Received 1 July 2024, accepted 7 July 2024, date of publication 11 July 2024, date of current version 19 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3426534

RESEARCH ARTICLE

Adaptive Selection of Loss Function for Federated Learning Clients Under Adversarial Attacks

SUCHUL LEE
Department of Data Science, Korea National University of Transportation, Uiwang-si, Gyeonggi-do 16106, Republic of Korea
e-mail: sclee@ut.ac.kr

This work was supported by the National Research Foundation of Korea (NRF) Grant through the Korean Government (MSIT) under Grant NRF-2020R1F1A1063942.

ABSTRACT Federated learning (FL) is a deep learning paradigm that allows clients to train deep learning models distributively, keeping raw data local rather than sending it to the cloud, thereby reducing security and privacy concerns. Although FL is designed to be inherently secure, it still has many vulnerabilities. In this paper, we consider an FL scenario where clients are subjected to an adversarial attack that exploits vulnerabilities in the decision-making process of deep learning models to induce misclassification. We observed that adversarial training has a trade-off relationship in which, as classification performance for adversarial examples increases, classification performance for normal samples decreases. To effectively utilize this trade-off relationship in adversarial training, we propose an adaptive selection scheme of the loss function depending on whether the FL client is attacked. The proposed scheme was experimentally proven to achieve the best robust accuracy while minimizing the decrease in natural accuracy. Further, we combined the proposed scheme with Byzantine-robust aggregation. We expected model training to converge stably because Byzantine-robust aggregation prevents highly distorted models from being aggregated, but we obtained experimental results that were contrary to our expectations.

INDEX TERMS Adaptive selection of loss function, adversarial attacks, adversarial training, Byzantine-robust aggregation, federated learning.

I. INTRODUCTION

Since the breakthrough in feasible learning has been made by Hinton et al. [1] in 2006, deep learning research has progressed rapidly over the past two decades. This technology has had a profound impact on human life in a variety of applications, including large-scale language models (LLM), healthcare, edge computing, smart cities, security & privacy, etc. For artificial intelligence (AI) realized through deep learning to truly think at a human level, it must be trained on a massive amount of real-world data [2]. However, deep learning inevitably has limitations in gaining users' trust, collecting personal data, and achieving large-scale practical applications due to cloud computing performance, data security & privacy, and other risks. Therefore, practical and effective technologies are urgently needed to alleviate the above problems and achieve technological leaps. Against this backdrop, the concept of federated learning (FL) has emerged.

The FL concept was first proposed by Google in 2017 and implemented in the keyboard application [3], allowing Android mobile phone users to update deep learning models locally without disclosing private personal data. This system, which focused on running the FedAVG algorithm on mobile phones, can be applied to monitor statistics and perform federated analysis on large-scale cluster machines without logging clients' raw data to the cloud server. FL is one of the most promising technologies in the field of privacy computing, and has become a mainstream solution in many privacy computing application scenarios due to its low technical complexity and advantages in deployment systems. This has also led to many research achievements in the field of FL.

FL is a secure deep learning paradigm that allows clients (users) to train deep learning models distributively while keeping raw data local [4]. Instead, models trained by clients

The associate editor coordinating the review of this manuscript and approving it for publication was Moussa Ayyash.

are sent to cloud servers and integrated into a global model through a model aggregation process. The global model is then distributed to each client to serve as the basis for the next phase of local training. Such FL method realizes the protection of the privacy of local client data as well as the save of computing and storage resources of the server. This approach is in contrast to the traditional centralized learning where all individual client data is collected and trained on cloud servers for model training [5]. FL provides a secure and efficient data federation model for implementing applications that leverage multi-source data scattered across multiple federated agencies. Additionally, it can realize data sharing among federated agencies through systematic expansion of sample size and increase of data dimension for big data fusion, thereby providing high-quality big data services and creating more value for social development. Due to these advantages, it has been applied to various practical applications such as intelligent healthcare [6], 6G communications [7], edge computing [8], smart cities [9], recommender systems [10], security & privacy [11], etc.

The explosive growth in demand for privacy computing platforms has led to an increasing number of real-world FL products, which has also raised several challenges: Current research on FL mainly faces three challenges: privacy and security threats, heterogeneity problem, and huge communication overhead. Early published literature focused on optimizing the learning process of FL. Even if only model parameters are transmitted, the overhead is not negligible, and the longer it takes for the global model to converge, the more severe the overhead becomes. To address this, the research community introduced FedAVG [3], a pioneering communication-efficient FL aggregation algorithm that takes into account data heterogeneity across clients. It is currently one of the most commonly used model aggregation algorithms, and since its proposal, research has been conducted to improve it in response to data heterogeneity, including FedPROX [12] and Scaffold [13].

In addition, FL has many unresolved vulnerabilities that can lead to privacy leaks and malicious attacks [4], [14]. Compared to existing privacy-preserving computing technologies, it was initially assumed that FL would be able to prevent exposure of sensitive information by only transmitting parameters and not storing raw data on the server. However, recent research has shown mathematically that this assumption is not necessarily objective [15]. Poisoning is one of the threats that can easily be inflicted on FL by exploiting vulnerabilities in the deep learning process. It can be divided into data poisoning [16], [17] that distorts learning data and model poisoning [18] that distorts learned models. Although the target of the attack has changed from cloud servers to clients, poisoning remains a powerful threat in FL. The clients under attack are considered Byzantine nodes and various defense strategies can be developed. For example, methods that differentially reflect or isolate distorted models can be applied in the process of integrating them into a global model [19], [20], [21].

Another type of malicious activity that can be committed on FL is an adversarial attack based on a Generative Adversarial Network (GAN) [22]. In this attack, the target of the attack is the same as in poisoning, but a completely different approach is used in defense. This defense mechanism, called adversarial training, preemptively trains a global model on the features of training data exposed to adversarial attacks, i.e. adversarial examples. To the best of our knowledge, Federated Adversarial Training (FAT) [23] is the first attempt to apply adversarial training to the FL environment, and subsequent related studies have been conducted [24], [25], [26], [27], [28], [29]. Unlike traditional deep learning, the ability to learn outlier features of adversarial examples in adversarial training stems from using a relatively flexible min-max based loss function [30]. However, learning outlier features from adversarial examples in FL exacerbates data heterogeneity. Most of the existing literature reported that performance degradation was observed due to data heterogeneity when applying adversarial training to FL. To address this issue, the research community has introduced various approaches to counter adversarial attacks in FL [24], [25], [26], [27], [28], [29]. In particular, [24], [27] alleviates problems caused by data heterogeneity through skewing according to class probability. Reference [25] presented an FL mechanism that propagates adversarial robustness from users with abundant learning resources to users with scarce learning resources. It was shown that aggregation of weights calibrated or reweighed based on adversarial robustness can also be applied [28], [29].

