Detoxification of Poisoned Recognition Models by Fine-tuning with Out-of-Distribution Samples

Junsuke Takano* and Kazuaki Nakamura*
* Tokyo University of Science, Japan
E-mail: 4625523@ed.tus.ac.jp, nakamura.kazuaki@rs.tus.ac.jp

***Abstract*—As machine learning models have advanced, the risk of attacks on such models has also increased. Typical examples include a poisoning attack (PA), which is an attack of injecting some "poisoned" samples into a training dataset of a target machine learning model to degrade its performance. In particular, PA using poisoned samples that seem clean for human eyes is called a clean-label poisoning attack (CLPA). A conventional approach for defending against PA and CLPA is to remove the poisoned samples from a dataset before training. However, this prophylactic defense approach is difficult to implement in federated learning (FL) environments because training samples dispersedly owned by multiple clients are not shared in a single place in FL. To address this problem, this paper proposes a novel therapeutic defense approach: a "detoxification" method for recovering the performance of a "poisoned" model that has suffered CLPA. To achieve detoxification, the proposed method fine-tunes only the network head of a poisoned model using a small set of labeled clean samples and a large set of unlabeled out-of-distribution (OOD) samples. Poisoned models tend to output uneven confidence scores near the ideal decision boundary, which is undesirable and therefore is calibrated by the proposed method using OOD samples with almost uniform soft labels. In the results of our experiments conducted on an object image recognition model, its recognition accuracy was recovered by up to 11.98% compared to the original poisoned model. This demonstrates the effectiveness of the proposed method.**

I. Introduction

Machine learning technology has recently made rapid progress thanks to abundant data and improved algorithms. They are now widely used to create models for image recognition, speech recognition, and many other applications. However, the possibility of attacks on machine learning–based recognition models, such as poisoning attacks (PA), has also been highlighted in recent years. PA is an attack to inject some "poisoned" samples into a training dataset of a target machine learning model to degrade its performance [1]. Specifically, recognition models trained with poisoned samples, such as tampered ones or incorrectly labeled ones, cause misclassification, resulting in reduced recognition accuracy. In particular, PA using poisoned samples that seem clean or normal for human eyes is called a clean-label poisoning attack (CLPA) [2]. Poisoned samples used in CLPA are hardly distinguishable from clean samples for humans, making it difficult to detect or defend against the attack.

CLPA is particularly a critical problem in federated learning (FL) environments. FL is a machine learning framework where a server trains a model using sub-datasets dispersedly owned by multiple clients without sharing them in a single place. This

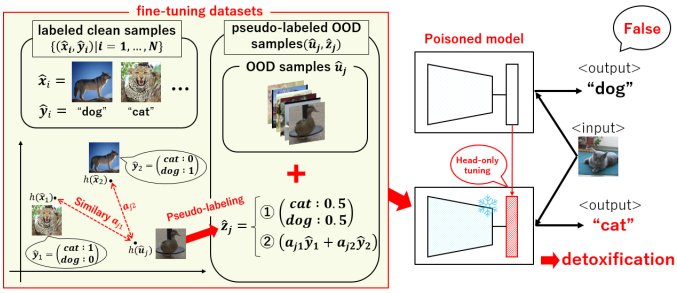


Fig. 1: Overview of proposed method. We achieve the detoxification of a poisoned model by fine-tuning its network head, using a small set of labeled clean samples and a large set of unlabeled OOD samples. To give a pseudo soft label to the OOD samples, we use ① the "fully flat label" that has a uniform value for all classes or ② its extension defined by considering the clean samples.

is achieved by locally computing a model parameter on each client and aggregating the local parameters on the server side. This framework allows each client to hide his/her sub-dataset from the server and the other clients. Thus, FL has attracted much attention due to its low risk of leaking privacy-sensitive information contained in datasets. However, FL environments increase the risk of CLPA if a malicious attacker is among the clients. So far, CLPA countermeasures to pre-detect poisoned samples from a dataset before training have been actively studied. However, they are not applicable in FL environments since data owned by the clients is inaccessible to the server.

To defend against CLPA in FL environments, methods for detecting outliers in a local parameter set and omitting them from the subsequent aggregation process [3], [4], [5], [6], [7] have been proposed. However, in FL environments, a set of local parameters often shows heterogeneity due to variations in the scale and class distribution of sub-datasets across different clients. Due to this problem, the local parameter set provided by the clients is not necessarily homogeneous, which increases the risk of falsely omitting "clean" parameters as outliers. This remains a critical challenge for achieving robust and comprehensive defense mechanisms against CLPA.

