FADE: Enabling Large-Scale Federated Adversarial Training on Resource-Constrained Edge Devices

Minxue Tang[1], **Jianyi Zhang**[1], **Mingyuan Ma**[1], **Louis DiValentin**[2], **Aolin Ding**[2], **Amin Hassanzadeh**[2], **Hai Li**[1], **Yiran Chen**[1]
[1]Department of Electrical and Computer Engineering, Duke University
[2]Cybersecurity Lab, Accenture
[1]{minxue.tang,jianyi.zhang,mingyuan.ma,hai.li,yiran.chen}@duke.edu
[2]{louis.divalentin,aolin.ding,amin.hassanzadeh}@accenture.com

Abstract

Adversarial Training (AT) has been proven to be an effective method of introducing strong adversarial robustness into deep neural networks. However, the high computational cost of AT prohibits the deployment of large-scale AT on resource-constrained edge devices, e.g., with limited computing power and small memory footprint, in Federated Learning (FL) applications. Very few previous studies have tried to tackle these constraints in FL at the same time. In this paper, we propose a new framework named Federated Adversarial Decoupled Learning (FADE) to enable AT on resource-constrained edge devices in FL. FADE reduces the computation and memory usage by applying Decoupled Greedy Learning (DGL) to federated adversarial training such that each client only needs to perform AT on a small module of the entire model in each communication round. In addition, we improve vanilla DGL by adding an auxiliary weight decay to alleviate objective inconsistency and achieve better performance. FADE offers a theoretical guarantee for the adversarial robustness and convergence. The experimental results also show that FADE can significantly reduce the computing resources consumed by AT while maintaining almost the same accuracy and robustness as fully joint training.

1 Introduction

As a privacy-preserving distributed learning paradigm, Federated Learning (FL) makes a meaningful step toward the practice of secure and trustworthy artificial intelligence [9, 11, 12, 15, 22]. In contrast to traditional centralized training, FL pushes the training to edge devices (clients), and only locally trained models are uploaded to the server for aggregation. Since no private data needs to be shared with other clients or the server, FL substantially improves the data privacy during the training process.

Although FL can provide strong privacy benefits, there exist other threats to the reliability of the machine learning model running on the FL system. One of such threats is adversarial attack, which aims to cause misclassifications of the model by adding imperceptible noise into the input data [6, 27, 30, 39]. In centralized training, a model can improve the robustness against adversarial attack by supplementing the training set with adversarial samples, i.e., adversarial training (AT). However, considering the systematic heterogeneity in FL and the large computational cost of AT, some edge devices with limited computing resources, such as mobile phones and IoT devices, may not be able to afford AT [8, 9, 15, 35]. Table 1 shows that strong robustness of the whole FL system cannot be attained by allowing only a small portion (e.g., 20%) of the clients to perform AT. Therefore, enabling clients with constrained computing resources (which usually contribute to majority of the clients in cross-device FL) to perform AT is necessary for achieving strong robustness in FL.

Preprint. Under review.

Table 1: Results of partial federated adversarial training with 1000 clients. "20% AT + 80% ST" means that 20% clients perform AT while 80% clients perform standard training (ST).

Training Scheme	FMNIST		CIFAR-10	
	Natural Acc.	Adversarial Acc.	Natural Acc.	Adversarial Acc.
100% AT + 0% ST	83.05%	70.44%	64.99%	28.66%
20% AT + 80% ST	85.65%	56.45%	74.51%	19.24%

Some previous works have tried to tackle client-wise systematic heterogeneity in FL [16, 19, 33, 37]. The most common method to deal with the slow devices is to allow them performing less epochs of local training than the others [16, 33]. While this method can reduce the amount of computation on the slow devices, the memory capacity limitation on edge devices has not been well discussed in these works. Since a large model is usually needed to achieve both high accuracy and strong robustness, a small memory becomes another bottleneck of applying AT on edge devices.

Another method to reduce both computation and memory requirements during training is Decoupled Greedy Learning (DGL) [3, 34], which decouples the whole neural network into several modules and trains each module separately. DGL can be naturally deployed in FL since the training of each module can be parallelized on different clients [3]. However, DGL has not yet been explored by the literature about AT, and some modifications on vanilla DGL are required to ensure the robustness of the entire network. Furthermore, vanilla DGL will increase the objective inconsistency on different clients and make FL even more difficult to converge in Non-IID settings [17, 33, 34].

In this paper, we propose **F**ederated **A**dversarial **DE**coupled Learning (FADE), which is the first to deploy and improve DGL in federated adversarial training. Our main contributions are:

1. We propose a more flexible decoupled learning scheme for heterogeneous Federated Learning, which allows different module partitions on devices with different resource budgets. We give a theoretical guarantee for the convergence of Federated Decoupled Learning.

2. We propose an adversarial training strategy, FADE, for Federated Decoupled Learning to attain theoretically guaranteed joint adversarial robustness in the whole model. Our experimental results show that FADE can achieve almost the same adversarial robustness as joint adversarial training, while largely reduce the memory and computation consumption.

3. We analyse the trade-off between objective consistency (natural accuracy) and adversarial robustness (adversarial accuracy) in FADE, and we propose an effective method to adjust the balance between them with the weight decay on auxiliary model.

2 Preliminary

Federated Learning In FL, different clients collaboratively train a shared global model $\boldsymbol{w}$ with locally stored data [22]. The objective of FL can be formulated as:

$$L(\boldsymbol{w}) = \frac{1}{\sum_i |\mathbb{D}_i|} \sum_{k=1}^{N} \sum_{(\boldsymbol{x},y)\in\mathbb{D}_k} l(\boldsymbol{x}, y; \boldsymbol{w}) = \sum_{k=1}^{N} p_k L_k(\boldsymbol{w}), \tag{1}$$

$$L_k(\boldsymbol{w}) = \frac{1}{|\mathbb{D}_k|} \sum_{(\boldsymbol{x},y)\in\mathbb{D}_k} l(\boldsymbol{x}, y; \boldsymbol{w}) = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathbb{D}_k} \left[l(\boldsymbol{x}, y; \boldsymbol{w}) \right], \tag{2}$$

where l is the task loss, e.g., cross-entropy loss for classification task. $\mathbb{D}_k$ and $p_k = |\mathbb{D}_k|/(\sum_i |\mathbb{D}_i|)$ is the dataset and weight of client k respectively. To solve for the optimal solution of this objective, in each communication round, FL first samples a subset of clients $\mathbb{S}^{(t)}$ to perform local training. These clients initialize their models with the global model $\boldsymbol{w}_k^{(t,0)} = \boldsymbol{w}^{(t)}$, and then run τ iterations of local SGD. After all these clients complete training in this round, their models are uploaded and averaged

to become the new global model [22]. We summarize this procedure as follows:

$$\boldsymbol{w}_k^{(t+1)} = \boldsymbol{w}^{(t)} - \eta_t \sum_{j=0}^{\tau-1} \nabla L_k(\boldsymbol{w}_k^{(t,j)}), \tag{3}$$

$$\boldsymbol{w}^{(t+1)} = \frac{1}{\sum_{i\in\mathbb{S}^{(t)}} p_i} \sum_{k\in\mathbb{S}^{(t)}} p_k \boldsymbol{w}_k^{(t+1)}, \tag{4}$$

where $\boldsymbol{w}_k^{(t,j)}$ is the local model of client k at the j-th iteration of round t.

Adversarial Training The goal of AT is to achieve robustness against small perturbation in the inputs. This can be formulated as follows [23]: given a perturbation tolerance ϵ, for some threshold c,

$$\forall \boldsymbol{\delta} \in \{\boldsymbol{\delta} : \|\boldsymbol{\delta}\|_p \leq \epsilon\}, l(\boldsymbol{x}+\boldsymbol{\delta}, y; \boldsymbol{w}) - l(\boldsymbol{x}, y; \boldsymbol{w}) \leq c, \tag{5}$$

where $\|\cdot\|_p$ is the ℓ_p norm of a vector[1]. The most popular method to achieve adversarial robustness is to train a model with adversarial samples, which can be formulated as a min-max problem [6, 20]:

$$\min_{\boldsymbol{w}} \max_{\boldsymbol{\delta}:\|\boldsymbol{\delta}\|_p\leq\epsilon} l(\boldsymbol{x}+\boldsymbol{\delta}, y; \boldsymbol{w}). \tag{6}$$

To solve Eq. 6, people usually alternatively solve the inner maximization and the outer minimization. While solving the inner maximization, Projected Gradient Descent (PGD) is shown to introduce the strongest robustness in AT [20]. However, performing PGD requires multiple iterations of backpropagation through the whole model, which introduces large computational overhead [35, 23].

3 Federated Adversarial Decoupled Learning (FADE)

In this section, we will introduce our method, Federated Adversarial Decoupled Learning (FADE), which aims at enabling all clients with different computing resources to participate in adversarial training. We first introduce the vanilla deployment of DGL in FL in Section 3.1. We then extend DGL into adversarial training in Section 3.2, and we give a theoretical analysis about its robustness. In Section 3.3, we propose an effective improvement for DGL in order to alleviate the objective inconsistency. The convergence analysis of FADE will be given in Section 3.4.

3.1 Federated Decoupled Learning

In cross-device FL, the main participants are usually small edge devices [9], whose poor computing resources (e.g., low computing power and small memory) may slow down and even prohibit them from training a large model [15, 16, 33]. Recently, Decoupled Greedy Learning (DGL) is proposed to reduce the computation and memory demand when training a deep neural network [2, 3, 34]. In DGL, the whole model is decoupled into several non-overlapping modules, and each module contains one or multiple adjacent layers of the neural network. One advantage of DGL is that each module can be loaded into the memory and trained independently, which allows the parallelization of the module training on different computing nodes [3]. Furthermore, since each node only needs to train a module instead of the whole model, the computing resource requirements on each node are largely reduced.

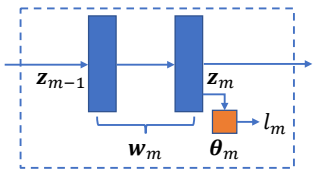

Figure 1: An illustration of the module m.

The key idea of DGL is introducing an auxiliary output model for each module such that each module can use this early-exit loss l_m to perform locally supervised training [3, 34]. As shown in Fig. 1, we denote a module $m = \{\boldsymbol{w}_m, \boldsymbol{\theta}_m\}$ as the set of the parameters in this module, where $\boldsymbol{w}_m$ is the parameters in the backbone model, and $\boldsymbol{\theta}_m$ is the parameters in the auxiliary output model of this module. Let $\boldsymbol{z}_m^{(t)} = f_m(\boldsymbol{z}_{m-1}^{(t)}; \boldsymbol{w}_m^{(t)})$ be the feature extracted by the module m at communication round t. We can formulate the locally supervised loss of the module m at communication round t as

$$L_m^{(t)}(\boldsymbol{w}_m^{(t)}, \boldsymbol{\theta}_m^{(t)}) = \mathbb{E}_{(\boldsymbol{z}_{m-1}^{(t)}, y)} \left[l_m(\boldsymbol{z}_{m-1}^{(t)}, y; \boldsymbol{w}_m^{(t)}, \boldsymbol{\theta}_m^{(t)}) \right], \tag{7}$$

[1]For simplicity, without specifying p, we use $\|\cdot\|$ for ℓ_2 norm. Our conclusions in the following sections can be extended to any ℓ_p norm with the equivalence of vector norms.