In this paper, we conduct an experimental study on adversarial training in the FL environment. First, we derive the challenges that arise when simply combining adversarial training and FL (Section IV). We observe significant performance degradation in our experiments, and one of the root causes is that there is a trade-off in which models trained to be robust to adversarial examples do not perform well on normal samples. Second, we remove the assumption in the existing literature [23], [24], [25], [26], [27], [28], [29] that "all clients are under attack". It starts from the natural proposition that it is difficult for an attacker to identify all clients, which becomes a much more challenging task since the heterogeneity of the training data distributed across clients is exacerbated. To effectively exploit the aforementioned trade-off, we propose a scheme in which FL clients adaptively select a loss function to perform training depending on whether they are under attack. The proposed scheme was experimentally proven to achieve the best robust accuracy while minimizing the decrease in natural accuracy (Section V). Finally, we extend our proposed scheme by incorporating Byzantine-robust aggregation [19], [20], [21]. With this, we expected that the convergence speed and/or stability of the model would increase, but the experimental results fell short of expectations. Rather, the proposed scheme and Byzantine-robust aggregation are not complementary. This can be explained as follows: adversarial training is ultimately intended to learn adversarial features, and Byzantine-robust aggregation prevents models learned with adversarial features from being aggregated into the global model. (Section VI).

In summary, this paper makes the following contributions.

- We experimentally derive the challenges that arise when simply combining adversarial training and FL. The major challenge is that adversarial training degrades classification performance on normal data, which we call a trade-off.
- We propose a local training scheme that performs adaptive selection of loss function depending on whether the FL client is under attack. The proposed scheme effectively balances the trade-off in a realistic scenario where some clients are under attack, thereby maximizing the classification performance on adversarial examples while maintaining the classification performance on normal samples.
- We extend our proposed scheme by combining it with Byzantine-robust aggregation rules and explore its dynamics.

This paper is structured as follows. Section II reviews related work. Section III defines the problem considered in this paper. Section IV describes the experiment that motivated the proposed scheme. In Section V, we propose and evaluate an adaptive loss function selection method. And Section VI extends our scheme to combine Byzantine-robust aggregation. Finally, Section VII concludes this paper.

II. RELATED WORK

FL has been continuously studied since FedAVG, the basis of the model aggregation method, was introduced by McMahan et al. [3] in 2017. In FL, if the data distributed to clients does not follow IID (independent and identically distributed), model learning performance suffers severe degradation. To solve this, there have been attempts to optimize FedAVG, such as FedPROX [12] and Scaffold [13]. FedPROX introduced a proximal term to limit delays in the global model convergence process due to data heterogeneity. Scaffold proposed a method to accelerate the convergence of the global model by reducing the gradient change of local updates using control variables.

Various studies have been conducted on security and privacy in FL. Attacks on FL that undermine robustness are broadly classified into training phase attacks and inference phase attacks. Poisoning is a typical training phase attack, including data poisoning [16], [17] and model poisoning [18]. The goal of data pollution is to intentionally perturb local data, leading to biased global model training. Data poisoning is performed by repeatedly injecting and accumulating small perturbations to the global model over multiple rounds to avoid detection, often significantly degrading the performance of the final global model [16], [17]. Model poisoning is performed in a more direct way by manipulating local model parameters rather than compromising data integrity to induce biased global model [18].

Adversarial attacks [30], [31], [32] intentionally manipulate deep learning models by injecting carefully crafted input data, that is, adversarial examples, into the training data. These attacks aim to cause incorrect classifications by exploiting vulnerabilities in the decision-making process of deep learning models. Goodfellow et al. [32] proposed the Fast Gradient Sign Method (FGSM), a method for generating adversarial examples by performing gradient ascent using a single-step first-order approximation. Projected Gradient Descent (PGD) is a form of repeatedly performing FGSM, and the more repetitions are accumulated, the more powerful the adversarial attack becomes. In particular, it is mathematically shown in the study by Madry et al. [30] that PGD is a first order adversary, which implies that a deep learning model trained to be robust to PGD is also robust to other adversarial attacks. Defense against such adversarial adversaries involves adversarial training, a special training strategy that leverages a min-max based loss function. Since its proposal, there have been attempts to improve the min-max based loss function. For example, Zhang et al. [33] proposed the TRADES loss function, which separates the prediction error for adversarial examples into natural error and bounded error and maintains a mutual balance between them. In [34], the MART loss function was proposed to improve prediction error.

FAT, which simply combines adversarial training and FL, was proposed in Zizzo et al. [23]. This study demonstrated the feasibility of FAT in IID and non-IID data, stimulating active follow-up research. Most of the literature consistently shows that simply combining adversarial training with FL significantly degrades performance, especially on non-IID data [24], [25], [26], [27], [28], [29]. Chen et al. [27] showed that skewed labels lead to non-identical class probabilities and heterogeneous local models, and proposed CalFAT that tackles this issue by calibrating the logits adaptively to balance the classes. Similarly, Zhang et al. [24] calibrates the logits before softmax cross-entropy according to the probability of occurrence of each class. Meanwhile, some studies [25], [28], [29] have proposed differential aggregation methods that take into account the significance of local models. In particular, [25] presented an aggregation method that propagates adversarial robustness from users with abundant learning resources to users with scarce learning resources. It was shown that aggregation of calibrated or reweighed weights based on adversarial robustness can also be applied [28], [29]. On the other hands, Shah et al. [26] focused on the communication overhead, and proposed a dynamic schedule for the number of local training epochs at each round. Unlike previous studies that applied adversarial training to FL [24], [25], [26], [27], [28], [29], this paper removes the assumption that all FL clients are under attack. In the situation where some of the clients are under attack, we propose an effective yet simple scheme that performs

TABLE 1. Comparison of related studies and the proposed scheme that applied adversarial training to FL.

Reference	Contribution	Novelty
Zizzo *et al.* [23]	• The first work that applies adversarial training to FL • It opened the door to research in the field and promoted the activation of related research.	• Derivation of requirements for adversarial robustness in a combination of adversarial training and FL • Examination of them from the perspective of model convergence.
Chen *et al.* [27], Zhang *et al.* [24]	• Proposal of an aggregation scheme that can balance classes through adaptive logit calibration	• Addressing the label asymmetry problem, which is one of the root causes of instability and decreased accuracy in FL training
Hong *et al.* [25]	• Proposal of an aggregation scheme that propagates adversarial robustness from users with abundant learning resources to users with scarce learning resources	• Presenting a practical solution to cooperatively alleviate the imbalance of computing and data resources between clients
Shah *et al.* [26]	• Proposal of a training scheme that improves both natural and adversarial accuracy and shortens model convergence time by dynamically adjusting the number of local epochs	• Experimentally revealing the influence of the number of epochs during which local adversarial training is performed
Zhu *et al.* [28], Zhang *et al.* [29]	• Proposal of an aggregation scheme for calibrated or re-weighted local models based on adversarial robustness	• Design of a differential aggregation method for local models with different adversarial robustness
Proposed scheme	• Proposal of a local training scheme that performs adaptive selection of loss function depending on whether the client is under attack	• Design of local training strategies for local clients depending on whether they are under attack • Examination of the dynamics for applying Byzantine-robust aggregation to the proposed scheme

adaptive selection of loss function depending on whether the client is under attack or not. Moreover, we experimentally demonstrate the dynamics of the proposed scheme when combined with Byzantine-robust aggregation [19], [20], [21], inspired by an idea that the attacked client can be considered as a Byzantine-fault node. We summarize relevant existing studies applying adversarial training to FL along with our proposed method in Table 1.

III. PROBLEM DEFINITION

In this section, we define the FL system and threat model considered in this paper.

A. SYSTEM MODEL

We assume a cross-silo FL model [4] for the purpose of detecting malicious Android apps in cellular networks, as shown in Figure 1. In the proposed FL model, the cloud data server becomes the central entity. Since most terminals are smartphones and the number of users is very large, it may seem reasonable to apply a cross-device FL model. However, users' personal data is often not enough to effectively train local models, and more importantly, cross-device models require excessive communication bandwidth between the central server and clients for model updates. This is unnecessary and unwelcome for individual smartphone users. Therefore, in the proposed FL model, it is assumed that a base station with relatively better computing and communication performance than the client becomes the edge entity (silo 2), or that the clients form a coalition and some members of it represent the coalition and act as the edge entity (silo 1 and 3).