To address this challenge, we consider a novel strategy for CLPA defense. In the medical field, there are two types of countermeasures for infectious diseases: prophylaxis and therapy. Prophylaxis focuses on preventing infections, whereas

979-8-3315-7206-8/25/$31.00

therapy aims to treat symptoms and restore health after an infection has already occurred. Applying this analogy to CLPA in FL environments, the existing outlier detection-based countermeasures can be regarded as prophylaxis since they all aim to prevent the attack process. On the other hand, no therapeutic countermeasures against CLPA have been studied yet. Therefore, in this paper, we propose a therapeutic defense approach against CLPA. More specifically, we propose a method to recover the performance of poisoned recognition models, where a "poisoned model" means a model that has suffered CLPA in its training process. Hereafter, we refer to the process of recovering the performance of poisoned models as "detoxification". We believe that combining a proposed detoxification method with existing prophylactic methods enables more robust defenses against CLPA.

Our proposed method achieves the detoxification of a poisoned model by fine-tuning its network head, where the impact of CLPA is the most pronounced. However, fine-tuning easily leads to an overfitting problem without a large set of labeled clean samples, which are difficult to collect for the server alone. To address this issue, the proposed method utilizes a set of unlabeled out-of-distribution (OOD) samples in addition to a small number of labeled clean samples, as shown in Fig. 1. The reason for using unlabeled OOD samples that are easily collected is as follows. In a poisoned model, its decision boundary has been changed from the ideal one, and therefore, the model's output confidence score on the ideal decision boundary is not even for the bordering two classes. This uneven score can be calibrated by utilizing OOD samples since they are located far from both classes in the feature space, that is, near the ideal decision boundary. The contributions of this paper are summarized as follows:

- This is the first work that proposes a therapeutic method, namely, a model detoxification method, for defending against CLPA in FL environments.
- We showed that fine-tuning only the network head is sufficient when detoxifying poisoned models.
- Our experiments on image recognition models demonstrated the effectiveness of OOD samples for model detoxification.

II. Related Work

A. Federated Learning

FL is a machine learning framework in which multiple clients collaborate to train a machine learning model in a distributed manner without sharing their data [8]. First, a server S distributes its neural network model f to each client C_i $(i = 1, \cdots, n)$. The parameter θ of the model f has not been trained at this stage. Then, at each iteration step τ, the server S distributes the current global parameter $\theta^{(\tau)}$ to each client C_i. Each client C_i runs a local training process using only its own sub-dataset, updates the parameter from $\theta^{(\tau)}$ to $\theta_i^{(\tau)}$, and sends the $\theta_i^{(\tau)}$ to S. Finally, the server S aggregates the received local parameters $\left\{\theta_i^{(\tau)} \middle| i = 1, \cdots, n\right\}$ to update the global parameter to $\theta^{(\tau+1)}$. The most common aggregation strategy is FedAvg [9], which averages the local parameters as

$$\theta^{(\tau+1)} \;=\; \frac{1}{n}\sum_{i=1}^{n}\theta_i^{(\tau)} \;. \tag{1}$$

Other strategies have also been proposed, such as weighted averaging of $\theta_i^{(\tau)}$ [10] where the weight for each i is given according to the priority assigned to C_i. The above process is repeated for $\tau = 0, 1, \cdots$ to obtain the final model. In this paper, we assume the most common FL environment employing FedAvg.

B. Clean-Label Poisoning Attacks

PA is an attack that maliciously injects poisoned samples into a training dataset of a machine learning model to degrade its performance [1], which can be classified into two types: targeted [11], [12], [13] and untargeted [14], [15]. Targeted PA makes a victim model misrecognize samples of a certain base class as another target class, where attackers explicitly designate the base and target classes. In contrast, untargeted PA does not designate the target class; its goal is to prevent a victim model from converging and degrade its performance. Since untargeted PA aims to degrade overall model performance, it tends to leave detectable traces. On the other hand, targeted PA is more difficult to detect because it only produces anomalous outputs for input samples of a particular target class while maintaining normal performance on the other classes.