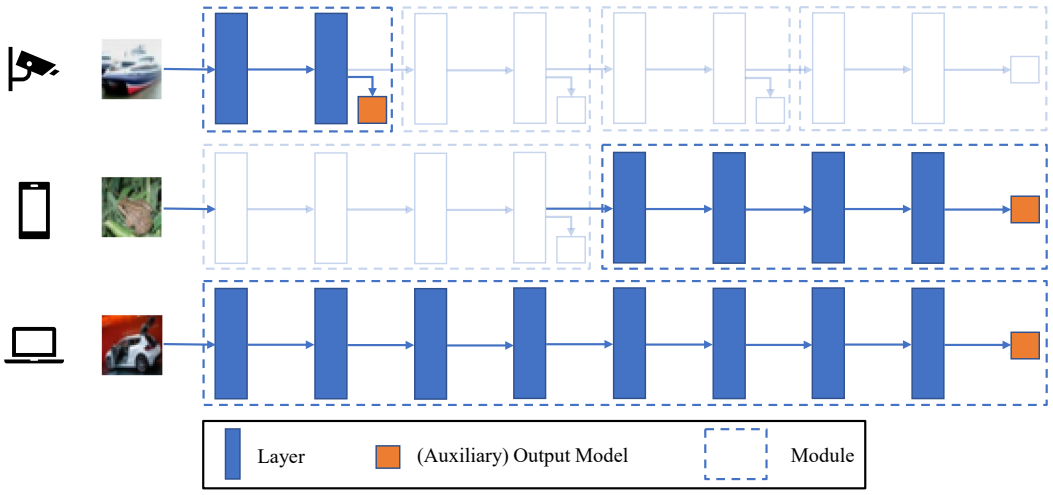

Figure 2: A framework of Federated Decoupled Learning. In contrast to the original unique partition [3], we allow different devices to adopt different module partitions according to their own computing resources. In each communication round, each device randomly select one module (highlighted) for training, and then only upload this trained module to the server for aggregation.

where $\boldsymbol{z}_0^{(t)} = \boldsymbol{x}$ is the original input of the whole model. When deploying DGL in FL, considering the systematic heterogeneity, we modify the original DGL [3] and allow different clients to partition the backbone model in different ways according to their own computing resources. In each communication round t, each client k will randomly select one module m_k^t for training. Let $\boldsymbol{\Theta}_m = (\boldsymbol{w}_m, \boldsymbol{\theta}_m)$, and $\boldsymbol{\omega}$ be any parameter in the whole model (including backbone network $\boldsymbol{w}$ and all auxiliary models $\boldsymbol{\theta}$). The training and aggregation rule for Federated Decoupled Learning can be formulated as

$$\boldsymbol{\Theta}_{m_k^t,k}^{(t+1)} = \boldsymbol{\Theta}_{m_k^t}^{(t)} - \eta_t \sum_{j=0}^{\tau-1} \nabla L_{m_k^t,k}^{(t)}(\boldsymbol{\Theta}_{m_k^t,k}^{(t,j)}), \tag{8}$$

$$\boldsymbol{\omega}^{(t+1)} = \frac{1}{\sum_{i\in\mathbb{S}_{\boldsymbol{\omega}}^{(t)}} p_i} \sum_{k\in\mathbb{S}_{\boldsymbol{\omega}}^{(t)}} p_k \boldsymbol{\omega}_k^{(t+1)}, \quad \text{where } \mathbb{S}_{\boldsymbol{\omega}}^{(t)} = \{k : \boldsymbol{\omega} \in m_k^t\}. \tag{9}$$

Here $L_{m_k^t,k}^{(t)}$ and $\boldsymbol{\Theta}_{m_k^t,k}^{(t,j)}$ are the local loss and the local model of client k as before in Section 2, and $\mathbb{S}_{\boldsymbol{\omega}}^{(t)}$ is the set of clients whose selected module m_k^t contains the parameter $\boldsymbol{\omega}$. Fig. 2 illustrates how clients with different computing resources collaborate in Federated Decoupled Learning.

3.2 Adversarial Decoupled Learning

A naive combination of AT and DGL is that one can generate adversarial samples by backpropagations in only the first module, and use them as $\boldsymbol{z}_0$ for DGL. However, this naive combination will not introduce strong robustness in the whole model. We can get some intuition for this phenomenon from the chain rule used in backpropogation:

$$\frac{\partial l}{\partial \boldsymbol{x}} = \frac{\partial \boldsymbol{z}_1}{\partial \boldsymbol{x}} \frac{\partial \boldsymbol{z}_2}{\partial \boldsymbol{z}_1} \cdots \frac{\partial l}{\partial \boldsymbol{z}_M}, \tag{10}$$

where M is the total number of modules. In a first-order view, AT decreases the gradient in input ($\partial l/\partial \boldsymbol{x}$) to attain robustness [23, 24]. While performing AT in the first module will force the first module to extract robust feature and consequently decrease the first factor ($\partial \boldsymbol{z}_1/\partial \boldsymbol{x}$) in Eq. 10 [26], the other factors in the chain rule remains unconstrained and the overall gradient can still be large.

Accordingly, to achieve a strong robustness in the whole model, we propose to perform AT in all the modules separately, where the objective of each module m is given by

$$\min_{\boldsymbol{\Theta}_m} \max_{\boldsymbol{\delta}_{m-1}} \quad l_m(\boldsymbol{z}_{m-1} + \boldsymbol{\delta}_{m-1}, y; \boldsymbol{\Theta}_m), \quad \text{subject to } \|\boldsymbol{\delta}_{m-1}\| \leq \epsilon_{m-1}. \tag{11}$$

With our adversarial decoupled learning, the robustness of the whole network can be guaranteed by the following theorem, which is proved in Appendix A.1.

Theorem 1. *Omitting all parameters $\boldsymbol{w}$ and $\boldsymbol{\theta}$, for adjacent modules m and $m+1$, given $\boldsymbol{z}_{m-1}$ and $\boldsymbol{z}_m = f_m(\boldsymbol{z}_{m-1})$, let $\tilde{l}_m(\boldsymbol{z}_m, y) = \tilde{l}_m(f_m(\boldsymbol{z}_{m-1}), y) = l_m(\boldsymbol{z}_{m-1}, y)$ and assume that $\tilde{l}_m(\boldsymbol{z}_m, y)$ is μ_m-strongly convex in $\boldsymbol{z}_m$, if we perform AT on these two modules separately such that*

$$\forall \boldsymbol{\delta}_{m-1} \in \{\boldsymbol{\delta}_{m-1} : \|\boldsymbol{\delta}_{m-1}\| \leq \epsilon_{m-1}\}, l_m(\boldsymbol{z}_{m-1} + \boldsymbol{\delta}_{m-1}, y) - l_m(\boldsymbol{z}_{m-1}, y) \leq c_m; \tag{12}$$

$$\forall \boldsymbol{\delta}_m \in \{\boldsymbol{\delta}_m : \|\boldsymbol{\delta}_m\| \leq \epsilon_m\}, l_{m+1}(\boldsymbol{z}_m + \boldsymbol{\delta}_m, y) - l_{m+1}(\boldsymbol{z}_m, y) \leq c_{m+1}, \tag{13}$$

with ϵ_m satisfying

$$\epsilon_m \geq \frac{g_m}{\mu_m} + \sqrt{\frac{2c_m}{\mu_m} + \frac{g_m^2}{\mu_m^2}}, \quad \textit{where } g_m = \|\nabla_{\boldsymbol{z}_m} \tilde{l}_m(\boldsymbol{z}_m, y)\|, \tag{14}$$

we will have the overall robustness in the joint model of these two modules:

$$\begin{aligned} &\forall \boldsymbol{\delta}_{m-1} \in \{\boldsymbol{\delta}_{m-1} : \|\boldsymbol{\delta}_{m-1}\| \leq \epsilon_{m-1}\}, \\ &l_{m+1}(f_m(\boldsymbol{z}_{m-1} + \boldsymbol{\delta}_{m-1}), y) - l_{m+1}(f_m(\boldsymbol{z}_{m-1}), y) \leq c_{m+1}. \end{aligned} \tag{15}$$

We can simply induce in m to get the robustness guarantee in the whole model with Theorem 1.

Remark 1. In Theorem 1, we assume that $\tilde{l}_m(\boldsymbol{z}_m, y)$ is strongly convex, namely, the auxiliary output model is strongly convex in its input $\boldsymbol{z}_m$. This assumption is realistic since the auxiliary model is usually a very simple model, e.g., only a linear layer followed by cross-entropy loss.

Remark 2. Theorem 1 gives a sufficient condition on ϵ_m for the overall robustness. We notice that with a larger μ_m which is determined by $\boldsymbol{\theta}_m$, the lower bound of ϵ_m can be smaller and thus the overall robustness will be easier to attain. We will discuss this property in more details in Section 3.3.

3.3 Alleviating Objective Inconsistency

One problem in DGL is that DGL will cause objective inconsistency in different modules, since they are trained with isolated gradients [34]. While objective inconsistency among clients can hinder the convergence of FL [17, 33], deploying vanilla DGL in FL will lead to serious performance drop.

The objective inconsistency in DGL is caused by the difference between the module gradient ($\nabla_{\boldsymbol{w}_m} l_m$) and the joint gradient ($\nabla_{\boldsymbol{w}_m} l$), which makes the optimal solution that minimizes the locally supervised loss l_m does not necessarily minimize the joint loss l. The following theorem gives the upper bound of this difference under certain conditions, and we prove it in Appendix A.2.