B. MATHEMATICAL FORMULATION

We assume that the training data is distributed across a total of K edge entities. In general, FL performs the following optimizations:

$$\min_{w} f(w) = \sum_{k=1}^{K} \frac{n_k}{n} F_k(w). \tag{1}$$

Here, we denote that the global approximate optimal is a sum of local models weighted by the local data size n_k, and n is the total data size of all edge entities that participate in a communication round. Each edge entity measures the empirical risk over possibly different data distributions D_k, which can be expressed as:

$$F_k(w) := \mathbb{E}_{x_k \sim D_k}[f_k(w; x_k)]. \tag{2}$$

Threat Model: The main content of existing adversarial attack-related research is 'Can a prediction error in a deep learning model be caused by injecting a very small perturbation into the data?'. In other words, 'Is it possible to craft powerful adversarial examples by injecting very small perturbations into the data?'. To answer the above question, a method to create adversarial examples through FGSM [32] was proposed. Let x be the original sample, x_{adv} be the corresponding adversarial example, and their labels y. We also assume that we are given a suitable loss function $L(\theta, x, y)$. Then, the injection of small perturbations δ by FGSM is written as follows:

$$x_{adv} = x + \underbrace{\epsilon sgn(\nabla_x L(\theta, x, y))}_{\delta} \tag{3}$$

where the second term, whose size is constrained by small ϵ values, represents these small perturbations. As shown in Equation (3), in FGSM, small perturbations are injected only once. PGD generalizes FGSM so that perturbation injection can be performed repeatedly (e.g., t steps), enabling much more powerful adversarial attacks. It has been mathematically proven in [30] that models trained to be robust against PGD

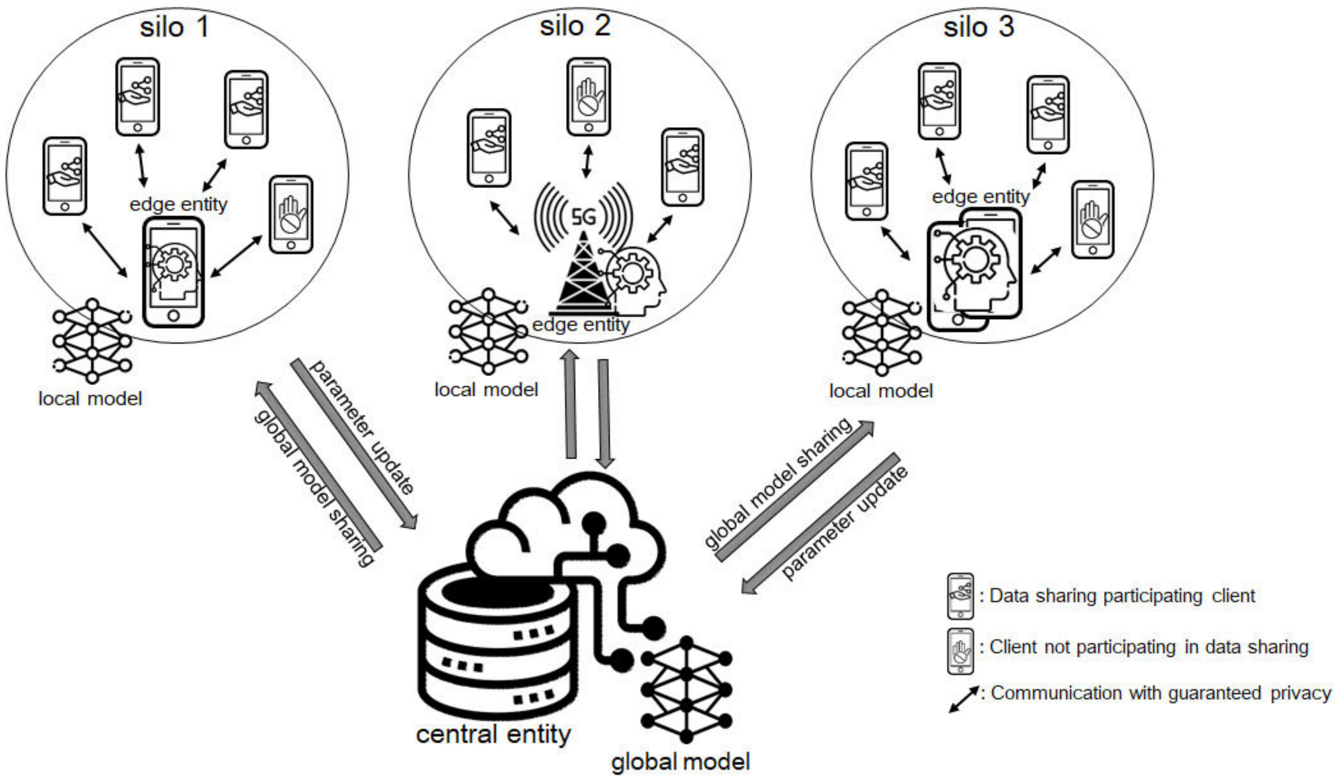

FIGURE 1. **The cross-silo FL model assumed in this paper consists of one central entity and a plurality of edge entities. The edge entity is a form of coalition between the client(s) and the base station, and a representative part of the component terminals communicates directly with the central entity to participate in global model training.**

attacks are robust against other adversarial attacks. This property is defined as a first order adversary, and therefore PGD has been adopted as a threat model in several existing studies dealing with adversarial training. Here, effective training of adversarial features is realized by maximizing the loss function $L(w, x^t_{adv}, y)$ in the local training of each edge entity. Substituting the loss function applied with PGD into equation (2), local training for edge entities now becomes equivalent to solving the following min-max optimization problem:

$$F_k(w) := \min \mathbb{E}_{x_k \sim D_k} \left[\max_{\|x^t_{adv} - x\|_\infty \leq \delta} L(w, x^t_{adv}, y) \right]. \quad (4)$$

We can see the basic principle of adversarial training in equation (4), i.e., the min-max based loss function, where the inner maximization problem aims to find effective adversarial examples that achieve high loss, while the outer minimization optimizes the training loss for edge entities.

IV. MOTIVATIONAL EXPERIMENTS

In this section, we describe the experiments and observations that motivated the proposed scheme. Before going into the details of the results, we describe the experimental setup.

A. EXPERIMENTAL SETUP

1) DATASETS

We used the CICMalDroid 2020 dataset [35] in the experiments. The CICMaldroid dataset is publicly available data collected in the form of Android Package Kit (APK) files, an Android app installation file, from third parties such as VirusTotal and Contagio blog from December 2017 to December 2018. It consists of a total of 17,341 samples, all of which belong to benign or one of four types of malware (adware, banking malware, mobile riskware, and SMS malware). The APK file was converted to an image file using stream order [36]. This transforms our problem into an image classification problem. Among a total of 17,341 Android app samples converted to images, 14,000 samples are randomly selected and used as training data, and the remaining 3341 samples are used as evaluation data. 14,000 training samples are randomly distributed to a total of 100 edge entities, which means that the distribution of training data across edge entities follows IID.

