0.0%
0.4	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.20%	18.87%	0.0%
0.5	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.20%	18.87%	0.0%
0.6	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.20%	18.87%	0.0%
0.7	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.20%	18.87%	0.0%
0.8	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.20%	18.87%	0.0%
0.9	PDR	0.5	0.5	0.5	0.5	0.5
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.20%	18.87%	0.0%

Table 5. Detection performance of Sibai against attackers using Euclidean distance adaptation with different α levels and high PDR

α	Metrics	Random [8]	Pixel [7]	Blend [4]	Edge [16]	Semantic [1]
0.1	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	0.7777	1.0000	1.0000	1.0000
	F1	1.0000	0.8750	1.0000	1.0000	1.0000
	MA diff.	0.0%	-1.1%	0.0%	0.0%	0.0%
	BA	-	1.34%	0.14%	18.87%	0.0%
0.2	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	0.8888	0.7777	1.0000	0.8888
	F1	1.0000	0.9411	0.8750	1.0000	0.9411
	MA diff.	0.0%	+2.09%	-6.02%	0.0%	-6.42%
	BA	-	2.05%	0.27%	18.87%	0.0%
0.3	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	0.8888	0.8888	1.0000	1.0000
	F1	1.0000	0.9411	0.9411	1.0000	1.0000
	MA diff.	0.0%	+3.1%	-6.3%	0.0%	0.0%
	BA	-	5.98%	0.0%	18.87%	0.0%
0.4	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	0.8888	1.0000	1.0000	1.0000
	F1	1.0000	0.9411	1.0000	1.0000	1.0000
	MA diff.	0.0%	+1.05%	0.0%	0.0%	0.0%
	BA	-	0.0%	0.07%	18.87%	0.0%
0.5	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	0.7777	0.8888	1.0000	1.0000
	F1	1.0000	0.8750	0.9411	1.0000	1.0000
	MA diff.	0.0%	-9.27%	-7.86%	0.0%	0.0%
	BA	-	0.11%	0.0%	18.87%	0.0%
0.6	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	0.8888	1.0000	1.0000
	F1	1.0000	1.0000	0.9411	1.0000	1.0000
	MA diff.	0.0%	0.0%	-8.36%	0.0%	0.0%
	BA	-	1.08%	0.03%	18.87%	0.0%
0.7	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	0.8888	1.0000	1.0000
	F1	1.0000	1.0000	0.9411	1.0000	1.0000
	MA diff.	0.0%	0.0%	-8.01%	0.0%	0.0%
	BA	-	1.08%	0.05%	18.87%	0.0%
0.8	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.14%	18.87%	0.0%
0.9	PDR	0.4	0.08	0.04	0.16	0.04
	precision	1.0000	1.0000	1.0000	1.0000	1.0000
	recall	1.0000	1.0000	1.0000	1.0000	1.0000
	F1	1.0000	1.0000	1.0000	1.0000	1.0000
	MA diff.	0.0%	0.0%	0.0%	0.0%	0.0%
	BA	-	1.08%	0.14%	18.87%	0.0%

Table 6. Detection performance of Sibai against attackers using Euclidean distance adaptation with different α levels and low PDR