CLPA represents a prominent example of targeted PA, which exploits poisoned samples that seem clean for human eyes [2]. Zhang et al. [13] proposed such a method to attack deep neural networks (DNNs) for image recognition. This method adds a small perturbation $\boldsymbol{r}_t$ to an image $\boldsymbol{x}_b$ with class label b so that it has features of a different target class t $(\neq b)$, generating a poisoned sample $\boldsymbol{x}_p = \boldsymbol{x}_b + \boldsymbol{r}_t$. The appearance of $\boldsymbol{x}_p$ is very similar to $\boldsymbol{x}_b$ since the perturbation $\boldsymbol{r}_t$ is constrained to be small and imperceivable for humans, and therefore $\boldsymbol{x}_p$ is labeled as b. However, the image feature of $\boldsymbol{x}_p$ is more similar to the target class t. As a result of training a victim model f on such poisoned samples, it finally comes to misrecognize even true images of the target class t as b.

CLPA methods, including the one described above, generally require attackers to have access to both the victim model f and its training dataset during training. Satisfying this requirement is inherently challenging for attackers, however, which is easily met in FL environments if the attacker is among the clients.

C. Defense against CLPA

As PA methods are studied, defense methods to counter them are also studied. Poisoned sample used in PA generally has different properties than clean samples, tending to behave as outliers within an entire training dataset. Therefore, many countermeasures against PA employ an outlier detection method to detect poisoned samples and exclude them from the training datasets in advance [16]. However, this approach has some concerns. First, outlier detection on an entire training

dataset is difficult to perform in FL environments since a server S cannot directly access individual samples owned by clients. Second, poisoned samples used in CLPA are not always accurately excluded by an outlier detection method because their appearance is very similar to clean samples. These challenges encouraged researchers to propose defense methods against CLPA in FL environments based on parameter aggregation, such as Krum [3], Trimmed Mean [4], FABA [5], FedInv [6], and AFA [7]. These methods perform outlier detection not in a training dataset but rather in the local parameter set $\{\theta_i^{(\tau)}\}$ updated by the clients $\{C_i\}$, and then mitigating or excluding their influence on the aggregation stage. Although the above defense methods demonstrate effectiveness, they also face another challenge. In FL environments, clients often have a different scale sub-dataset, whose class distribution also varies for each client, resulting in heterogeneity in a set of local parameters. Due to this heterogeneity, the updated parameters of the clients, $\theta_1^{(\tau)}, \cdots, \theta_n^{(\tau)}$, are not necessarily similar to each other. This increases the risk of incorrectly detecting clean parameters updated by normal clients as outliers.

Based on the concerns discussed in the last section, this paper proposes a novel defense method to detoxify a recognition model poisoned by CLPA as a complementary approach to the outlier detection-based methods.

III. Proposed Methods

A. Overview

Detoxification of a poisoned model f can be fundamentally achieved through model retraining; that is, the server S in FL can recover the performance of f by fine-tuning it with a sufficient amount of labeled clean samples. However, the main advantage of FL for the server S is that it can receive cooperation from clients to train a recognition model f instead of acquiring a large number of samples by self-effort. Hence, it is impractical for S to obtain substantial quantities of labeled clean samples for fine-tuning. On the other hand, a limited number of labeled clean samples often causes an overfitting problem for the model f, degrading the performance of the fine-tuned model.

To mitigate the overfitting problem, the proposed method limits the scope of parameters targeted in fine-tuning. Typically, neural network models for recognition tasks consist of a backbone part, which extracts a feature from a given sample, and a head part called the network head, which predicts a class label based on the extracted features. Because PA is a stratagem to confuse the relationship between features and class labels, it particularly has a strong effect on the network head that is responsible for class prediction. Hence, fine-tuning the backbone part, which is less vulnerable to PA, is not so effective for detoxification; rather, it may lead to reduced recognition accuracy. For this consideration, we exclude the backbone part from the target of fine-tuning and retrain only the network head.

Furthermore, to complement a small number of labeled clean samples, our proposed method also incorporates the use of a different type of data for fine-tuning. As previously discussed, it is impractical for the server S to collect a large number of labeled clean samples in FL environments. In contrast, it is relatively easy to collect unlabeled OOD samples, where "OOD samples" in our context means data samples of domains or categories that are not necessarily related to any target class of the model f. Thus, the proposed method utilizes unlabeled OOD samples as additional data samples for fine-tuning, which are expected to enhance detoxification performance due to the reasons mentioned in the next section.