2) IMPLEMENTATION PLATFORM

PyTorch [38] has been adopted by many literature and researchers due to its flexible graph modeling, strong community support, and Python-based efficiency and scalability. Based on these advantages, we implemented all our code used in the experiments in PyTorch.

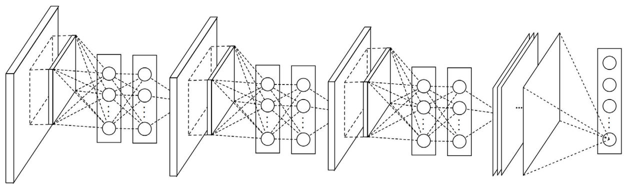
FIGURE 2. **NIN using multiple MLP convolutional layers on the front and a global mean pooling layer.**

3) NEURAL NETWORK ARCHITECTURE

Adversarial training is implemented to learn data perturbation patterns through internal optimization of the min-max loss function [30]. To take full advantage of the dual optimization of adversarial training, the neural network being trained must have a structure complex enough to accommodate the aspect of perturbation. Therefore, we adopted Network in Network (NIN), a fairly complex and well-known neural network [37]. As shown in Figure 2, NIN's dual structure, implemented using layers modularized through 1×1 convolution along with existing convolution layers, is designed to train rich features, improving the model's expressiveness and classification performance. Although NIN's dual structure is designed to be quite complex, global average pooling has been used to make it lightweight for training in practice. In [26] and [28], NIN was used in adversarial training using the CIFAR datasets that have a certain level of classification difficulty among publicly available datasets. On this basis, we believe that the dual structure is suitable for training the perturbation of adversarial examples.

4) FL PARAMETERS

In each round, 10% of the 100 edge entities are selected to participate in local training. Local training takes place in 10 rounds, and the local batch size is 10. The learning rate, which affects the model's convergence speed, was set to 0.01. Although these parameters have a significant impact on the performance of FL, their impact has been sufficiently discussed in previous studies [3], [26], [36], so we do not discuss them in this paper and use their default values.

5) PERFORMANCE METRICS

Robust accuracy refers to the performance of a deep learning model on adversarial examples. It measures how well the model performs when tested against inputs that are intentionally manipulated to cause misclassification or errors. Natural accuracy refers to the performance of the model on the original, unaltered test examples, without any intentional adversarial perturbations. It measures how well the model performs under normal conditions, where the inputs are not deliberately crafted to deceive the model. In this paper, robust accuracy and natural accuracy are expressed as '*robust acc*' and '*natural acc*', respectively.

6) COMPARED SCHEMES

Standard training (ST) refers to general federated learning in a situation that is not affected by adversarial attacks. This is typical federated learning, forming a baseline for situations that are not under attack. Therefore, we refer to it as Baseline-ST. In the other schemes, all edge entities are subject to adversarial attacks. In particular, there is a case of adversarial training using a min-max based loss function in equation (4), which is typical federated adversarial training, becoming a baseline for situations that are under attack. Therefore, we refer to it as Baseline-AT. Unlike Baseline-AT, TRADES and MART use loss functions developed in [33] and [34], respectively.

B. EXPERIMENTAL OBSERVATION

We describe some observations that motivated us to conduct this study. Figure 3 shows the classification performance of the deep learning model obtained through four training methods.

1) TRADE-OFF

Robust acc and *natural acc* are in a trade-off relationship. In order to increase *robust acc*, the unique features of adversarial examples must be trained, and the more the model trains these features, the more it has a negative impact on *natural acc*. This can be explained as follows. An adversarial attack injects a perturbation into the data to move and/or obscure the location of the 'inter-class boundary' to be trained in a deep learning model. In particular, ambiguous boundaries must be trained using a relatively flexible min-max based loss function, and this flexibility in boundaries causes misclassification of unattacked data. In other words, a model well trained on the features of adversarial examples often causes misclassification of normal samples.

When not subjected to PGD-20 attack, i.e., Baseline-ST *natural acc* is about 85% and *robust acc* is 23.3%. Considering that the dataset consists of five classes, it can be seen that the model trained without any exposure to the features of adversarial attacks has almost no classification function for PGD-20 based adversarial examples. All three adversarial training schemes (Baseline-AT, TRADES, MART) achieved similar performance after sufficient training rounds, with the *robust acc* being around 50%. Baseline-AT shows a performance curve similar to the other two adversarial training schemes, but as the training round exceeds 160, *natural acc* deteriorates significantly and a drift-shaped curve is observed. This means that simply combining adversarial training and FL is difficult to apply in practice, because the purpose of using adversarial training is to improve the classification performance of adversarial samples while maintaining an appropriate level of classification accuracy for normal samples. Meanwhile, TRADES, which uses the loss function proposed in [33], is the only training method whose *natural acc* is higher than *robust acc* under PGD-20 attack. *Natural acc* is about 55% and *robust acc* is about 50%. These results show that TRADES' loss function performs a form of training that most effectively utilizes the trade-off relationship mentioned above. MART's *robust acc* performs similarly to other adversarial training methods.

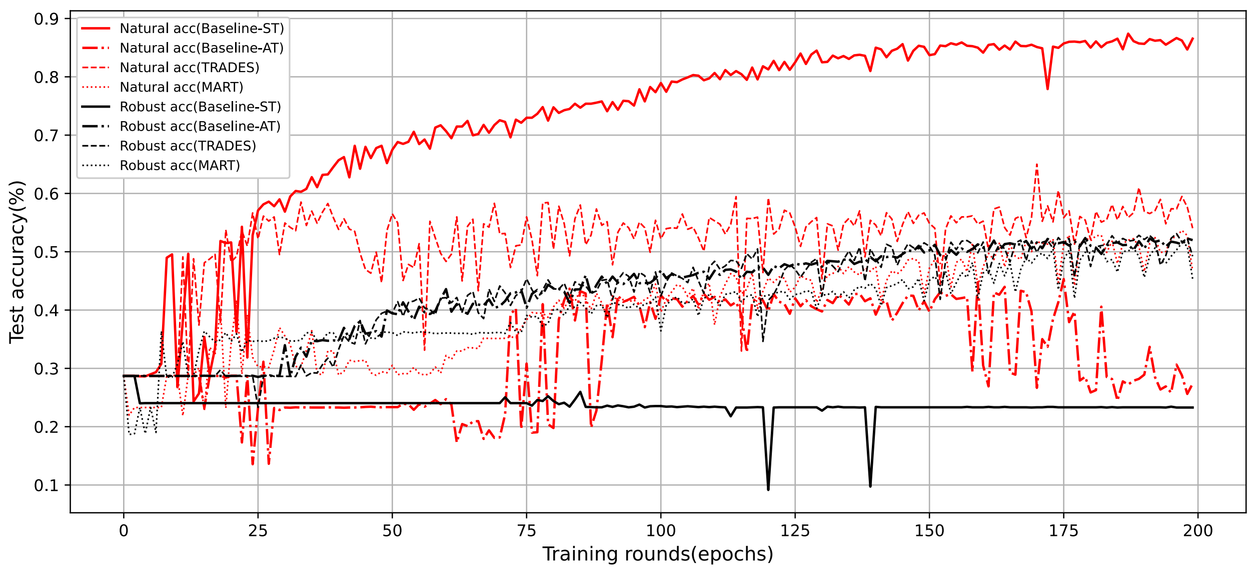

FIGURE 3. **The classification performance of adversarial training is compared according to the adoption of the loss function, and the performance curve of federated learning when there is no attack is also plotted.**

However, in terms of *natural acc*, it was clearly observed that performance was lower than that of TRADES.