Theorem 2. *Omitting all parameters $\boldsymbol{w}$ and $\boldsymbol{\theta}$, given $\boldsymbol{z}_m = f_m(\boldsymbol{z}_{m-1}) = f_m \circ f_{m-1} \circ \cdots \circ f_1(\boldsymbol{x})$, let $\tilde{l}_m(\boldsymbol{z}_m, y) = \tilde{l}_m(f_m(\boldsymbol{z}_{m-1}), y) = l_m(\boldsymbol{z}_{m-1}, y)$ and $\tilde{l}(\boldsymbol{z}_m, y) = \tilde{l}(f_m \circ \cdots \circ f_1(\boldsymbol{x}), y) = l(\boldsymbol{x}, y)$, assume $\tilde{l}_m(\boldsymbol{z}_m, y)$ and $\tilde{l}(\boldsymbol{z}_m, y)$ are β_m, β'_m-smooth in $\boldsymbol{z}_m$, if there exist c_m and c'_m such that*

$$\begin{aligned} &\exists r \geq \sqrt{2\frac{c_m + c'_m}{\beta_m + \beta'_m}}, \forall \boldsymbol{\delta}_m \in \{\boldsymbol{\delta}_m : \|\boldsymbol{\delta}_m\| \leq r\}, \\ &|\tilde{l}_m(\boldsymbol{z}_m + \boldsymbol{\delta}_m, y) - \tilde{l}_m(\boldsymbol{z}_m, y)| \leq c_m, \end{aligned} \tag{16}$$

$$|\tilde{l}(\boldsymbol{z}_m + \boldsymbol{\delta}_m, y) - \tilde{l}(\boldsymbol{z}_m, y)| \leq c'_m, \tag{17}$$

then we have

$$\|\nabla_{\boldsymbol{w}_m} l - \nabla_{\boldsymbol{w}_m} l_m\| \leq \left\| \frac{\partial \boldsymbol{z}_m}{\partial \boldsymbol{w}_m} \right\| \sqrt{2(c_m + c'_m)(\beta_m + \beta'_m)}. \tag{18}$$

The first factor in Eq. 18, $\|\partial \boldsymbol{z}_m / \partial \boldsymbol{w}_m\|$, is usually difficult to control with low computational overhead during training, and thus we turn to minimize the second factor, i.e., c_m and c'_m. Notice that c'_m in Eq. 17 is small given the robustness in the following layers after module m, which can be guaranteed by the adversarial decoupled learning according to Theorem 1. So we only need to minimize c_m, namely, improve the adversarial robustness in the auxiliary model $\boldsymbol{\theta}_m$. One can perform additional AT for the auxiliary model with perturbed feature $\boldsymbol{z}_m$, but we can find another much simpler way to attain the robustness in the auxiliary model. Since the auxiliary model is

Algorithm 1 FADE: Federated Adversarial Decoupled Learning

Initialize $\boldsymbol{w}_m^{(0)}, \boldsymbol{\theta}_m^{(0)}$ for each module m.
for $t = 1, 2, \cdots, T$ **do**
  for each client k in parallel **do**
    Randomly select a module m_k^t that will be trained in this round.
    Request current global model $\boldsymbol{w}_i^{(t)}$ for $i \leq m_k^t$ and auxiliary model $\boldsymbol{\theta}_{m_k^t}^{(t)}$ from the server.
    Generate input features $\boldsymbol{z}_{m_k^t-1}^{(t)}$ for all data $\boldsymbol{x} \in \mathbb{D}_k$.
    Perform AT in module m_k^t with $l_{m_k^t}^{\text{FADE}}$ in Eq. 19 for τ iterations, and get $\boldsymbol{\Theta}_{m_k^t,k}^{(t+1)}$.
    Upload $\boldsymbol{\Theta}_{m_k^t,k}^{(t+1)}$ to the server.
  end for
  The server aggregates $\boldsymbol{\Theta}_{m,k}^{(t+1)}$ to get $\boldsymbol{\Theta}_m^{(t+1)}$ according to Eq. 9 for each m.
end for

usually small (e.g., only a single linear layer as in our experiment), a large weight decay on it will make the auxiliary model insensitive to the small perturbation in the feature $\boldsymbol{z}_m$ [6]. Our analysis in Appendix A.3 also shows that c_m will decrease with a large weight decay hyperparameter λ_m.

Remark 3. Notice that in Theorem 2 we do not use any conditions on the difference between $\tilde{l}$ and $\tilde{l}_m$, namely, we do not require the auxiliary model to perform as well as the joint model. Thus, it is sufficient to use a small auxiliary model with weight decay to mitigate the objective inconsistency.

Accordingly, we propose to use the following adversarial locally supervised loss for each module to alleviate the objective inconsistency in adversarial decoupled learning.

$$l_m^{\text{FADE}}(\boldsymbol{z}_{m-1}^{(t)}, y; \boldsymbol{w}_m^{(t)}, \boldsymbol{\theta}_m^{(t)}) = \max_{\boldsymbol{\delta}_{m-1}^{(t)}} \left[l_m(\boldsymbol{z}_{m-1}^{(t)} + \boldsymbol{\delta}_{m-1}^{(t)}, y; \boldsymbol{w}_m^{(t)}, \boldsymbol{\theta}_m^{(t)}) \right] + \lambda_m \|\boldsymbol{\theta}_m^{(t)}\|^2. \tag{19}$$

Replacing the original loss function l_m in Eq. 7 with this improved adversarial locally supervised loss l_m^{FADE}, we get our Federated Adversarial Decoupled learning (FADE) framework, which is summarized in Algorithm 1.

Trade-off Between Objective Consistency and Robustness As we mentioned in Section 3.2, a larger μ_m, i.e., the smallest eigenvalue of the Hessian of $\tilde{l}_m(\boldsymbol{z}_m, y)$ in $\boldsymbol{z}_m$, will introduce stronger robustness. However, we decrease the eigenvalue of the Hessian when we apply large weight decay to mitigate the objective inconsistency (See Appendix A.3 for more details). Therefore, the value of λ_m balances the natural accuracy (objective consistency) and the adversarial accuracy (robustness).

3.4 Convergence Analysis

In this section, we analyze the convergence property of Federated Decoupled Learning. Following Belilovsky et al. [3], at communication round t, let $(\boldsymbol{z}_{m-1}^{(t)}, y)$ follow the distribution with probability density $p_{m-1}^{(t)}(\mathbf{z})$, and we denote its converged density as $p_{m-1}^*(\mathbf{z})$. With these notations, we have

$$L_m^{(t)}(\boldsymbol{\Theta}_m^{(t)}) = \mathbb{E}_{(\boldsymbol{z}_{m-1}^{(t)}, y) \sim p_{m-1}^{(t)}} \left[l_m(\boldsymbol{z}_{m-1}^{(t)}, y; \boldsymbol{\Theta}_m^{(t)}) \right]; \tag{20}$$

$$L_{m,k}^{(t)}(\boldsymbol{\Theta}_{m,k}^{(t)}) = \mathbb{E}_{(\boldsymbol{z}_{m-1,k}^{(t)}, y) \sim p_{m-1,k}^{(t)}} \left[l_m(\boldsymbol{z}_{m-1,k}^{(t)}, y; \boldsymbol{\Theta}_{m,k}^{(t)}) \right]; \tag{21}$$

$$L_m(\boldsymbol{\Theta}_m^{(t)}) = \mathbb{E}_{(\boldsymbol{z}_{m-1}, y) \sim p_{m-1}^*} \left[l_m(\boldsymbol{z}_{m-1}, y; \boldsymbol{\Theta}_m^{(t)}) \right], \tag{22}$$

where $p_{m-1,k}^{(t)}$ is the distribution of $(\boldsymbol{z}_{m-1,k}^{(t)}, y)$ on client k at the t-th communication round, and L_m is the global loss taking expectation over the converged density. We also use the following distance [3] between the current density and the converged density for our analysis:

$$\rho_{m-1}^{(t)} \triangleq \int \left| p_{m-1}^{(t)}(\mathbf{z}) - p_{m-1}^*(\mathbf{z}) \right| d\mathbf{z}. \tag{23}$$

We will discuss the convergence of L_m for each module m. Following Belilovsky et al. [3] and Wang et al. [33], we make the following assumptions.

Assumption 1 ($\mathcal{L}$-smoothness [3, 33])**.** *L_m is differentiable and its gradient is $\mathcal{L}_m$-Lipschitz. Similarly, $L_{m,k}^{(t)}$ is differentiable and its gradient is $\widetilde{\mathcal{L}}_m$-Lipschitz for all t and k.*

Assumption 2 (Robbins-Monro conditions [3])**.** *The learning rates satisfy $\sum_{t=0}^{\infty}\eta_t = \infty$ yet $\sum_{t=0}^{\infty}\eta_t^2 < \infty$.*

Assumption 3 (Finite variance [3, 33])**.** *There exists some positive constant $G > 0$ such that $\forall t, k, \mathbf{\Theta}_m$, $\mathbb{E}_{(\boldsymbol{z}_{m-1,k}^{(t)}, y)\sim p_{m-1,k}^{(t)}}\left[\left\|\nabla l_m\left(\boldsymbol{z}_{m-1,k}^{(t)}, y; \mathbf{\Theta}_m\right)\right\|^2\right] \leq G$.*

Assumption 4 (Bounded Dissimilarity [33])**.** *There exist constants $\beta^2 \geq 1, \kappa^2 \geq 0$ such that $\forall t, \mathbf{\Theta}_m$, $\sum_{k=1}^{m} p_k \left\|\nabla L_{m,k}^{(t)}(\mathbf{\Theta}_m)\right\|^2 \leq \beta^2 \left\|\sum_{k=1}^{m} p_k \nabla L_{m,k}^{(t)}(\mathbf{\Theta}_m)\right\|^2 + \kappa^2$.*

Assumption 5 (Convergence of the previous layer [3])**.** *We assume that $\sum_{t=0}^{\infty}\rho_{m-1}^{(t)} < \infty$.*

With all above assumptions, we get the following theorem that guarantees the convergence of Federated Decoupled Learning, and we prove it in Appendix B.

Theorem 3. *Under Assumption 1- 5, Federated Decoupled Learning converges as follows:*

$$\inf_{t\leq T}\mathbb{E}\left[\left\|\nabla L_m\left(\mathbf{\Theta}_m^{(t)}\right)\right\|^2\right] \leq \mathcal{O}\left(\frac{1}{\sum_{t=0}^{T}\eta_t}\right) + \mathcal{O}\left(\frac{\sum_{t=0}^{T}\rho_{m-1}^{(t)}\eta_t}{\sum_{t=0}^{T}\eta_t}\right) + \mathcal{O}\left(\frac{\sum_{t=0}^{T}\eta_t^2}{\sum_{t=0}^{T}\eta_t}\right) \tag{24}$$

Remark 4. Theorem 3 guarantees the convergence of locally supervised loss L_m in each module. However, because of the existence of the objective inconsistency that we mentioned in the last section, we cannot guarantee the convergence of the joint loss L with this result. Our objective inconsistency alleviation method in Section 3.3 can reduce the gap between ∇L_m and ∇L such that we can make the joint loss gradient ∇L closer to 0 when the locally supervised loss converges.

4 Experimental Results

We conduct our experiments on two datasets, FMNIST [36] and CIFAR-10 [13]. To simulate the heterogeneity in cross-device FL, we adopt a challenging experimental FL setting with $N = 1000$ clients, and we use the Non-IID data partition in [22] where each client only has the data of two labels. For the experiments on FMNIST, we sample $C = 10$ clients for local training in each communication round. For CIFAR-10, we sample $C = 30$ clients in each communication round.

For AT, we use l_∞ norm to bound the perturbation. We use PGD-10 to generate adversarial samples during training, and we test the adversarial accuracy under PGD-20 attack [20, 23, 39]. The perturbation bound at the input $\boldsymbol{z}_0 = \boldsymbol{x}$ is set to be $\epsilon_0 = 0.15$ for FMNIST, with PGD step size $\alpha_0 = 0.02$. For CIFAR-10, we set $\epsilon_0 = 8/255$ and $\alpha_0 = 2/255$ [39]. See Appendix C for more details.