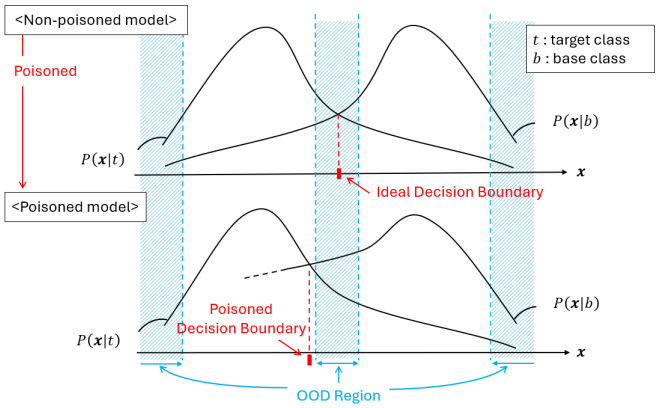


Fig. 2: The reason why OOD samples work well for detoxifying poisoned models. PA manipulate the sample distributions of the target class $P(\boldsymbol{x}|t)$ and the base class $P(\boldsymbol{x}|b)$ to shift the ideal decision boundary, which is generally within the OOD region and can be corrected by OOD samples.

B. Effectiveness of Unlabeled OOD Samples

Before describing the details of the proposed fine-tuning process, we discuss why OOD samples work well.

A non-poisoned recognition model that has the ideal decision boundary outputs the same confidence score for the two bordering classes when input data located on the ideal boundary is given. However, in a poisoned model, its decision boundary has been changed by PA, resulting in uneven confidence scores for the bordering classes. More specifically, a score for the true class is smaller than that of the other class. Calibrating such undesired confidence scores is effective for recovering the model performance, i.e., detoxification. On the other hand, since the data distribution of OOD samples differs from that of clean samples, OOD samples generally lie far from any clean sample in a training dataset of f, namely, either in the marginal regions or near the ideal decision boundary in the feature space. Therefore, if we give a pseudo soft label consisting of a uniform value for all classes to the OOD samples located near the decision boundary, they are expected to be helpful in calibrating the confidence score of a poisoned model f. Fig. 2 depicts the situation. For this reason, fine-tuning with OOD samples is considered an effective method for detoxifying a poisoned model.

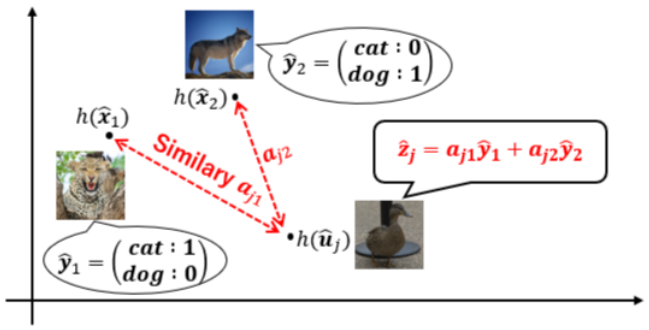


Fig. 3: Pseudo-labeling method for OOD samples considering clean samples in Proposed Method 2.

C. Pseudo-Labeling for Unlabeled OOD Samples

Hereafter, we let $\hat{\boldsymbol{x}}_i$ and $\hat{\boldsymbol{y}}_i$ $(i = 1, \cdots, N)$ denote the i-th clean sample and its label for fine-tuning, respectively, where N is the total number of the clean samples. Note that $\hat{\mathbf{y}}_i$ is represented as a d-dimensional one-hot vector, where d is the number of target classes considered in the model f. Additionally, let $\hat{\boldsymbol{u}}_j$ $(j = 1, \cdots, M)$ be the j-th unlabeled OOD samples, where M $(\gg N)$ is the total number of the unlabeled samples. To fine-tune the poisoned model f with the above samples, we have to give a pseudo soft label $\hat{\boldsymbol{z}}_j$ to each $\hat{\boldsymbol{u}}_j$. In this paper, we propose two ways of this pseudo-labeling, as described below.