2) SUMMARY

We made the following observations: 1) Simply combining adversarial training with FL leads to performance degradation. 2) One of the root causes of this performance degradation is that models trained to be robust to adversarial examples have a trade-off relationship where they perform poorly on normal samples. 3) A training strategy is needed to effectively balance this relationship.

V. ADAPTIVE SELECTION OF LOSS FUNCTION

From the experimental observations in the previous section, we need an adversarial training strategy that can effectively exploit the apparent trade-off between *robust acc* and *natural acc*. Specifically, under adversarial attacks, even considering the decrease in *natural acc*, adversarial training must be used to learn the features of adversarial examples. On the other hand, in the absence of adversarial attacks, standard training must be used, as unnecessary adversarial training can lead to a loss of *natural acc*.

Some Edge Entities Are Under Attack: Prior studies regarding adversarial training on FL assume a situation where all clients are attacked, which is not natural in practice. This is because the attacker cannot know all the clients in the FL environment, as well as which clients participate in each training round. Therefore, we remove the above unnatural but common assumption. Below, we briefly describe the experimental setup involving the adversarial adversary considered in this paper. We define a new term, attack rate (AR), as the fraction of edge entities exposed to adversarial attacks. For example, if AR is 1, all edge entities are under adversarial attacks, which is the same scenario as prior studies [24], [25], [26], [27], [28], [29]. In our experiments, AR values are set from 0.1 to 0.9 in steps of 0.2. Note that edge entities participating in local training rounds are set to change continuously, while edge entities under attack remain unchanged. It is because this setting is more consistent with real-world situations.

Proposed Scheme: According to observations from the previous section, we have shown experimentally that performing adversarial training using the TRADES loss function [33] or the MART loss function [34] is better than using a traditional min-max based loss function. In particular, in the case of *robust acc*, best performance could be reached regardless of loss function selection, but this was not the case in *natural acc*.

We propose a scheme for edge entities that adaptively selects a loss function in adversarial training depending on whether the edge entity is under attack. For example, the edge entities that are attacked perform adversarial training using the TRADES or the MART loss function, while the other edge entities perform standard FL learning (i.e., non-adversarial training). Note that edge entities can choose between adversarial and standard training simply by adopting different loss functions.

The proposed approach should allow models trained using different loss functions to be aggregated. To this end, two challenges must be addressed: (i) When starting each local training round, the edge entity can check whether it is under attack by inferring local data with the global model sent from the central entity. However, in this paper, we assume that we know the list of attacked edge entities. The mathematical analysis required to remove this assumption and/or the development of algorithms to detect attacks on their own are beyond the scope of this paper and are left to future work. (ii) The second question is whether models trained

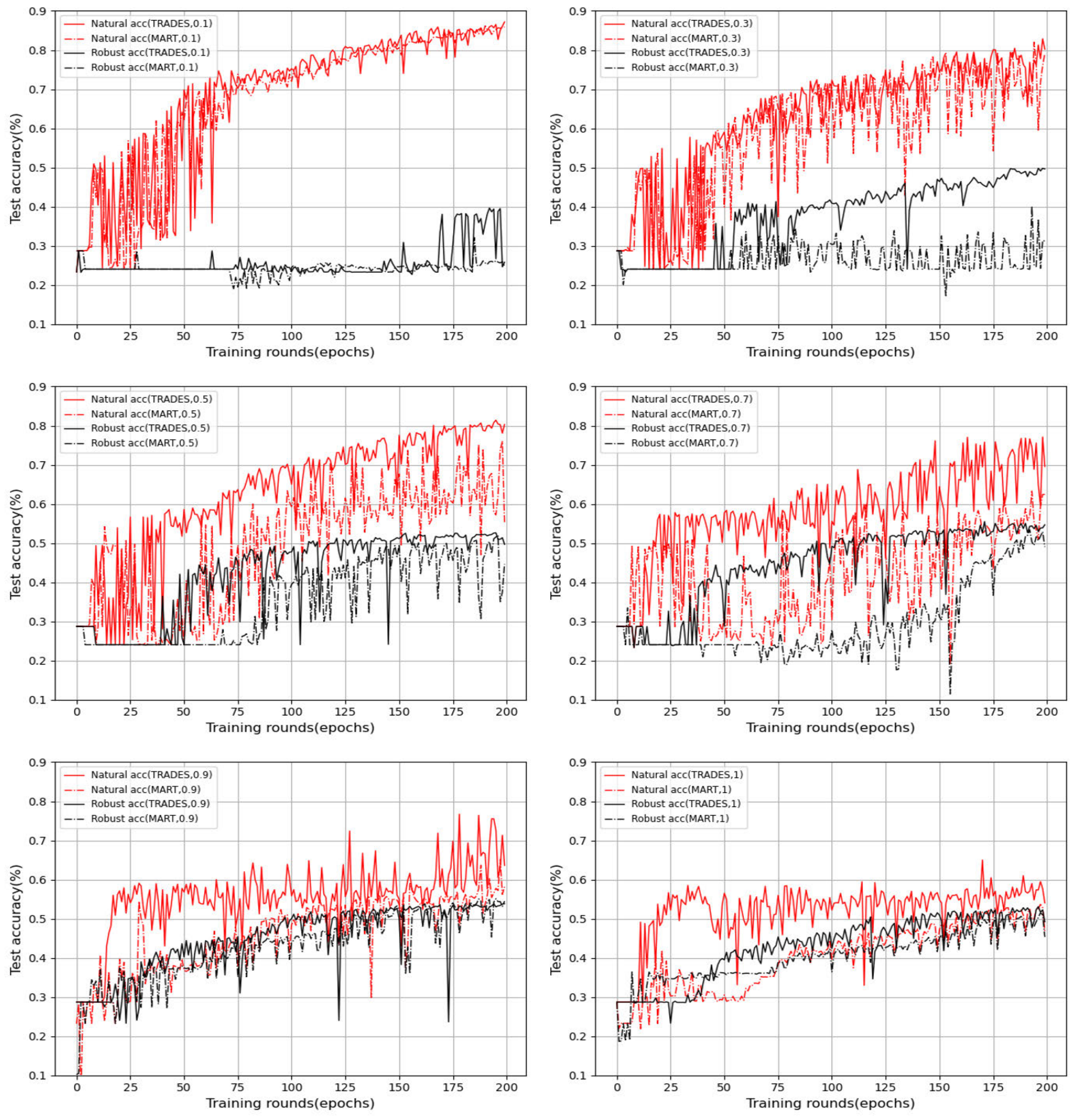

FIGURE 4. **Comparison of performance changes in adversarial training using two loss functions (TRADES, MART) as the proportion of edge entities subjected to adversarial attack increases from 0.1 to 0.9.**

using different loss functions can be integrated through an aggregation process such as FedAVG. We experimentally have proven that model parameters of the same network structure can be directly aggregated even if the models are trained with different loss functions.

A. RESULTS

Figure 4 shows the deep learning model's malicious app classification performance when some of the edge entities are under adversarial attacks. The decimal point shown in the legend is the AR value. The total number of edge entities in the experiment is 100, and 10 edge entities participate in FL in each learning round. Therefore, an AR value of 0.1 means that on average, one attacked edge entity participates in each FL training round.