4.1 Model and Module Partition

For FMNIST, we adopt a 7-layer CNN (CNN-7) with five convolutional layers and two fully connected layers. For CIFAR-10, we adopt VGG-11 [28]. Considering the amount of computation and the number of parameters, we decouple CNN-7 and VGG-11 into modules as follows:

CNN-7: We decouple the whole network into two modules. The first module [CNN-7 (1/2)] contains the first three layers and an auxiliary linear output layer, and the second module [CNN-7 (2/2)] contains the rest four layers of CNN-7.

VGG-11: We decouple the whole network in two ways, with two modules and three modules respectively. For the two-module partition, the first module [VGG-11 (1/2)] contains the first six layers with an auxiliary linear output layer, and the second module [VGG-11 (2/2)] contains the rest five layers of VGG-11. For the three-module partition, the first module [VGG-11 (1/3)] contains the first four layers with an auxiliary linear output layer, the second module [VGG-11 (2/3)] contains the fifth and the sixth layer with an auxiliary linear output layer, and the third module [VGG-11 (3/3)] contains the rest five layers of VGG-11.

Table 2: The natural accuracy (clean data) and adversarial accuracy (adversarial data) of different training scheme. Results are reported in the average and standard deviation over 3 random seeds.

Training Scheme	FMNIST		CIFAR-10	
	Natural Acc.	Adversarial Acc.	Natural Acc.	Adversarial Acc.
Fully Joint (100% AT + 0% ST)	$83.05 \pm 0.41\%$	$70.44 \pm 0.73\%$	$64.99 \pm 1.60\%$	$28.66 \pm 1.73\%$
Fully Joint (20% AT + 80% ST)	$85.65 \pm 0.08\%$	$56.45 \pm 0.70\%$	$74.51 \pm 0.30\%$	$19.24 \pm 0.19\%$
FADE (100% 2-Module)	$81.17 \pm 1.58\%$	$65.19 \pm 1.90\%$	$66.58 \pm 0.73\%$	$27.15 \pm 0.55\%$
FADE (100% 3-Module)	-	-	$64.68 \pm 0.60\%$	$26.93 \pm 0.41\%$
FADE (20% Joint + 80% 2-Module)	$80.47 \pm 1.48\%$	$70.35 \pm 0.97\%$	$64.92 \pm 0.66\%$	$27.95 \pm 0.70\%$
FADE (20% Joint + 80% 3-Module)	-	-	$64.52 \pm 0.85\%$	$28.10 \pm 0.47\%$
FADE (20% Joint + 30% 2-Module + 50% 3-Module)	-	-	$64.07 \pm 1.09\%$	$28.80 \pm 0.78\%$

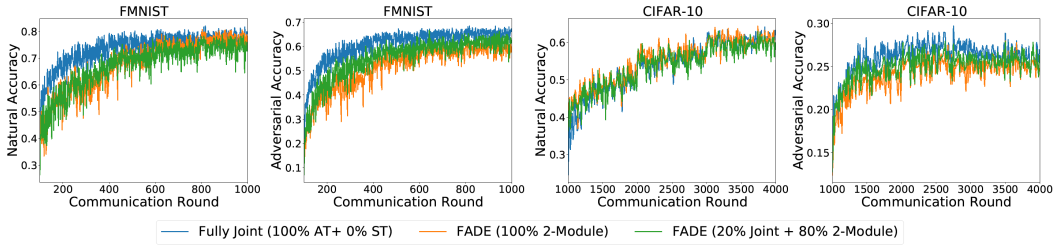

Figure 4: Learning curves of different training schemes. Following Zizzo et al. [39], we begin with a warm-up phase with only standard training and then turn to adversarial training. The length of the warm-up phase is 100 rounds for FMNIST and 1000 rounds for CIFAR-10.

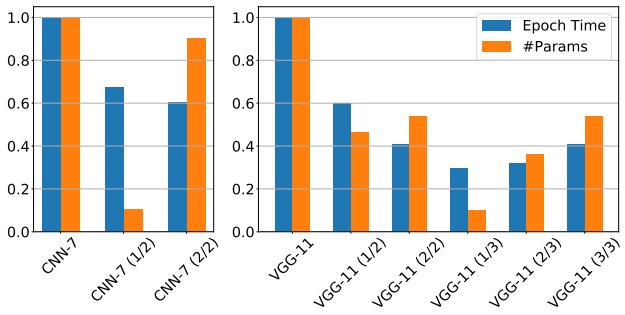

Figure 3: Profiles of different modules. The results are shown as the percentage of the value in joint training.

The local epoch time and the number of parameters of each module are shown in Fig. 3. Comparing with joint training, we can reduce the epoch time by around 40% in both CNN-7 and VGG-11 when decoupling the whole model into two modules, and we can reduce the maximal number of parameters by around 50% in VGG-11. When applying 3-module partitioning, the epoch time of VGG-11 can be further reduced to less than 40% of the joint training. Although the number of parameters are concentrated in the second module of CNN-7, the memory usage for storing the parameters is not the main bottleneck in this case since the total number of parameters in CNN-7 is small enough (around 0.6M).

4.2 Performance of FADE

In this section, we test the performance of FADE in different training schemes. Our baseline is the fully joint training, which is only feasible in the ideal scenario where all the clients have enough resources to complete AT on the whole neural network. We first test FADE in the worst case where no clients can afford joint AT but can only conduct AT on a module ("FADE (100% 2/3-Module)"). Then we turn to a realistic scenario where different clients have different resources, and 20% of them can afford joint AT while the others can complete AT on a module ("FADE (20% Joint + 80% 2/3-Module)"). We set $\lambda_m = 0.001$ for FMNIST and $\lambda_m = 0.003$ for CIFAR-10.

As shown in Table 2, even in the worst case, FADE can attain a much stronger robustness than the partial federated adversarial training (20% AT + 80% ST). When we allow 20% clients to perform joint AT, FADE can almost recover the adversarial accuracy of the fully joint training, using fewer computing resources on small edge devices. The learning curves of joint training and 2-Module FADE are also shown in Fig. 4, while the learning curves for 3-Module FADE are given in Appendix C. We can see that FADE can converge to the same natural accuracy as that in joint training, with only marginal adversarial accuracy drop comparing with joint training.

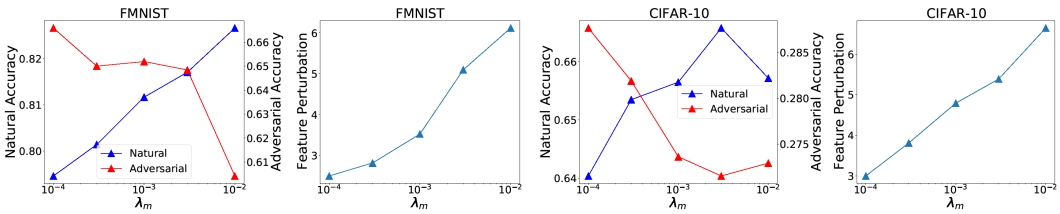

Figure 5: Performance and feature perturbation with different auxiliary model weight decay.

Furthermore, We mix three kinds of clients (joint training, 2-module training, 3-module training) together to verify the scalability of FADE. The result "FADE (20% Joint + 30% 2-Module + 50% 3-Module)" in Table 2 shows that different clients with different module partitions are compatible, and they are able to collaborate in FADE to achieve strong robustness.

4.3 The Influence of Auxiliary Model Weight Decay

As we suggested in Section 3.3, the auxiliary model weight decay hyperparameter λ_m acts as an important role for balancing the natural accuracy and the adversarial robustness. To show this trade-off, we conduct experiments with 2-Module FADE and test the natural accuracy and the adversarial accuracy with different $\lambda_m \in \{0.0001, 0.0003, 0.001, 0.003, 0.01\}$. We also measure the perturbation on the feature outputed by the first module, namely, $\|f_1(\boldsymbol{x}+\boldsymbol{\delta}_0) - f_1(\boldsymbol{x})\|_2$, to show the influence of λ_m on the lower bound of ϵ_1 for guaranteed robustness. The results are shown in Fig. 5.

We can observe the consistency between the experimental results and our theory that a larger λ_m can lead to higher natural accuracy while decreasing the adversarial accuracy. When applying a larger λ_m to mitigate the objective inconsistency, the perturbation on the feature also increases, which leads to a weaker overall adversarial robustness in the backbone network.

It is noteworthy that when applying a too large λ_m, the auxiliary model may fail to learn meaningful representation and the natural accuracy will drop, as shown in the experiment on CIFAR-10 with $\lambda_m = 0.01$. This result also aligns with the well-known fact in normal training that a too large regularization term can undermine performance of a model.

5 Related Works

Federated Learning In recent years, Federated Learning (FL) was proposed as a solution to trustworthy and communication-efficient distributed learning [11, 12, 22]. Although FL attains high privacy by storing data locally and pushing training to the edge devices, it also suffers from slow and unstable convergence caused by the heterogeneity in both local data (statistical heterogeneity) and devices (systematic heterogeneity) [9, 15, 16, 33]. Many studies have tried to overcome the statistical heterogeneity [10, 18, 31, 32] and the systematic heterogeneity [14, 16, 33]. Beyond the heterogeneity, FL is also vulnerable in several kinds of attack, such as model poisoning attack [4, 29] and adversarial attack [39]. In this paper, we mainly focus on the adversarial attack and try to deal with the challenge in federated adversarial training caused by client-wise heterogeneity [8, 27].

Adversarial Training Adversarial Training (AT) has been shown to be a effective method for attaining robustness against adversarial samples [6, 20]. However, to achieve strong robustness, AT requires PGD with multiple iterations [5], which consumes large amounts of computational resources. Recent studies have explored the possibility of reducing the computation in AT [25, 38], such as replacing PGD with FGSM [1, 35] or using other regularization methods for robustness [23, 24]. These methods have been developed for centralized AT but they can additionally be deployed in FL together with FADE to further reduce the computational cost for small edge devices.

Decoupled Greedy Learning As deeper and deeper neural networks are used for better performance, the low efficiency of end-to-end (joint) training is exposed because it hinders the model parallelization and requires large memory for model parameters and intermediate results [3, 7]. As an alternative, Decoupled Greedy Learning (DGL) is proposed, which decouples the whole neural network into several modules and trains them separately without gradient dependency [2, 3, 21, 34].

In this paper, we deploy and improve DGL for federated adversarial training such that edge-devices with few computing resources could also participate in AT for large models.

6 Conclusions

In this paper, we propose Federated Adversarial Decoupled Learning (FADE), a novel framework to reduce the computing resource requirement for small edge devices in large-scale federated adversarial training. We amend vanilla Decoupled Greedy Learning (DGL) such that we can guarantee the adversarial robustness of the whole model and alleviate the objective inconsistency when deploying it in Federated Learning (FL). Our experimental results show that FADE can significantly reduce the computing resource demand on small edge devices, while maintaining almost the same accuracy as fully joint training on both clean and adversarial samples. FADE is complementary to existing fast adversarial training methods (e.g., [1, 23, 35]) and can be combined with them to further accelerate federated adversarial training in the future.