The first one is to be used mainly in the case that no labeled clean samples are available, namely the case of $N = 0$. As mentioned in Section III-B, giving a pseudo soft label that has a uniform value for all dimensions is desirable to calibrate the confidence score of f. Thus, we propose to set $\hat{\boldsymbol{z}}_j$ as

$$\hat{\boldsymbol{z}}_j = \left(\frac{1}{n}, \frac{1}{n}, \cdots, \frac{1}{n}\right)^\top . \tag{2}$$

Hereafter, we refer to this soft label as the "fully flat label", and to the fine-tuning the network head of f using OOD samples with these labels as "Proposed Method 1".

The second one is to be used in the case that we can collect at least a small set of labeled clean samples, namely the case of $N \geq 1$. In this case, some of the OOD samples might be somewhat similar to one or more labeled clean samples. For such OOD samples, a fully flat label is not always appropriate. Therefore, we set the pseudo soft label $\hat{\boldsymbol{z}}_j$ as

$$\hat{\boldsymbol{z}}_j = \left(\frac{1}{\sum_{i=1}^{N} a_{ji}}\right) \sum_{i=1}^{N} a_{ji}\hat{\boldsymbol{y}}_i \ , \quad \text{where} \quad a_{ji} = \exp\left(-\frac{1}{\varepsilon^2}\|h(\hat{\boldsymbol{u}}_j) - h(\hat{\boldsymbol{x}}_i)\|^2\right) \tag{3}$$

based on the similarity between $h(\hat{\boldsymbol{u}}_j)$ and $h(\hat{\boldsymbol{x}}_i)$, as shown in Fig.3. In the above Formula, $h(\cdot)$ is the feature extraction part (or the backbone part) of the model f, and ε is a parameter that controls the influence of the similarity between $h(\hat{\boldsymbol{u}}_j)$ and $h(\hat{\boldsymbol{x}}_i)$; $\hat{\boldsymbol{z}}_j$ approaches the fully flat label in the limit as $\varepsilon \to \infty$. We refer to the way of fine-tuning only the network head of f using OOD samples with $\hat{\boldsymbol{z}}_j$ defined by Formula (3) as "Proposed Method 2".

In summary, we proposed two methods to detoxify poisoned recognition models: Proposed Method 1, which can be used regardless of the availability of labeled clean samples, and Proposed Method 2, which is applicable only when a small number of labeled clean samples is available.

D. Loss Function

The loss functions for fine-tuning the network head of a poisoned model f in the two proposed methods are as follows.

The most generally used loss function for training or fine-tuning a recognition model is the cross-entropy loss, which is not necessarily suitable for training data with soft labels. Therefore, our methods employ KL-divergence loss, abbreviated as KLDLoss. Specifically, in Proposed Method 1, we use $\hat{\boldsymbol{z}}_j$ given by Formula (2) to define the KLDLoss L as

$$L = \frac{1}{M}\sum_{j=1}^{M}\sum_{k=1}^{d} \hat{z}_{j,k} \cdot \log \frac{\hat{z}_{j,k}}{\hat{z}'_{j,k}} \ , \tag{4}$$

where $\hat{z}_{j,k}$ represents the k-th dimension of $\hat{\boldsymbol{z}}_j$ and $\hat{z}'_{j,k}$ denotes the k-th dimension of the logit vector $f(\hat{\boldsymbol{u}}_j)$ after the softmax activation. Besides, in Proposed Method 2, we use $\hat{\boldsymbol{z}}_j$ given by Formula (3) and $\hat{\boldsymbol{y}}_i$ for the clean samples to define L as

$$L = \frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{d} \hat{y}_{i,k} \cdot \log \frac{\hat{y}_{i,k}}{\hat{y}'_{i,k}} + \frac{1}{M}\sum_{j=1}^{M}\sum_{k=1}^{d} \hat{z}_{j,k} \cdot \log \frac{\hat{z}_{j,k}}{\hat{z}'_{j,k}} \ , \tag{5}$$

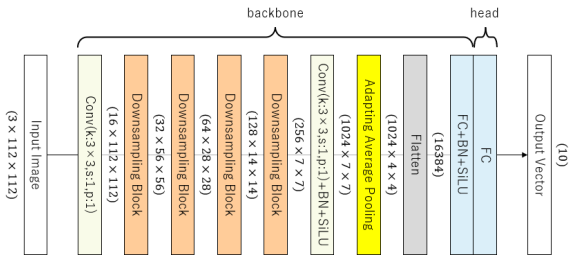
where $\hat{y}_{i,k}$ represents the k-th dimension of $\hat{\boldsymbol{y}}_i$ and $\hat{y}'_{i,k}$ denotes the k-th dimension of the logit vector $f(\hat{\boldsymbol{x}}_i)$ after the softmax activation.