1) WHEN A SMALL NUMBER OF EDGE ENTITIES ARE UNDER ATTACK

When AR = 0.1, both TRADES and MART can hardly train the features of adversarial examples. We did not observe a decrease in *natural acc* caused by training on adversarial features. In the case of MART, *robust acc* is no different from that in the case where adversarial training was not performed. In the case of TRADES, it appears that the adversarial features begin to melt into the model as the training round exceeds 150, but not only is the convergence of training slow, but the fluctuation is severe, so it is not significant. These experimental results can be interpreted that the opportunity to train the features of adversarial examples is limited in the scenario where AR = 0.1. When AR = 0.3, the best *natural acc* for both schemes decreases to about 80%. However, a clear difference is found in *robust acc*. Through the *robust acc* curve, we observed that adversarial training was performed in the case of TRADES, and it converged at about 48%. In the case of MART, the *robust acc* curve shows that adversarial training is performed to some extent, but the fluctuations are severe and it does not converge to a particular point.

2) WHEN HALF OF THE EDGE ENTITIES ARE UNDER ATTACK

When AR = 0.5, the *natural acc* of TRADES does not show much difference compared to that when AR = 0.3. However, in the case of MART, not only did it experience severe fluctuations, but its best performance also dropped to about 75%. The best *robust acc* is about 50% for both TRADES and MART schemes, which means that the features of the adversarial samples have been sufficiently trained. However, MART's *robust acc* curve shows significant fluctuations and appears to lack convergence stability, and its overall performance is also lower than that of adversarial training using the TRADES loss function.

3) WHEN A LARGE NUMBER OF EDGE ENTITIES ARE UNDER ATTACK

When AR = 0.7, both TRADES and MART sufficiently train the features of adversarial attacks. In the case of adversarial training using the TRADES loss function, significant fluctuations were observed in the *natural acc* curve, and the best performance decreased to about 75%. Training of features of adversarial samples begins in the early stages of FL, and shows a stable convergence *robust acc* curve. For adversarial training using the MART loss function, we observed dramatic performance degradation and fluctuations in the *natural acc* curve due to training on adversarial features. The best *natural acc* was around 60%. The best *robust acc* is 50%, which is similar to the case of AR = 0.5, but fluctuation is reduced. When AR = 0.9, both TRADES and MART sufficiently train the capabilities of adversarial attacks. In the *robust acc* curve, it was observed that both schemes converged stably and without fluctuation to about 50%, which is the best performance. However, the *natural acc* curve shows a noticeable difference. TRADES' best *natural acc* is around 75%, but there is a lot of fluctuation between 60% and 70%, so it is difficult to say that it is representative. However, it is clear that the performance is better than adversarial training using the MART loss function. In fact, when AR = 0.9, it is difficult to find any difference from the case where all edge entities are subject to adversarial attacks (last graph in Figure 4), except for the *natural acc* curve of TRADES.

4) SUMMARY

We summarize our proposed scheme and its experimental results as follows: 1) We propose a scheme that adaptively selects a loss function and performs local training depending on whether or not it is under adversarial attacks. 2) The proposed scheme achieves *robust acc* comparable to when all clients are under adversarial attacks while minimizing the decrease in *natural acc*. 3) In particular, the TRADES loss function is the most effective among the loss functions considered in this paper.

VI. COMBINING WITH BYZANTINE-ROBUST AGGREGATION

In the previous section, the edge entities exposed to adversarial attacks were trained adversarially, while the others were trained standardly. Now we extend the proposed scheme by incorporating it into the aggregation process. To this end, we consider aggregation rules to discriminately reflect or exclude edge entity models that are under adversarial attacks. The rationale for applying such a strategy is as follows. Training adversarial features on a small number of edge entities is sufficient to induce a robust global model, while excessive training can significantly degrade *natural acc*.

A. EXISTING BYZANTINE-ROBUST AGGREGATION RULES

Before combining the proposed approach with Byzantine-robust aggregation rules, we review some rules that are frequently used in related studies.

1) KRUM [19] AND BULYAN [21]

Bulyan is an algorithm designed based on Krum. Krum selects one of the local models that are similar to other models transmitted to the central server as the global model. Intuitively, even if the local model of the client under attack is selected, its impact may be limited because it is similar to other local models. It is assumed that among the total m edge entities, up to e_{atk} edge entities are attacked. For each local model, the central server finds m-e_{atk}-2 local models with the smallest Euclidean distance from the local model. Among these m-e_{atk}-2 local models, the local model with the smallest squared distance is selected as the global model. Therefore, Krum cannot be completely free from the influence of attacked edge entity models. To address this, Mhamdi et al. [21] proposed Bulyan, which essentially combines variants of Krum and trimmed averaging. The i_{th} parameter of the global model is calculated as follows. Bulyan first applies Krum iteratively to select θ ($\theta \leq m$-$2e_{atk}$) local models. Sort the i_{th} parameters of the selected θ models, and use the average of γ ($\gamma \leq \theta$-$2e_{atk}$) parameters close to the median as the i_{th} parameter of the global model.