References

[1] Maksym Andriushchenko and Nicolas Flammarion. Understanding and improving fast adversarial training. *Advances in Neural Information Processing Systems*, 33:16048–16059, 2020.

[2] Eugene Belilovsky, Michael Eickenberg, and Edouard Oyallon. Greedy layerwise learning can scale to imagenet. In *International conference on machine learning*, pages 583–593. PMLR, 2019.

[3] Eugene Belilovsky, Michael Eickenberg, and Edouard Oyallon. Decoupled greedy learning of cnns. In *International Conference on Machine Learning*, pages 736–745. PMLR, 2020.

[4] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In *International Conference on Machine Learning*, pages 634–643. PMLR, 2019.

[5] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning*, pages 2206–2216. PMLR, 2020.

[6] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[7] Chris Hettinger, Tanner Christensen, Ben Ehlert, Jeffrey Humpherys, Tyler Jarvis, and Sean Wade. Forward thinking: Building and training neural networks one layer at a time. *arXiv preprint arXiv:1706.02480*, 2017.

[8] Junyuan Hong, Haotao Wang, Zhangyang Wang, and Jiayu Zhou. Federated robustness propagation: Sharing adversarial robustness in federated learning. *arXiv preprint arXiv:2106.10196*, 2021.

[9] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[10] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J Reddi, Sebastian U Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for on-device federated learning. *arXiv preprint arXiv:1910.06378*, 2019.

[11] Jakub Konečnỳ, Brendan McMahan, and Daniel Ramage. Federated optimization: Distributed optimization beyond the datacenter. *arXiv preprint arXiv:1511.03575*, 2015.

[12] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

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

[14] Ang Li, Jingwei Sun, Pengcheng Li, Yu Pu, Hai Li, and Yiran Chen. Hermes: an efficient federated learning framework for heterogeneous mobile clients. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, pages 420–437, 2021.

[15] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[16] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 2018.

[17] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019.

[18] Xianfeng Liang, Shuheng Shen, Jingchang Liu, Zhen Pan, Enhong Chen, and Yifei Cheng. Variance reduced local sgd with lower communication complexity. *arXiv preprint arXiv:1912.12844*, 2019.

[19] Xiaofeng Lu, Yuying Liao, Pietro Lio, and Pan Hui. Privacy-preserving asynchronous federated learning mechanism for edge network computing. *IEEE Access*, 8:48970–48981, 2020.

[20] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[21] Enrique S Marquez, Jonathon S Hare, and Mahesan Niranjan. Deep cascade learning. *IEEE transactions on neural networks and learning systems*, 29(11):5475–5485, 2018.

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

[23] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. Robustness via curvature regularization, and vice versa. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9078–9086, 2019.

[24] Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Krishnamurthy Dvijotham, Alhussein Fawzi, Soham De, Robert Stanforth, and Pushmeet Kohli. Adversarial robustness through local linearization. *Advances in Neural Information Processing Systems*, 32, 2019.

[25] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *Advances in Neural Information Processing Systems*, 32, 2019.

[26] Ali Shafahi, Parsa Saadatpanah, Chen Zhu, Amin Ghiasi, Christoph Studer, David Jacobs, and Tom Goldstein. Adversarially robust transfer learning. *arXiv preprint arXiv:1905.08232*, 2019.

[27] Devansh Shah, Parijat Dube, Supriyo Chakraborty, and Ashish Verma. Adversarial training in communication constrained federated learning. *arXiv preprint arXiv:2103.01319*, 2021.

[28] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[29] Jingwei Sun, Ang Li, Louis DiValentin, Amin Hassanzadeh, Yiran Chen, and Hai Li. Fl-wbc: Enhancing robustness against model poisoning attacks in federated learning from a client perspective. *Advances in Neural Information Processing Systems*, 34, 2021.

[30] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[31] Minxue Tang, Xuefei Ning, Yitu Wang, Jingwei Sun, Yu Wang, Hai Li, and Yiran Chen. Fedcor: Correlation-based active client selection strategy for heterogeneous federated learning, 2021.

[32] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. *arXiv preprint arXiv:2002.06440*, 2020.

[33] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. *arXiv preprint arXiv:2007.07481*, 2020.

[34] Yulin Wang, Zanlin Ni, Shiji Song, Le Yang, and Gao Huang. Revisiting locally supervised learning: An alternative to end-to-end training. *arXiv preprint arXiv:2101.10832*, 2021.

[35] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*, 2020.

[36] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[37] Cong Xie, Sanmi Koyejo, and Indranil Gupta. Asynchronous federated optimization. *arXiv preprint arXiv:1903.03934*, 2019.

[38] Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. You only propagate once: Accelerating adversarial training via maximal principle. *Advances in Neural Information Processing Systems*, 32, 2019.

[39] Giulio Zizzo, Ambrish Rawat, Mathieu Sinn, and Beat Buesser. Fat: Federated adversarial training. *arXiv preprint arXiv:2012.01791*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] Our method may lead to marginal accuracy drop as in Section 4.
   (c) Did you discuss any potential negative societal impacts of your work? [No] Our work does not have negative social impacts.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 3.
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix A and Appendix B.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See supplementary material and Appendix C.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 4 and Appendix C.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix C.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No] We do not have new assets in this work.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Proofs and Additional Analysis

A.1 Proof of Theorem 1

Theorem 1. *Omitting all parameters $\boldsymbol{w}$ and $\boldsymbol{\theta}$, for adjacent modules m and $m+1$, given $\boldsymbol{z}_{m-1}$ and $\boldsymbol{z}_m = f_m(\boldsymbol{z}_{m-1})$, let $\tilde{l}_m(\boldsymbol{z}_m, y) = \tilde{l}_m(f_m(\boldsymbol{z}_{m-1}), y) = l_m(\boldsymbol{z}_{m-1}, y)$ and assume that $\tilde{l}_m(\boldsymbol{z}_m, y)$ is μ_m-strongly convex in $\boldsymbol{z}_m$, if we perform AT on these two modules separately such that*

$$\forall \boldsymbol{\delta}_{m-1} \in \{\boldsymbol{\delta}_{m-1} : \|\boldsymbol{\delta}_{m-1}\| \leq \epsilon_{m-1}\}, l_m(\boldsymbol{z}_{m-1} + \boldsymbol{\delta}_{m-1}, y) - l_m(\boldsymbol{z}_{m-1}, y) \leq c_m; \tag{25}$$

$$\forall \boldsymbol{\delta}_m \in \{\boldsymbol{\delta}_m : \|\boldsymbol{\delta}_m\| \leq \epsilon_m\}, l_{m+1}(\boldsymbol{z}_m + \boldsymbol{\delta}_m, y) - l_{m+1}(\boldsymbol{z}_m, y) \leq c_{m+1}, \tag{26}$$

with ϵ_m satisfying

$$\epsilon_m \geq \frac{g_m}{\mu_m} + \sqrt{\frac{2c_m}{\mu_m} + \frac{g_m^2}{\mu_m^2}}, \quad \textit{where } g_m = \|\nabla_{\boldsymbol{z}_m} \tilde{l}_m(\boldsymbol{z}_m, y)\|, \tag{27}$$

we will have the overall robustness in the joint model of these two modules:

$$\begin{aligned} &\forall \boldsymbol{\delta}_{m-1} \in \{\boldsymbol{\delta}_{m-1} : \|\boldsymbol{\delta}_{m-1}\| \leq \epsilon_{m-1}\}, \\ &l_{m+1}(f_m(\boldsymbol{z}_{m-1} + \boldsymbol{\delta}_{m-1}), y) - l_{m+1}(f_m(\boldsymbol{z}_{m-1}), y) \leq c_{m+1}. \end{aligned} \tag{28}$$

Proof. Given μ_m-strongly convexity of $\tilde{l}_m(\boldsymbol{z}_m, y)$, for any $\boldsymbol{\delta}_{m-1} \in \{\boldsymbol{\delta}_{m-1} : \|\boldsymbol{\delta}_{m-1}\| \leq \epsilon_{m-1}\}$, let $\boldsymbol{r} = f_m(\boldsymbol{z}_{m-1} + \boldsymbol{\delta}_{m-1}) - f_m(\boldsymbol{z}_{m-1})$, we have

$$\nabla_{\boldsymbol{z}_m} \tilde{l}_m(\boldsymbol{z}_m, y)^T \boldsymbol{r} + \frac{\mu_m}{2}\|r\|^2 \leq \tilde{l}_m(\boldsymbol{z}_m + \boldsymbol{r}, y) - \tilde{l}_m(\boldsymbol{z}_m, y) \leq c_m \tag{29}$$

$$\Rightarrow \frac{\mu_m}{2}\left[\left\|\boldsymbol{r} + \frac{\nabla_{\boldsymbol{z}_m} \tilde{l}_m(\boldsymbol{z}_m, y)}{\mu_m}\right\|^2 - \frac{\|\nabla_{\boldsymbol{z}_m} \tilde{l}_m(\boldsymbol{z}_m, y)\|^2}{\mu_m^2}\right] \leq c_m \tag{30}$$

$$\Rightarrow \left\|\boldsymbol{r} + \frac{\nabla_{\boldsymbol{z}_m} \tilde{l}_m(\boldsymbol{z}_m, y)}{\mu_m}\right\| \leq \sqrt{\frac{2c_m}{\mu_m} + \frac{\|\nabla_{\boldsymbol{z}_m} \tilde{l}_m(\boldsymbol{z}_m, y)\|^2}{\mu_m^2}} \tag{31}$$

$$\Rightarrow \|\boldsymbol{r}\| \leq \frac{\|\nabla_{\boldsymbol{z}_m} \tilde{l}_m(\boldsymbol{z}_m, y)\|}{\mu_m} + \sqrt{\frac{2c_m}{\mu_m} + \frac{\|\nabla_{\boldsymbol{z}_m} \tilde{l}_m(\boldsymbol{z}_m, y)\|^2}{\mu_m^2}} = \frac{g_m}{\mu_m} + \sqrt{\frac{2c_m}{\mu_m} + \frac{g_m^2}{\mu_m^2}}. \tag{32}$$

And we know that

$$\epsilon_m \geq \frac{g_m}{\mu_m} + \sqrt{\frac{2c_m}{\mu_m} + \frac{g_m^2}{\mu_m^2}} \geq \|\boldsymbol{r}\|, \tag{33}$$

which gives us

$$\forall \boldsymbol{\delta}_{m-1} \in \{\boldsymbol{\delta}_{m-1} : \|\boldsymbol{\delta}_{m-1}\| \leq \epsilon_{m-1}\}, \|f_m(\boldsymbol{z}_{m-1} + \boldsymbol{\delta}_{m-1}) - f_m(\boldsymbol{z}_{m-1})\| \leq \epsilon_m. \tag{34}$$

And with Eq. 26, we get the robustness guarantee:

$$\begin{aligned} &\forall \boldsymbol{\delta}_{m-1} \in \{\boldsymbol{\delta}_{m-1} : \|\boldsymbol{\delta}_{m-1}\| \leq \epsilon_{m-1}\}, \\ &l_{m+1}(f_m(\boldsymbol{z}_{m-1} + \boldsymbol{\delta}_{m-1}), y) - l_{m+1}(f_m(\boldsymbol{z}_{m-1}), y) \leq c_{m+1}. \end{aligned} \tag{35}$$