IV. Experiments

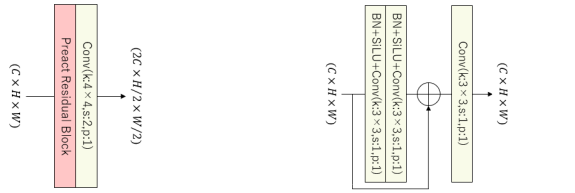
A. Experimental Setup

We experimentally evaluated our two proposed methods described in Section III-C on the task of detoxifying a poisoned model. Since no prior work has addressed this task, to the best of our knowledge, we tested the model's performance gain resulting from applying our methods.

In our experiments, we used 50,000 images in the training set of CIFAR-10 [17], consisting of 5,000 images per class, to train an image recognition model f in an FL environment with five clients, one of which acted as an attacker. Specifically, we first selected 1,000 images (100 images per class) from the 50,000 images as a validation set for monitoring the training process. Then we divided the remaining 49,000 images among the five clients $\{C_i | i = 1, \cdots, 5\}$, whose data distribution was non-uniform. This setup simulates the challenging scenario described in Section II-C, where existing defense methods are less effective. For unlabeled OOD samples and a small number of labeled clean samples required for fine-tuning, we used CIFAR-100 [17]. The CIFAR-100 training set originally contains 50,000 images, 9,940 of which we randomly selected



(a) Overall architecture



(b) Downsampling Block (c) Pre-activation Residual Block [18]

Fig. 4: Network architecture of recognition model used in our experiments. Conv denotes convolutional layers, BN denotes batch normalization layers, SiLU denotes Sigmoid-weighted Linear Units, and FC denotes fully connected layers. For convolutional layers, k, s, and p represent kernel size, stride, and padding size, respectively. C, H, and W represent the number of channels, height, and width of feature maps, respectively.

as OOD samples $\hat{\boldsymbol{u}}_j$ and gave them a pseudo soft label $\hat{\boldsymbol{z}}_j$ based on either Formula (2) or (3). In addition, we further selected 60 images, which are visually similar to the 10 object classes of CIFAR10, out of the remaining 40,060 images and used them as labeled clean samples $\hat{\boldsymbol{x}}_i$ for the Proposed Method 2. Note that we manually annotated these 60 images, i.e., we manually assigned one of the CIFAR10 class labels to them as their ground truth label $\hat{\boldsymbol{y}}_i$ so that the resultant clean image set has 6 images per class.

The network architecture of the image recognition model f used in this experiment is shown in Fig. 4. We regarded only the last fully connected layer as the network head and all the remaining components as the backbone, with fine-tuning performed exclusively on the network head in the proposed methods. As the CLPA method against this f, we employed the one proposed by Zhang et al. [13], whose training configuration was as follows; we set the number of epochs to 500, batch size to 128, loss function to cross-entropy loss, and optimization algorithm to Adam. Note that we began to inject poisoned samples after the 100th epoch of the training process of f rather than immediately after it started. This is because Zhang's CLPA method shows a greater performance degradation capability on image recognition models when attacking after their training process has converged to some extent [19]. We applied our proposed detoxification methods to the model f poisoned in the above procedure.

We evaluated the effectiveness of the proposed methods based on the recognition accuracy of f and its detoxified version on 10,000 test images in CIFAR-10, with 1,000 images per class. The poisoned model f trained under the above settings achieved a recognition accuracy of 66.48%. Using this accuracy as a baseline, we evaluated how much it can be recovered by the proposed methods and comparison methods.

TABLE I: Performance evaluation of Proposed Method 1.

Condition	Target of fine-tuning	
	Entire model	Network head only
Dataset A (clean samples only)	70.52% (+4.04%)	76.88% (+10.40%)
Dataset B (OOD samples only)	67.25% (+0.77%)	77.88% (+11.40%)
Dataset C (both samples)	67.54% (+1.06%)	**78.01% (+11.53%)**

TABLE II: Performance evaluation of Proposed Method 2.