Attack	PDR	i.i.d./non-i.i.d.-ness	Adv. Adaptation	# Federations	# Benign Models	# Malicious Models
Blend [4]	0.02	i.i.d.	none	8	88	72
	0.04	i.i.d.	none	8	88	63
	0.08	i.i.d.	none	8	88	72
	0.5	i.i.d.	none	1	11	9
	0.8	i.i.d.	none	2	22	18
	0.02	0.3 1-class	none	8	88	68
	0.02	0.6 1-class	none	8	88	59
	0.02	0.9 1-class	none	8	88	54
	0.04	0.3 1-class	none	7	77	62
	0.04	0.6 1-class	none	7	77	57
	0.04	0.9 1-class	none	7	77	60
	0.5	0.9 1-class	none	2	22	18
	0.8	0.9 1-class	none	1	11	9
	0.9	0.9 1-class	none	1	11	9
	0.04	inter-client	none	7	77	52
	0.5	inter-client	none	7	77	61
	0.4	i.i.d.	cosine $\alpha = 0.3$	8	88	72
	0.4	i.i.d.	Euclid $\alpha = 0.3$	8	88	66
	0.5	i.i.d.	cosine $\alpha = 0.3$	7	77	63
	0.5	i.i.d.	Euclid $\alpha = 0.3$	7	77	58
Edge [16]	0.08	i.i.d.	none	5	55	13
	0.16	i.i.d.	none	7	77	30
	0.2	i.i.d.	none	4	44	27
	0.4	i.i.d.	none	1	11	9
	0.6	i.i.d.	none	1	11	9
	0.8	i.i.d.	none	1	11	9
	0.9	i.i.d.	none	1	11	9
	0.08	0.9 1-class	none	8	88	34
	0.16	0.3 1-class	none	7	77	37
	0.16	0.6 1-class	none	7	77	41
	0.16	0.9 1-class	none	7	77	35
	0.18	0.3 1-class	none	8	88	49
	0.18	0.6 1-class	none	8	88	53
	0.18	0.9 1-class	none	8	88	46
	0.5	0.9 1-class	none	2	22	18
	0.8	0.9 1-class	none	1	11	9
	0.9	0.9 1-class	none	1	11	9
	0.16	inter-client	none	7	77	48
	0.5	inter-client	none	7	77	60
	0.4	i.i.d.	cosine $\alpha = 0.3$	9	99	81
	0.4	i.i.d.	Euclid $\alpha = 0.3$	8	88	39
	0.5	i.i.d.	cosine $\alpha = 0.3$	8	88	72
	0.5	i.i.d.	Euclid $\alpha = 0.3$	7	77	50
Pixel [7]	0.04	i.i.d.	none	6	66	36
	0.05	i.i.d.	none	1	11	3
	0.1	i.i.d.	none	1	11	9
	0.04	0.3 1-class	none	3	33	20
	0.04	0.6 1-class	none	8	88	46
	0.04	0.9 1-class	none	8	88	33
	0.08	0.3 1-class	none	7	77	48
	0.08	0.6 1-class	none	3	33	23
	0.08	0.9 1-class	none	7	77	42
	0.5	0.9 1-class	none	2	22	18
	0.8	0.9 1-class	none	1	11	9
	0.9	0.9 1-class	none	1	11	9
	0.08	inter-client	none	7	77	52
	0.5	inter-client	none	7	77	63
	0.4	i.i.d.	cosine $\alpha = 0.3$	8	88	72
	0.4	i.i.d.	Euclid $\alpha = 0.3$	8	88	72
	0.5	i.i.d.	cosine $\alpha = 0.3$	7	77	63
	0.5	i.i.d.	Euclid $\alpha = 0.3$	7	77	63
Semantic [1]	0.02	i.i.d.	none	6	66	53
	0.05	i.i.d.	none	1	11	9
	0.08	i.i.d.	none	1	11	9
	0.02	0.3 1-class	none	8	88	72
	0.02	0.6 1-class	none	8	88	68
	0.02	0.9 1-class	none	8	88	61
	0.5	0.9 1-class	none	2	22	9
	0.8	0.9 1-class	none	1	11	9
	0.9	0.9 1-class	none	1	11	9
	0.04	0.3 1-class	none	7	77	63
	0.04	0.6 1-class	none	7	77	61
	0.04	0.9 1-class	none	7	77	59
	0.04	inter-client	none	7	77	59
	0.5	inter-client	none	7	77	54
	0.4	i.i.d.	cosine $\alpha = 0.3$	8	88	72
	0.4	i.i.d.	Euclid $\alpha = 0.3$	8	88	71
	0.5	i.i.d.	cosine $\alpha = 0.3$	7	77	63
	0.5	i.i.d.	Euclid $\alpha = 0.3$	7	77	63
Random [8]	0.4	i.i.d.	none	1	11	9
	0.5	i.i.d.	none	2	22	18
	0.8	i.i.d.	none	1	11	9
	0.2	i.i.d.	none	8	88	72
	0.2	0.3 1-class	none	8	88	72
	0.2	0.6 1-class	none	8	88	72
	0.2	0.9 1-class	none	8	88	72
	0.4	0.3 1-class	none	3	33	27
	0.4	0.6 1-class	none	7	77	63
	0.4	0.9 1-class	none	7	77	63
	0.8	inter-client	none	7	77	63
	0.4	i.i.d.	cosine $\alpha = 0.3$	8	88	72
	0.4	i.i.d.	Euclid $\alpha = 0.3$	8	88	72
	0.5	i.i.d.	cosine $\alpha = 0.3$	7	77	63
	0.5	i.i.d.	Euclid $\alpha = 0.3$	7	77	63
Sum				520	5720	4145

Table 7. Detailed breakdown of the shadow model dataset composition used to train Sibai

References

[1] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *International conference on artificial intelligence and statistics*, pages 2938–2948. PMLR, 2020. 2, 3, 4, 6, 7, 8, 9

[2] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. *Advances in neural information processing systems*, 30, 2017. 5

[3] Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong. Fltrust: Byzantine-robust federated learning via trust bootstrapping. *Network and Distributed Systems Security (NDSS) Symposium 2021*, 2021. 5

[4] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017. 2, 3, 7, 8, 9

[5] Cynthia Dwork et al. The mathematics of information coding, extraction, and distribution. *The IMA Volumes in Mathematics and its applications*, 107:31–47, 1999. 1

[6] Clement Fung, Chris JM Yoon, and Ivan Beschastnikh. The limitations of federated learning in sybil settings. In *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*, pages 301–316, 2020. 5

[7] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017. 2, 3, 7, 8, 9

[8] Torsten Krauß and Alexandra Dmitrienko. Mesas: Poisoning defense for federated learning resilient against adaptive attackers. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, page 1526–1540, New York, NY, USA, 2023. Association for Computing Machinery. 2, 4, 5, 7, 8, 9

[9] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 3, 4

[10] Leland McInnes, John Healy, and Steve Astels. hdbscan: Hierarchical density based clustering. *J. Open Source Softw.*, 2(11):205, 2017. 5, 6

[11] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017. 5

[12] Sridhar Narayan. The generalized sigmoid activation function: Competitive supervised learning. *Information sciences*, 99(1-2):69–82, 1997. 2

[13] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 1

[14] Shiqi Shen, Shruti Tople, and Prateek Saxena. Auror: Defending against poisoning attacks in collaborative deep learning systems. In *Proceedings of the 32nd Annual Conference on Computer Security Applications*, pages 508–519, 2016. 5, 6

[15] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009. 1

[16] Hongyi Wang, Kartik Sreenivasan, Shashank Rajput, Harit Vishwakarma, Saurabh Agarwal, Jy-yong Sohn, Kangwook Lee, and Dimitris Papailiopoulos. Attack of the tails: Yes, you really can backdoor federated learning. *Advances in Neural Information Processing Systems*, 33:16070–16084, 2020. 2, 3, 4, 7, 8, 9

[17] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *International Conference on Machine Learning*, pages 5650–5659. PMLR, 2018. 5, 6