□

A.2 Proof of Theorem 2

Theorem 2. *Omitting all parameters $\boldsymbol{w}$ and $\boldsymbol{\theta}$, given $\boldsymbol{z}_m = f_m(\boldsymbol{z}_{m-1}) = f_m \circ f_{m-1} \circ \cdots \circ f_1(\boldsymbol{x})$, let $\tilde{l}_m(\boldsymbol{z}_m, y) = \tilde{l}_m(f_m(\boldsymbol{z}_{m-1}), y) = l_m(\boldsymbol{z}_{m-1}, y)$ and $\tilde{l}(\boldsymbol{z}_m, y) = \tilde{l}(f_m \circ \cdots \circ f_1(\boldsymbol{x}), y) = l(\boldsymbol{x}, y)$, assume $\tilde{l}_m(\boldsymbol{z}_m, y)$ and $\tilde{l}(\boldsymbol{z}_m, y)$ are β_m, β'_m-smooth in $\boldsymbol{z}_m$, if there exist c_m and c'_m such that*

$$\exists r \geq \sqrt{2\frac{c_m + c'_m}{\beta_m + \beta'_m}}, \forall \boldsymbol{\delta}_m \in \{\boldsymbol{\delta}_m : \|\boldsymbol{\delta}_m\| \leq r\},$$

$$|\tilde{l}_m(\boldsymbol{z}_m + \boldsymbol{\delta}_m, y) - \tilde{l}_m(\boldsymbol{z}_m, y)| \leq c_m, \tag{36}$$

$$|\tilde{l}(\boldsymbol{z}_m + \boldsymbol{\delta}_m, y) - \tilde{l}(\boldsymbol{z}_m, y)| \leq c'_m, \tag{37}$$

then we have

$$\|\nabla_{\boldsymbol{w}_m} l - \nabla_{\boldsymbol{w}_m} l_m\| \leq \left\| \frac{\partial \boldsymbol{z}_m}{\partial \boldsymbol{w}_m} \right\| \sqrt{2(c_m + c'_m)(\beta_m + \beta'_m)}. \tag{38}$$

Proof. With the chain rule, we know that

$$\nabla_{\boldsymbol{w}_m} l - \nabla_{\boldsymbol{w}_m} l_m = \frac{\partial \boldsymbol{z}_m}{\partial \boldsymbol{w}_m} \frac{\partial (l - l_m)}{\partial \boldsymbol{z}_m} = \frac{\partial \boldsymbol{z}_m}{\partial \boldsymbol{w}_m} \frac{\partial (\tilde{l} - \tilde{l}_m)}{\partial \boldsymbol{z}_m}, \tag{39}$$

and thus

$$\|\nabla_{\boldsymbol{w}_m} l - \nabla_{\boldsymbol{w}_m} l_m\| \leq \left\| \frac{\partial \boldsymbol{z}_m}{\partial \boldsymbol{w}_m} \right\| \left\| \frac{\partial (\tilde{l} - \tilde{l}_m)}{\partial \boldsymbol{z}_m} \right\|. \tag{40}$$

We now need to find the upper bound of the second factor. We define $h(\boldsymbol{z}_m) = \tilde{l}(\boldsymbol{z}_m, y) - \tilde{l}_m(\boldsymbol{z}_m, y)$, which is $(\beta_m + \beta'_m)$-smooth in $\boldsymbol{z}_m$. For any $\boldsymbol{\delta}_m, \|\boldsymbol{\delta}_m\| \leq r$, we have

$$|h(\boldsymbol{z}_m + \boldsymbol{\delta}_m) - h(\boldsymbol{z}_m)| \leq |\tilde{l}_m(\boldsymbol{z}_m + \boldsymbol{\delta}_m, y) - \tilde{l}_m(\boldsymbol{z}_m, y)| + |\tilde{l}(\boldsymbol{z}_m + \boldsymbol{\delta}_m, y) - \tilde{l}(\boldsymbol{z}_m, y)| \tag{41}$$

$$\leq c_m + c'_m. \tag{42}$$

And with the $(\beta_m + \beta'_m)$-smoothness, we know that

$$\left(\frac{\partial h(\boldsymbol{z}_m)}{\partial \boldsymbol{z}_m} \right)^T \boldsymbol{\delta}_m - \frac{\beta_m + \beta'_m}{2} \|\boldsymbol{\delta}_m\|^2 \leq h(\boldsymbol{z}_m + \boldsymbol{\delta}_m) - h(\boldsymbol{z}_m) \leq c_m + c'_m. \tag{43}$$

The maximum of the LHS is achieved when $\boldsymbol{\delta}_m^* = \frac{1}{\beta_m + \beta'_m} \frac{\partial h(\boldsymbol{z}_m)}{\partial \boldsymbol{z}_m}$, and thus we get

$$\frac{1}{2(\beta_m + \beta'_m)} \left\| \frac{\partial h(\boldsymbol{z}_m)}{\partial \boldsymbol{z}_m} \right\|^2 \leq c_m + c'_m \tag{44}$$

$$\Rightarrow \left\| \frac{\partial h(\boldsymbol{z}_m)}{\partial \boldsymbol{z}_m} \right\| \leq \sqrt{2(c_m + c'_m)(\beta_m + \beta'_m)}. \tag{45}$$

To check the achievability of this maximum, we have

$$\|\boldsymbol{\delta}_m^*\| = \frac{1}{\beta_m + \beta'_m} \left\| \frac{\partial h(\boldsymbol{z}_m)}{\partial \boldsymbol{z}_m} \right\| \leq \sqrt{2 \frac{c_m + c'_m}{\beta_m + \beta'_m}} \leq r. \tag{46}$$

Thus, we get our final result

$$\|\nabla_{\boldsymbol{w}_m} l - \nabla_{\boldsymbol{w}_m} l_m\| \leq \left\| \frac{\partial \boldsymbol{z}_m}{\partial \boldsymbol{w}_m} \right\| \sqrt{2(c_m + c'_m)(\beta_m + \beta'_m)}. \tag{47}$$

□

A.3 Case Study: Linear Auxiliary Output Model

For a linear auxiliary output model $\boldsymbol{\theta}_m = \{\boldsymbol{W}_m, \boldsymbol{b}_m\}$, the cross-entropy loss is given as

$$\tilde{l}(\boldsymbol{z}_m, \boldsymbol{y}) = \mathcal{L}(\sigma(\boldsymbol{W}_m^T \boldsymbol{z}_m + \boldsymbol{b}_m), \boldsymbol{y}), \tag{48}$$

where $\mathcal{L}(\boldsymbol{p}, \boldsymbol{y}) = -\sum_{i=1} y_i \log(p_i)$ and $\sigma(\boldsymbol{q})_i = \exp(q_i)/(\sum_j \exp(q_j))$ are cross-entropy loss and softmax function respectively. Let $\boldsymbol{p}_m = \sigma(\boldsymbol{W}_m^T \boldsymbol{z}_m + \boldsymbol{b}_m)$, we know that

$$\nabla_{\boldsymbol{z}_m} \tilde{l}(\boldsymbol{z}_m, \boldsymbol{y}) = \boldsymbol{W}_m (\boldsymbol{p}_m - \boldsymbol{y}), \tag{49}$$

and

$$\boldsymbol{H}_m = \nabla^2_{\boldsymbol{z}_m} \tilde{l}(\boldsymbol{z}_m, \boldsymbol{y}) = \boldsymbol{W}_m \boldsymbol{J}_m \boldsymbol{W}_m^T, \tag{50}$$

where

$$\boldsymbol{J}_m = \text{diag}(\boldsymbol{p}_m) - \boldsymbol{p}_m \boldsymbol{p}_m^T \tag{51}$$

is the Jacobi of the softmax function. We have the following properties related to the robustness and objective consistency in Theorem 1 and Theorem 2:

1. (First Order Property) Smaller $\|\boldsymbol{W}_m\|$ leads to smaller g_m and c_m.

$$g_m = \|\nabla_{\boldsymbol{z}_m} \tilde{l}_m(\boldsymbol{z}_m, y)\| = \|\boldsymbol{W}_m(\boldsymbol{p}_m - \boldsymbol{y})\| \leq \sqrt{2}\|\boldsymbol{W}_m\|, \tag{52}$$

$$c_m = \max_{\|\boldsymbol{\delta}_m\| \leq r} |\tilde{l}_m(\boldsymbol{z}_m + \boldsymbol{\delta}_m, y) - \tilde{l}_m(\boldsymbol{z}_m, y)| \leq \sqrt{2}r\|\boldsymbol{W}_m\|. \tag{53}$$

2. (Second Order Property) Smaller $\|\boldsymbol{W}_m\|$ leads to smaller μ_m and β_m.

$$\begin{aligned}\sum_i \lambda_i(\boldsymbol{H}_m) = \text{tr}(\boldsymbol{H}_m) = \text{tr}(\boldsymbol{W}_m \boldsymbol{J}_m \boldsymbol{W}_m^T) = \text{tr}(\boldsymbol{W}_m^T \boldsymbol{W}_m \boldsymbol{J}_m) \\ \leq \|\boldsymbol{W}_m\|_F^2 (\sum_j p_{m,j} - p_{m,j}^2),\end{aligned} \tag{54}$$

where $\lambda_i(\boldsymbol{H}_m)$ means the eigenvalues of $\boldsymbol{H}_m$ in increasing order. $\lambda_1(\boldsymbol{H}_m) = \mu_m$ and $\lambda_{-1}(\boldsymbol{H}_m) = \beta_m$.

We notice that when increasing λ_m, namely, decreasing $\|\boldsymbol{W}_m\|_F^2$, we will decrease g_m, c_m, μ_m and β_m. According to Theorem 1 and 2, smaller g_m, c_m and β_m can lead to stronger robustness as well as smaller objective inconsistency, while smaller μ_m will lead to weaker robustness as shown in Eq. 27.