ε	Model's accuracy	ε	Model's accuracy
2.0	76.87% (+10.39%)	12.0	78.34% (+11.86%)
3.0	77.21% (+10.73%)	13.0	78.38% (+11.90%)
4.0	77.37% (+10.89%)	14.0	**78.46% (+11.98%)**
5.0	77.45% (+10.97%)	15.0	78.41% (+11.93%)
6.0	77.42% (+10.94%)	16.0	78.38% (+11.90%)
7.0	77.63% (+11.15%)	17.0	78.28% (+11.80%)
8.0	77.96% (+11.48%)	18.0	78.29% (+11.81%)
9.0	78.34% (+11.86%)	19.0	78.12% (+11.64%)
10.0	78.19% (+11.71%)	20.0	78.10% (+11.62%)
11.0	78.32% (+11.84%)	-	-

B. Results and Discussion

First, we evaluated the effectiveness of Proposed Method 1. In this evaluation, we tested the strategy of fine-tuning the entire model f as a comparison method in addition to the proposed method that fine-tunes only the network head. To separately confirm the impact of a small number of labeled clean samples and that of a large set of unlabeled OOD samples, we used three datasets: Dataset A containing only 60 labeled clean samples, Dataset B containing only 9,940 OOD samples, and Dataset C containing both 60 labeled clean samples and 9,940 OOD samples. Table I shows the evaluation results, where we compared the degree of recovering recognition accuracy from the baseline under each condition. As seen in Table I, fine-tuning only the network head using Datasets B and C achieves higher accuracy than the other conditions. This demonstrates the effectiveness of Proposed Method 1 for detoxifying poisoned models. Comparing the cases of using Datasets A, B, and C in the strategy of fine-tuning only the network head, Dataset C showed the highest accuracy. Since Dataset C is a combination of Datasets A and B, it can be inferred that the labeled clean samples and the OOD samples with fully flat labels provide different recovering effects in a complementary manner. Conversely, in the strategy of fine-tuning the entire model f, Datasets B and C underperform Dataset A with lower accuracy. This suggests that, although ODD samples with the fully flat label are expected to calibrate the confidence scores of a poisoned model, they are generally not suitable for training and prevent performance recovery when used for fine-tuning the entire model.

Next, we evaluated the effectiveness of Proposed Method 2. In this evaluation, we judged there is no need to test the

strategy of fine-tuning the entire model f on the ground of the results in Table I. The dataset used for detoxification was Dataset C, namely a set of 60 labeled clean samples and 9,940 OOD samples with pseudo labels assigned using Formula (3), where the parameter ε was varied from 2.0 to 20.0. Table II shows the results.

We can see from Table II that the result of $\varepsilon = 14.0$ showed an accuracy recovery of +11.98%, which is the highest performance among all the results including those in Table I, demonstrating the effectiveness of Proposed Method 2. Compared with the case of fine-tuning only the network head using Dataset C reported in Table I, Proposed Method 2 performs worse when $\varepsilon < 9.0$ but performs better when $\varepsilon \geq 9.0$. The reason is considered as follows. In Formula (3), ε is a parameter that controls the similarity between each OOD sample $\hat{\boldsymbol{u}}_j$ and clean sample $\hat{\boldsymbol{x}}_i$; a smaller ε results in stronger influence from the nearest $\hat{\boldsymbol{x}}_i$, making $\hat{\boldsymbol{z}}_j$ near a hard label (one-hot vector) and far from a soft label that has uniform value across dimensions. However, since most OOD samples are fundamentally unrelated to any class label targeted by the model f, assigning a highly biased soft label that is close to a hard label to OOD samples would provide a negative effect on the performance recovery. On the other hand, with too large ε, $\hat{\boldsymbol{z}}_j$ approaches the fully flat label too much, diminishing the influence of $\hat{\boldsymbol{x}}_i$. That is why the recognition accuracy began to decrease when $\varepsilon > 14.0$. Based on the above consideration, it is inferred that $\hat{\boldsymbol{z}}_j$ can appropriately reflect the influence of $\hat{\boldsymbol{x}}_i$ while maintaining relatively uniform values across dimensions when $\varepsilon = 14.0$. However, the optimal value of ε might not always be 14.0; it can vary depending on the fine-tuning conditions, such as the number of unlabeled OOD samples and their ratio to labeled clean samples. Currently, the proposed method lacks a mechanism for estimating and setting the optimal value of ε. Therefore, we will focus on developing such a mechanism and formulating it as a theoretical framework in future work. Another drawback of our current method is that we manually annotate clean samples, which is a subjective approach and does not necessarily provide reliable ground truth labels. Thus, we will also consider an automated acquisition approach for labeled clean samples of enough quality.