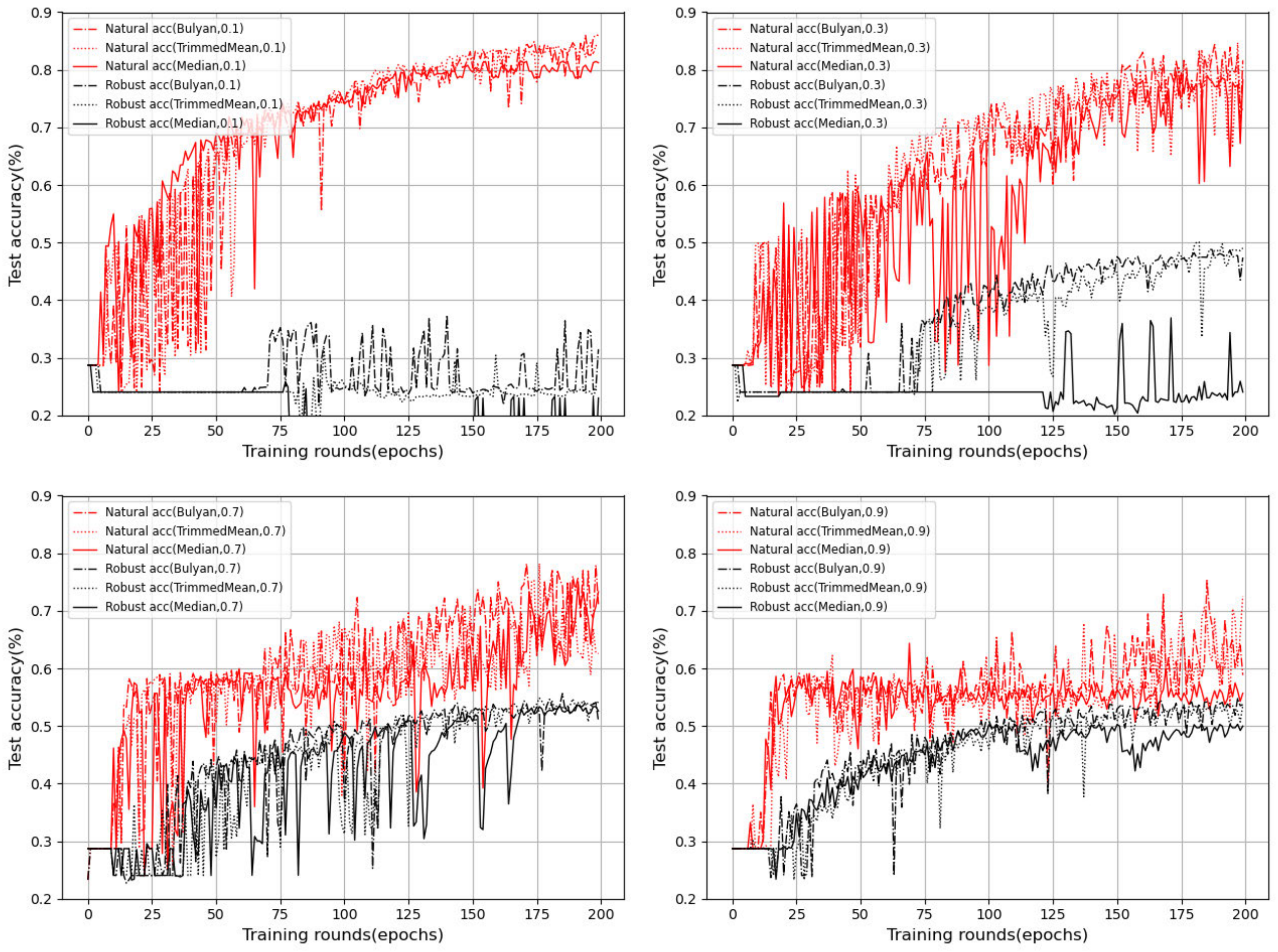

FIGURE 5. **Comparison of the performance changes of adversarial training combining the proposed loss function adaptive selection method and Byzantine-robust aggregation.**

2) TRIMMED MEAN AND MEDIAN [20]

This aggregation rule aggregates the i_{th} parameter of the global model as follows: Specifically, sorting is performed on the i_{th} model parameter of local models transmitted from the edge entity. In Trimmed mean aggregation, the average of the sorted i_{th} parameter values excluding the min and max values is used as the i_{th} parameter of the global model. And the aggregation method that uses the middle value in the sorted list of i_{th} parameter values as the i_{th} parameter of the global model is called Median.

B. RESULTS

We now combine our adaptive loss function selection scheme with the Byzantine-robust aggregation rules. Specifically, in the model aggregation process of the central entity, Bulyan, Trimmed mean, and Median aggregation are applied instead of FedAVG. The TRADES loss function, which showed the best adversarial training performance in the section V, is used in local adversarial training. Here, Byzantine-robust aggregation algorithms are basically methods of isolating nodes that are attacked (referred to as Byzantine-fault nodes) during the aggregation process, so the number of Byzantine nodes must be entered into the algorithm. In this experiment, the number of edge entities participating in each FL round $\times$ AR was entered as the parameter. Figure 5 shows the experimental results.

1) WHEN A SMALL NUMBER OF EDGE ENTITIES ARE UNDER ATTACK

When AR $= 0.1$, *natural acc* shows a similar form of convergence for all three aggregation methods (Bulyan, Trimmed mean, Median). We expected that the convergence curve of *natural acc* would stabilize faster by isolating Byzantine-fault nodes from the global model aggregation, but we observed no experimental results meeting our predictions. Even for *robust acc*, Median cannot train adversarial features at all. This can be explained as follows. Since Median, as the name implies, selects the median value as the aggregated ones, it is rare for the model of the edge entity under attack to be selected as the global model.

This phenomenon becomes worse when AR $= 0.3$. When Median aggregation was used, *natural acc* became more fluctuating, but converged to the best performance. However, in *robust acc*, the local model that trained adversarial features still does not merge into the global model through Median aggregation. Bulyan and Trimmed mean showed *robust acc* curves similar to FedAVG.

2) WHEN A LARGE NUMBER OF EDGE ENTITIES ARE UNDER ATTACK

When AR = 0.7, the best *natural acc* shows no significant difference among all aggregation methods, but significant fluctuations are observed. In particular, in the case of Median aggregation, fluctuation is most noticeable because the aggregate value is determined by the value of a specific local model. The convergence pattern of the *robust acc* curve is also similar to that of *natural acc*.

This convergence pattern is reversed when AR = 0.9. Median aggregation's best *natural acc* is about 55%, a decrease of about 15-20%, and fluctuations were greatly reduced during the training process. This is explained for the following reasons. Median aggregation selects the median as the combined value, so most of the selected model parameters come from the local model of the edge entity under attack. This makes it difficult to combine models from different edge entities (e.g. those that are not under attack). In the *robust acc* curve of Median aggregation, a sharp drop is observed during convergence, and the best performance was observed to be about 50%, which is about 5% lower than other aggregation methods.

3) SUMMARY

We summarize experimental results on the combination of our proposed scheme and Byzantine-robust aggregation. 1) We expected that the convergence speed and/or stability of the model would increase by separating the severely distorted models, but the experimental results were contrary to our expectations. 2) Rather, the proposed scheme and Byzantine-robust aggregation are not complementary. This can be explained as follows: adversarial training is ultimately intended to learn adversarial features, and Byzantine-robust aggregation prevents models learned with adversarial features from being aggregated into the global model. The Median method that performs extreme separation is particularly incompatible with adversarial training.

VII. CONCLUSION

In this paper, we experimentally investigated the performance degradation problem and underlying causes of the simple combination of adversarial training and FL. This was due to the trade-off relationship in which the accuracy for normal samples decreases as the features of adversarial examples are trained. To leverage this, we proposed a method to perform FL local training through adaptive selection of loss functions, and demonstrated its effectiveness through extensive experiments. Further, we applied our proposed scheme to Byzantine-robust aggregation to isolate models that overfit to adversarial features, thereby improving convergence speed and/or model stability. Contrary to expectations, we experimentally confirmed that these schemes are not complementary and cannot currently be used simultaneously.

A. APPLICATIONS

Since adversarial attacks are very powerful attacks that can be easily implemented without requiring much prior information, our research results can be widely used as a countermeasure in various security & privacy fields such as cellular network, edge computing, medical, and malware detection.

B. LIMITATIONS AND FUTURE WORK

Most deep learning based malware classification/detection are applicable only to attacks that are known in advance. In this context, the answer to the question, "How can we detect newly emerged malware (so-called zero-day attacks)?" has been a major obstacle for deep learning to solve security problems. Existing unsupervised learning faces accuracy issues, which limits its practical use. Recently, research on semi-supervised learning, which utilizes both labeled and unlabeled data during learning, has become increasingly popular. In the security field, data labeling is expensive, time-consuming, and requires expert knowledge, and there is usually a huge amount of unlabeled data. Semi-supervised learning utilizes this richness to learn better representations and improve detection capabilities. Therefore, research on semi-supervised learning will be one of the future research directions in the security & privacy fields that can answer the above question.

REFERENCES

[1] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006, doi: 10.1162/neco.2006.18.7.1527.

[2] A. S. Fokas, "Can artificial intelligence reach human thought?" *PNAS Nexus*, vol. 2, no. 12, Dec. 2023, Art. no. pgad409, doi: 10.1093/pnasnexus/pgad409.

[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, Fort Lauderdale, FL, USA, 2017, pp. 1273–1282.

[4] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, "A survey on security and privacy of federated learning," *Future Gener. Comput. Syst.*, vol. 115, pp. 619–640, Feb. 2021, doi: 10.1016/j.future.2020.10.007.