B Convergence

Theorem 3. *Under Assumption 1- 5, Federated Decoupled Learning converges as follows:*

$$\inf_{t \leq T} \mathbb{E}\left[\left\|\nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right)\right\|^2\right] \leq \mathcal{O}\left(\frac{1}{\sum_{t=0}^T \eta_t}\right) + \mathcal{O}\left(\frac{\sum_{t=0}^T \rho_{m-1}^{(t)} \eta_t}{\sum_{t=0}^T \eta_t}\right) + \mathcal{O}\left(\frac{\sum_{t=0}^T \eta_t^2}{\sum_{t=0}^T \eta_t}\right). \tag{55}$$

Proof. We consider the SGD scheme with learning rate $\{\eta_t\}_t$:

$$\boldsymbol{\Theta}_m^{(t+1)} = \boldsymbol{\Theta}_m^{(t)} - \eta_t \frac{\sum_{k \in \mathbb{S}_m^{(t)}} p_k \boldsymbol{h}_{m,k}^{(t)}}{\sum_{k \in \mathbb{S}_m^{(t)}} p_k}.$$

The size of $\mathbb{S}_m^{(t)}$ is S_t and $\boldsymbol{h}_{m,k}^{(t)}$ is defined as $\boldsymbol{h}_{m,k}^{(t)} = \sum_{j=0}^{\tau-1} \nabla l_m(\boldsymbol{z}_{m-1,k}^{(t,j)}, y; \boldsymbol{w}_{m,k}^{(t,j)}, \boldsymbol{\theta}_{m,k}^{(t,j)}) = \sum_{j=0}^{\tau-1} \nabla l_m(\boldsymbol{z}_{m-1,k}^{(t,j)}, y; \Theta_{m,k}^{(t,j)})$. According to the Lipschitz-smooth assumption for the global objective function L_m , it follows that

$$\begin{aligned}&\mathbb{E}\left[L_m\left(\boldsymbol{\Theta}_m^{(t+1)}\right)\right] - L_m\left(\boldsymbol{\Theta}_m^{(t)}\right) \\ \leq &-\eta_t \mathbb{E}\underbrace{\left[\left\langle \nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right), \frac{\sum_{k \in \mathbb{S}_l^{(t)}} p_k \boldsymbol{h}_{m,k}^{(t)}}{\sum_{k \in \mathbb{S}_l^{(t)}} p_k} \right\rangle\right]}_{T_1} + \frac{\eta_t^2 \mathcal{L}_m}{2} \mathbb{E}\underbrace{\left[\left\| \frac{\sum_{k \in \mathbb{S}_l^{(t)}} p_k \boldsymbol{h}_{m,k}^{(t)}}{\sum_{k \in \mathbb{S}_l^{(t)}} p_k} \right\|^2\right]}_{T_2}.\end{aligned} \tag{56}$$

Similar to the proof in [33], to bound the T_1 in Eq. 56, we should notice that

$$
\begin{aligned}
T_1 =& \mathbb{E}\left[\left\langle \nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right), \frac{\sum_{k\in\mathbb{S}_l^{(t)}} p_k \boldsymbol{h}_{m,k}^{(t)}}{\sum_{k\in\mathbb{S}_l^{(t)}} p_k}\right\rangle\right] \\
=& \left\langle \nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right), \sum_{k=1}^{N} p_k \mathbb{E}\boldsymbol{h}_{m,k}^{(t)}\right\rangle \\
=& \frac{1}{2}\left\|\nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right)\right\|^2 + \frac{1}{2}\left\|\sum_{k=1}^{N} p_k \mathbb{E}\boldsymbol{h}_{m,k}^{(t)}\right\|^2 - \frac{1}{2}\left\|\nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right) - \sum_{k=1}^{N} p_k \mathbb{E}\boldsymbol{h}_{m,k}^{(t)}\right\|^2 \qquad (57) \\
\geq& \frac{1}{2}\left\|\nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right)\right\|^2 - \frac{1}{2}\left\|\nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right) - \sum_{k=1}^{N} p_k \mathbb{E}\boldsymbol{h}_{m,k}^{(t)}\right\|^2 \\
\geq& \frac{1}{2}\left\|\nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right)\right\|^2 - \left\|\nabla L_m^{(t)}\left(\boldsymbol{\Theta}_m^{(t)}\right) - \sum_{k=1}^{N} p_k \mathbb{E}\boldsymbol{h}_{m,k}^{(t)}\right\|^2 \\
& - \left\|\nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right) - \nabla L_m^{(t)}(\boldsymbol{\Theta}_m^{(t)})\right\|^2 \qquad (58) \\
\geq& \frac{1}{2}\left\|\nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right)\right\|^2 - \sum_{k=1}^{N} p_k \left\|\mathbb{E}\boldsymbol{h}_{m,k}^{(t)} - \nabla L_{m,k}^{(t)}(\boldsymbol{\Theta}_m^{(t)})\right\|^2 \\
& - \left\|\nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right) - \nabla L_m^{(t)}(\boldsymbol{\Theta}_m^{(t)})\right\|^2. \qquad (59)
\end{aligned}
$$

Eq. 57 uses the fact: $2\langle a, b\rangle = \|a\|^2 + \|b\|^2 - \|a-b\|^2$. Eq. 58 uses the fact: $\|a+b\|^2 \leq 2\|a\|^2 + 2\|b\|^2$. Eq. 59 uses the fact: $L_m^{(t)} = \sum_{k=1}^{N} p_k L_{m,k}^{(t)}$ and Jenson's inequality $\left\|\sum_{i=1}^{m} b_i a_i\right\|^2 \leq \sum_{i=1}^{m} b_i \left\|a_i\right\|^2$.

We denote $(\boldsymbol{z}, y)$ as $\mathbf{z}$. Based on the proof of Lemma 3.2 in [3], we have

$$
\begin{aligned}
& \left\|\nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right) - \nabla L_m^{(t)}\left(\boldsymbol{\Theta}_m^{(t)}\right)\right\|^2 \\
=& \left\|\mathbb{E}_{(\boldsymbol{z},y)\sim p_{m-1}^{(t)}}\left[\nabla l_m(\boldsymbol{z}_{m-1}^{(t)}, y; \boldsymbol{\Theta}_m^{(t)})\right] - \mathbb{E}_{(\boldsymbol{z},y)\sim p_{m-1}^{*}}\left[\nabla l_m(\boldsymbol{z}^{(t)}, y; \boldsymbol{\Theta}_m^{(t)})\right]\right\|^2 \\
=& \left\|\int \nabla l_m(\mathbf{z}; \boldsymbol{\Theta}_m^{(t)}) p_{m-1}^{(t)}(\mathbf{z}) d\mathbf{z} - \int \nabla l_m(\mathbf{z}; \boldsymbol{\Theta}_m^{(t)}) p_{m-1}^{*}(\mathbf{z}) d\mathbf{z}\right\|^2 \\
\leq& \left(\int \|\nabla l_m(\mathbf{z}; \boldsymbol{\Theta}_m^{(t)})\| \sqrt{|p_{m-1}^{(t)}(\mathbf{z}) - p_{m-1}^{*}(\mathbf{z})| |p_{m-1}^{(t)}(\mathbf{z}) - p_{m-1}^{*}(\mathbf{z})|} d\mathbf{z}\right)^2.
\end{aligned}
$$

According to the Cauchy-Schwartz inequality, we have

$$
\begin{aligned}
& \left\|\nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right) - \nabla L_m^{(t)}\left(\boldsymbol{\Theta}_m^{(t)}\right)\right\|^2 \\
\leq& \int \|\nabla l_m(\mathbf{z}; \boldsymbol{\Theta}_m^{(t)})\|^2 |p_{m-1}^{(t)} - p_{m-1}^{*}| d\mathbf{z} \int |p_{m-1}^{(t)} - p_{m-1}^{*}| d\mathbf{z} \\
=& \rho_m^{(t)} \int \|\nabla l_m(\mathbf{z}; \boldsymbol{\Theta}_m^{(t)})\|^2 |p_{m-1}^{(t)} - p_{m-1}^{*}| d\mathbf{z} \\
\leq& \rho_m^{(t)} \int \|\nabla l_m(\mathbf{z}; \boldsymbol{\Theta}_m^{(t)})\|^2 \left(p_{m-1}^{(t)} + p_{m-1}^{*}\right) d\mathbf{z} \\
\leq& \rho_m^{(t)} \left[\mathbb{E}_{(\boldsymbol{z},y)\sim p_{m-1}^{(t)}} \left\|\nabla l_m(\boldsymbol{z}_{m-1}^{(t)}, y; \boldsymbol{\Theta}_m^{(t)})\right\|^2 + \mathbb{E}_{(\boldsymbol{z},y)\sim p_{m-1}^{*}} \left\|\nabla l_m(\boldsymbol{z}^{(t)}, y; \boldsymbol{\Theta}_m^{(t)})\right\|^2\right] \\
\leq& 2G\rho_m^{(t)}, \qquad (60)
\end{aligned}
$$

where the last inequality comes from the the Lemma 4.1 in [3] and Assumption 3. Hence, we have

$$T_1 \geq \frac{1}{2}\left\|\nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right)\right\|^2 - \sum_{k=1}^{N} p_k \left\|\mathbb{E}\boldsymbol{h}_{m,k}^{(t)} - \nabla L_{m,k}^{(t)}(\boldsymbol{\Theta}_m^{(t)})\right\|^2 - 2G\rho_m^{(t)}. \tag{61}$$

According to the proof in Section C.3 and Lemma 5 in [33] , we have the following bound for T_2:

$$\begin{aligned}
T_2 \leq & 2\left[\mathbb{E}\left\|\frac{\sum_{k\in\mathbb{S}_l^{(t)}} p_k \boldsymbol{h}_{m,k}^{(t)}}{\sum_{k\in\mathbb{S}_l^{(t)}} p_k} - \frac{\sum_{k\in\mathbb{S}_l^{(t)}} p_k \mathbb{E}\boldsymbol{h}_{m,k}^{(t)}}{\sum_{k\in\mathbb{S}_l^{(t)}} p_k}\right\|^2\right] + 2\left[\left\|\frac{\sum_{k\in\mathbb{S}_l^{(t)}} p_k \mathbb{E}\boldsymbol{h}_{m,k}^{(t)}}{\sum_{k\in\mathbb{S}_l^{(t)}} p_k}\right\|^2\right] \\
\leq & 2\left[\mathbb{E}\left\|\frac{\sum_{k\in\mathbb{S}_l^{(t)}} p_k \boldsymbol{h}_{m,k}^{(t)}}{\sum_{k\in\mathbb{S}_l^{(t)}} p_k}\right\|^2 + \left\|\frac{\sum_{k\in\mathbb{S}_l^{(t)}} p_k \mathbb{E}\boldsymbol{h}_{m,k}^{(t)}}{\sum_{k\in\mathbb{S}_l^{(t)}} p_k}\right\|^2\right] + 2\left[\left\|\frac{\sum_{k\in\mathbb{S}_l^{(t)}} p_k \mathbb{E}\boldsymbol{h}_{m,k}^{(t)}}{\sum_{k\in\mathbb{S}_l^{(t)}} p_k}\right\|^2\right] \\
\leq & 4\left[\mathbb{E}\frac{\sum_{k\in\mathbb{S}_l^{(t)}} p_k \|\boldsymbol{h}_{m,k}^{(t)}\|^2}{\sum_{k\in\mathbb{S}_l^{(t)}} p_k} + \frac{\sum_{k\in\mathbb{S}_l^{(t)}} p_k \mathbb{E}\|\boldsymbol{h}_{m,k}^{(t)}\|^2}{\sum_{k\in\mathbb{S}_l^{(t)}} p_k}\right] + 2\left[\left\|\frac{\sum_{k\in\mathbb{S}_l^{(t)}} p_k \mathbb{E}\boldsymbol{h}_{m,k}^{(t)}}{\sum_{k\in\mathbb{S}_l^{(t)}} p_k}\right\|^2\right] \\
\leq & 8\tau^2 G \frac{\sum_{k\in\mathbb{S}_l^{(t)}} p_k^2}{(\sum_{k\in\mathbb{S}_l^{(t)}} p_k)^2} + 2\left[\left\|\frac{\sum_{k\in\mathbb{S}_l^{(t)}} p_k \mathbb{E}\boldsymbol{h}_{m,k}^{(t)}}{\sum_{k\in\mathbb{S}_l^{(t)}} p_k}\right\|^2\right] \qquad (62) \\
\leq & 8\tau^2 G + 2\left[\left\|\frac{\sum_{k\in\mathbb{S}_l^{(t)}} p_k \mathbb{E}\boldsymbol{h}_{m,k}^{(t)}}{\sum_{k\in\mathbb{S}_l^{(t)}} p_k}\right\|^2\right] \\
\leq & 8\tau^2 G + 6\sum_{k=1}^{N} p_k \left\|\mathbb{E}\boldsymbol{h}_{m,k}^{(t)} - \nabla L_{m,k}^{(t)}(\boldsymbol{\Theta}_m^{(t)})\right\|^2 \\
& + 6\|\nabla L_m^{(t)}(\boldsymbol{\Theta}_m^{(t)})\|^2 + \frac{6}{S_t}\left(\beta^2\|\nabla L_m^{(t)}(\boldsymbol{\Theta}_m^{(t)})\|^2 + \kappa^2\right), \qquad (63)
\end{aligned}$$