V. Conclusions

This paper proposed a detoxification method for poisoned models as a countermeasure against CLPA in FL environments. To avoid the overfitting problem, our proposed method fine-tunes only the network head of a given poisoned model. In addition, we use not only a small number of clean samples but also unlabeled OOD samples, which are relatively easy to collect, for fine-tuning. The OOD samples are pseudo-labeled with either a fully flat label or another soft label that considers the influence of clean samples. These pseudo-labeled OOD samples are expected to calibrate the undesired confidence scores of the poisoned model. Our experimental results showed that the proposed method recovers the recognition accuracy of a poisoned image classification model by up to 11.98%, demonstrating its effectiveness. However, in the proposed method, we have not yet theoretically formulated an approach for balancing the uniformity of a soft label and its received influence from clean samples. We will address this issue in future work. This study is partially supported by JST CREST Grant (JPMJCR20D3).

References

[1] Z. Tian, L. Cui, J. Liang, and S. Yu, "A comprehensive survey on poisoning attacks and countermeasures in machine learning," *ACM Computing Surveys*, vol. 55, no. 8, pp. 1–35, 2022.

[2] A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, and T. Goldstein, "Poison frogs! targeted clean-label poisoning attacks on neural networks," *Advances in Neural Information Processing Systems*, vol. 31, pp. 6106–6116, 2018.

[3] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *Advances in Neural Information Processing Systems*, vol. 30, pp. 119–129, 2017.

[4] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proceedings of the 35th International Conference on Machine Learning*. Pmlr, 2018, pp. 5650–5659.

[5] Q. Xia, Z. Tao, Z. Hao, and Q. Li, "Faba: an algorithm for fast aggregation against byzantine attacks in distributed neural networks," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 2019, pp. 4824–4830.

[6] B. Zhao, P. Sun, T. Wang, and K. Jiang, "Fedinv: Byzantine-robust federated learning by inversing local model updates," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 8, 2022, pp. 9171–9179.

[7] L. Muñoz-González, K. T. Co, and E. C. Lupu, "Byzantine-robust federated machine learning through adaptive model averaging," *arXiv preprint arXiv:1909.05125*, 2019.

[8] S. AbdulRahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, and M. Guizani, "A survey on federated learning: The journey from centralized to distributed on-site learning and beyond," *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5476–5497, 2020.

[9] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. PMLR, 2017, pp. 1273–1282.

[10] V. W. Anelli, Y. Deldjoo, T. Di Noia, and A. Ferrara, "Towards effective device-aware federated learning," in *Proceedings of the 18th International Conference of the Italian Association for Artificial Intelligence*. Springer, 2019, pp. 477–491.

[11] C. Yang, Q. Wu, H. Li, and Y. Chen, "Generative poisoning attack method against neural networks," *arXiv preprint arXiv:1703.01340*, 2017.

[12] G. Baruch, M. Baruch, and Y. Goldberg, "A little is enough: Circumventing defenses for distributed learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[13] C. Zhang, Z. Tang, and K. Li, "Clean-label poisoning attack with perturbation causing dominant features," *Information Sciences*, vol. 644, p. 118899, 2023.

[14] B. Biggio, B. Nelson, and P. Laskov, "Support vector machines under adversarial label noise," in *Proceedings of the 3rd Asian Conference on Machine Learning*. PMLR, 2011, pp. 97–112.

[15] L. Muñoz-González, B. Biggio, A. Demontis, A. Paudice, V. Wongrassamee, E. C. Lupu, and F. Roli, "Towards poisoning of deep learning algorithms with back-gradient optimization," in *Proceedings of the 10th ACM workshop on artificial intelligence and security*, 2017, pp. 27–38.

[16] A. Paudice, L. Muñoz-González, A. Gyorgy, and E. C. Lupu, "Detection of adversarial training examples in poisoning attacks through anomaly detection," *arXiv preprint arXiv:1802.03041*, 2018.

[17] A. Krizhevsky, "Learning multiple layers of features from tiny images," 2009.

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

[19] J. Yang, J. Zheng, T. Baker, S. Tang, Y.-a. Tan, and Q. Zhang, "Clean-label poisoning attacks on federated learning for iot," *Expert Systems*, vol. 40, no. 5, p. e13161, 2023.