[5] G. Drainakis, K. V. Katsaros, P. Pantazopoulos, V. Sourlas, and A. Amditis, "Federated vs. centralized machine learning under privacy-elastic users: A comparative analysis," in *Proc. IEEE 19th Int. Symp. Netw. Comput. Appl. (NCA)*, Cambridge, MA, USA, Nov. 2020, pp. 1–8.

[6] D. N. Sachin, B. Annappa, S. Hegde, C. S. Abhijit, and S. Ambesange, "FedCure: A heterogeneity-aware personalized federated learning framework for intelligent healthcare applications in IoMT environments," *IEEE Access*, vol. 12, pp. 15867–15883, 2024, doi: 10.1109/ACCESS.2024.3357514.

[7] Z. Yang, M. Chen, K.-K. Wong, H. V. Poor, and S. Cui, "Federated learning for 6G: Applications, challenges, and opportunities," *Engineering*, vol. 8, pp. 33–41, Jan. 2022, doi: 10.1016/j.eng.2021.12.002.

[8] Y. Ye, S. Li, F. Liu, Y. Tang, and W. Hu, "EdgeFed: Optimized federated learning based on edge computing," *IEEE Access*, vol. 8, pp. 209191–209198, 2020, doi: 10.1109/ACCESS.2020.3038287.

[9] X. Yuan, J. Chen, J. Yang, N. Zhang, T. Yang, T. Han, and A. Taherkordi, "FedSTN: Graph representation driven federated learning for edge computing enabled urban traffic flow prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 8, pp. 8738–8748, Aug. 2023, doi: 10.1109/TITS.2022.3157056.

[10] K. Muhammad, Q. Wang, D. O'Reilly-Morgan, E. Tragos, B. Smyth, N. Hurley, J. Geraci, and A. Lawlor, "Fedfast: Going beyond average for faster training of federated recommender systems," in *Proc. ACM SIGKDD KDD* Aug. 2020, pp. 1234–1242.

[11] M. Tayyab, M. Marjani, N. Z. Jhanjhi, I. A. T. Hashem, R. S. A. Usmani, and F. Qamar, "A comprehensive review on deep learning algorithms: Security and privacy issues," *Comput. Secur.*, vol. 131, Aug. 2023, Art. no. 103297, doi: 10.1016/j.cose.2023.103297.

[12] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proc. MLSys*, 2020, pp. 429–450.

[13] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "SCAFFOLD: Stochastic controlled averaging for federated learning," in *Proc. ICML* 2020, pp. 5132–5143.

[14] L. Lyu, H. Yu, X. Ma, C. Chen, L. Sun, J. Zhao, Q. Yang, and P. S. Yu, "Privacy and robustness in federated learning: Attacks and defenses," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 7, pp. 8726–8746, Jul. 2022, doi: 10.1109/TNNLS.2022.3216981.

[15] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," *Proc. ICML* 2019, pp. 634–643.

[16] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," in *Proc. ESORICS*, Guildford, U.K., 2020, pp. 480–501.

[17] S. Farhadkhani, R. Guerraoui, and O. Villemaud, "An equivalence between data poisoning and Byzantine gradient attacks," in *Proc. ICML*, Baltimore, MD, USA, 2022, pp. 6284–6323.

[18] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to Byzantine-robust federated learning," in *Proc. USENIX Secur.*, 2020, pp. 1605–1622.

[19] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Proc. NIPS*, Long Beach, CA, USA, 2017, pp. 1–11.

[20] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proc. Int. Conf. Mach. Learn.*, Stockholm, Sweden, Jul. 2018, pp. 5650–5659.

[21] E. M. E. Mhamdi, R. Guerraoui, and S. Rouault, "The hidden vulnerability of distributed learning in byzantium," in *Proc. ICML*, Stockholm, Sweden, 2018, pp. 3521–3530.

[22] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, C. Aaron, and Y. Bengio, "Generative adversarial nets," in *Proc. NIPS*, Montreal, QC, Canada, 2014, pp. 1–9.

[23] G. Zizzo, A. Rawat, M. Sinn, and B. Buesser, "FAT: Federated adversarial training," in *Proc. NIPS*, 2020, pp. 1–8.

[24] J. Zhang, Z. Li, B. Li, J. Xu, S. Wu, S. Ding, and C. Wu, "Federated learning with label distribution skew via logits calibration," in *Proc. ICML*, Baltimore, MD, USA, 2022, pp. 26311–26329.

[25] J. Hong, H. Wang, Z. Wang, and J. Zhou, "Federated robustness propagation: Sharing adversarial robustness in heterogeneous federated learning," in *Proc. AAAI*, 2023, vol. 37, no. 7, pp. 7893–7901.

[26] D. Shah, P. Dube, S. Chakraborty, and A. Verma, "Adversarial training in communication constrained federated learning," 2021, *arXiv:2103.01319*.

[27] C. Chen, Y. Liu, X. Ma, and L. Lyu, "CalFAT: Calibrated federated adversarial training with label skewness," in *Proc. NIPS*, 2022, pp. 1–13.

[28] J. Zhu, J. Yao, T. Liu, Q. Yao, J. Xu, and B. Han, "Combating exacerbated heterogeneity for robust models in federated learning," in *Proc. ICLR*, Kigali, Rwanda, 2023, pp. 1–34.

[29] J. Zhang, B. Li, C. Chen, L. Lyu, S. Wu, S. Ding, and C. Wu, "Delving into the adversarial robustness of federated learning," in *Proc. AAAI*, 2023, vol. 37, no. 9, pp. 11245–11253.

[30] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Proc. ICLR*, Vancouver, BC, Canada, 2018.

[31] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the GAN: Information leakage from collaborative deep learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Dallas, TX, USA, Oct. 2017, pp. 603–618.

[32] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014, *arXiv:1412.6572*.

[33] H. Zhang, Y. Yu, J. Jiao, E. P. Xing, L. E. Ghaoui, and M. I. Jordan, "Theoretically principled trade-off between robustness and accuracy," in *Proc. ICML*, Long beach, CA, USA, 2019, pp. 7472–7482.

[34] Y. Wang, D. Zou, J. Yi, J. Bailey, X. Ma, and Q. Gu, "Improving adversarial robustness requires revisiting misclassified examples," in *Proc. ICLR*, New Orleans, LA, USA, 2019, pp. 1–14.

[35] (2020). *CICMalDroid 2020 Datasets*. [Online]. Available: https://www.unb.ca/cic/datasets/maldroid-2020.html

[36] S. Lee, "Distributed detection of malicious Android apps while preserving privacy using federated learning," *Sensors*, vol. 23, no. 4, p. 2198, Feb. 2023, doi: 10.3390/s23042198.

[37] M. Lin, Q. Chen, and S. Yan, "Network in network," in *Proc. ICLR*, Banff, AB, Canada, 2014.

[38] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, and S. Chintala, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. NIPS*, 2019, pp. 1–12.

SUCHUL LEE received the B.S. and Ph.D. degrees in computer science and engineering from Seoul National University, South Korea, in 2008 and 2014, respectively. From 2014 to 2016, he was a Member of Senior Research Staff at the National Security Research Institute, South Korea. Since 2016, he has been an Associate Professor with the Department of Artificial Intelligence and Data Engineering, Korea National University of Transportation, South Korea. His research interests include artificial intelligence, information security, internet applications, wireless and mobile communications, such as 4/5/6G cellular networks, 802.11, cognitive radio, and traffic analysis, with a particular focus on data analytics and performance optimization.