where the inequality 62 is based on the Assumption 3 and the definition of $\boldsymbol{h}_{m,k}^{(t)}$.
According to the Assumption 3, we have

$$\begin{aligned}
& \mathbb{E}_{\mathbf{z}\sim p_{m-1,k}^{(t)}}\left[\left\|\nabla l_m\left(\mathbf{z};\boldsymbol{\Theta}_m\right) - \mathbb{E}_{\mathbf{z}\sim p_{m-1,k}^{(t)}}\nabla l_m\left(\mathbf{z};\boldsymbol{\Theta}_m\right)\right\|^2\right] \\
\leq & \mathbb{E}_{\mathbf{z}\sim p_{m-1,k}^{(t)}}\left[2\left\|\nabla l_m\left(\mathbf{z};\boldsymbol{\Theta}_m\right)\right\|^2 + 2\|\mathbb{E}_{\mathbf{z}\sim p_{m-1,k}^{(t)}}\nabla l_m\left(\mathbf{z};\boldsymbol{\Theta}_m\right)\|^2\right] \\
\leq & 4\mathbb{E}_{\mathbf{z}\sim p_{m-1,k}^{(t)}}\left\|\nabla l_m\left(\mathbf{z};\boldsymbol{\Theta}_m\right)\right\|^2 \\
\leq & 4G. \qquad (64)
\end{aligned}$$

According to the proof in Section C.5 of [33] and 64, we have the following bound

$$\begin{aligned}
& \frac{1}{2}\sum_{k=1}^{N} p_k \left\|\mathbb{E}\boldsymbol{h}_{m,k}^{(t)} - \nabla L_{m,k}^{(t)}(\boldsymbol{\Theta}_m^{(t)})\right\|^2 \\
\leq & \frac{4\eta_t^2 \widetilde{\mathcal{L}}_m^2 G}{1-D}(\tau^2 - 1) + \frac{D\beta^2}{2(1-D)}\left\|\nabla L_m^{(t)}(\boldsymbol{\Theta}_m^{(t)})\right\|^2 + \frac{D\kappa^2}{2(1-D)}, \qquad (65)
\end{aligned}$$

where $D = 4\eta_t^2\widetilde{\mathcal{L}}_m^2\tau(\tau-1) < 1$. If $D \leq \frac{1}{12\beta^2+1}$, then it follows that $\frac{1}{1-D} \leq 1 + \frac{1}{12\beta^2} \leq 2$ and $\frac{3D\beta^2}{1-D} \leq \frac{1}{4}$. These facts can help us further simplify the inequality. One can obtain

$$\begin{aligned}
&6\sum_{k=1}^{N} p_k \left\| \mathbb{E}\boldsymbol{h}_{m,k}^{(t)} - \nabla L_{m,k}^{(t)}(\boldsymbol{\Theta}_m^{(t)}) \right\|^2 \\
\leq & 96\eta_t^2\widetilde{\mathcal{L}}_m^2 G(\tau^2-1) + \frac{1}{2}\left\|\nabla L_m^{(t)}(\boldsymbol{\Theta}_m^{(t)})\right\|^2 + 48\eta_t^2\widetilde{\mathcal{L}}_m^2\kappa^2\tau(\tau-1) \\
\leq & 96\eta_t^2\widetilde{\mathcal{L}}_m^2 G(\tau^2-1) + \frac{1}{2}\left\|\nabla L_m(\boldsymbol{\Theta}_m^{(t)})\right\|^2 + G\rho_m^{(t)} + 48\eta_t^2\widetilde{\mathcal{L}}_m^2\kappa^2\tau(\tau-1). \qquad (66)
\end{aligned}$$

Then we have

$$\begin{aligned}
T_2 \leq & 8\tau^2 G + 6\sum_{k=1}^{N} p_k \left\| \mathbb{E}\boldsymbol{h}_{m,k}^{(t)} - \nabla L_{m,k}^{(t)}(\boldsymbol{\Theta}_m^{(t)}) \right\|^2 \\
& + 6\|\nabla L_m^{(t)}(\boldsymbol{\Theta}_m^{(t)})\|^2 + \frac{6}{S_t}\left(\beta^2\|\nabla L_m^{(t)}(\boldsymbol{\Theta}_m^{(t)})\|^2 + \kappa^2\right) \\
\leq & 8\tau^2 G + 96\eta_t^2\widetilde{\mathcal{L}}_m^2 G(\tau^2-1) + \frac{1}{2}\left\|\nabla L_m(\boldsymbol{\Theta}_m^{(t)})\right\|^2 + G\rho_m^{(t)} + 48\eta_t^2\widetilde{\mathcal{L}}_m^2\kappa^2\tau(\tau-1) \\
& + 6\|\nabla L_m(\boldsymbol{\Theta}_m^{(t)})\|^2 + 12G\rho_m^{(t)} + \frac{6}{S_t}\left(\beta^2\|\nabla L_m(\boldsymbol{\Theta}_m^{(t)})\|^2 + 2\beta^2 G\rho_m^{(t)} + \kappa^2\right), \qquad (67)
\end{aligned}$$

where Eq. 67 uses the difference bound in Eq. 60. Plugging Eq. 61 and Eq. 67 back into Eq. 56, since $S_t \geq 1$, we have

$$\begin{aligned}
& \mathbb{E}\left[L_m\left(\boldsymbol{\Theta}_m^{(t+1)}\right)\right] - L_m\left(\boldsymbol{\Theta}_m^{(t)}\right) \\
\leq & -\frac{1}{2}\eta_t\left\|\nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right)\right\|^2 + 2\eta_t G\rho_m^{(t)} \\
& + \eta_t\left[16\eta_t^2\widetilde{\mathcal{L}}_m^2 G(\tau^2-1) + \frac{1}{12}\left\|\nabla L_m(\boldsymbol{\Theta}_m^{(t)})\right\|^2 + \frac{1}{6}G\rho_m^{(t)} + 8\eta_t^2\widetilde{\mathcal{L}}_m^2\kappa^2\tau(\tau-1)\right] \\
& + \frac{\eta_t^2\mathcal{L}_m}{2}\Bigg[8\tau^2G^2 + 96\eta_t^2\widetilde{\mathcal{L}}_m^2 G(\tau^2-1) + \frac{1}{2}\left\|\nabla L_m(\boldsymbol{\Theta}_m^{(t)})\right\|^2 + G\rho_m^{(t)} + 48\eta_t^2\widetilde{\mathcal{L}}_m^2\kappa^2\tau(\tau-1) \\
& \qquad + 6\|\nabla L_m(\boldsymbol{\Theta}_m^{(t)})\|^2 + 12G\rho_m^{(t)} + 6(\beta^2\|\nabla L_m(\boldsymbol{\Theta}_m^{(t)})\|^2 + 2\beta^2 G\rho_m^{(t)} + \kappa^2)\Bigg] \qquad (68) \\
\leq & -\left(\frac{5}{12}\eta_t - \frac{\eta_t^2\mathcal{L}_m}{4} - 3\eta^2\mathcal{L}_m - 3\eta_t^2\mathcal{L}_m\beta^2\right)\left\|\nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right)\right\|^2 \\
& + \eta_t\left[2G\rho_m^{(t)} + 16\eta_t^2\widetilde{\mathcal{L}}_m^2 G(\tau^2-1) + \frac{1}{6}G\rho_m^{(t)} + 8\eta_t^2\widetilde{\mathcal{L}}_m^2\kappa^2\tau(\tau-1)\right] \\
& + \frac{\eta_t^2\mathcal{L}_m}{2}\Bigg[8\tau^2G^2 + 96\eta_t^2\widetilde{\mathcal{L}}_m^2 G(\tau^2-1) + G\rho_m^{(t)} + 48\eta_t^2\widetilde{\mathcal{L}}_m^2\kappa^2\tau(\tau-1) \\
& \qquad + 12G\rho_m^{(t)} + 6(2\beta^2 G\rho_m^{(t)} + \kappa^2)\Bigg]. \qquad (69)
\end{aligned}$$

When we set $\eta_t \leq \min\{\frac{2}{(39+36\beta^2)\mathcal{L}_m}, 1\}$, we have

$$\begin{aligned}
& \frac{1}{4}\eta_t\|\nabla L_m\left(\boldsymbol{\Theta}_m^{(t)}\right)\|^2 \\
\leq & L_m\left(\boldsymbol{\Theta}_m^{(t)}\right) - \mathbb{E}\left[L_m\left(\boldsymbol{\Theta}_m^{(t+1)}\right)\right] \\
& + \left(2G + \frac{1}{6}G + \frac{\mathcal{L}_m}{2}G + 6\mathcal{L}_m G + 6\beta^2\mathcal{L}_m G\right)\eta_t\rho_m^{(t)} \\
& + \Bigg[16\widetilde{\mathcal{L}}_m^2 G(\tau^2-1) + 8\widetilde{\mathcal{L}}_m^2\kappa^2\tau(\tau-1) + 4\mathcal{L}_m\tau^2G^2 \\
& \qquad + 48\widetilde{\mathcal{L}}_m^2\mathcal{L}_m G(\tau^2-1) + 24\widetilde{\mathcal{L}}_m^2\mathcal{L}_m\kappa^2\tau(\tau-1) + 3\mathcal{L}_m\kappa\Bigg]\eta_t^2.
\end{aligned}$$

Taking the total expectation and averaging across all rounds, one can obtain

$$\frac{1}{4}\sum_{t=0}^{T}\eta_t\left\|\nabla L_m\left(\mathbf{\Theta}_m^{(t)}\right)\right\|^2 \leq \left[L_m\left(\mathbf{\Theta}_m^{(0)}\right) - L_m\left(\mathbf{\Theta}_m^{(T+1)}\right)\right] + A\sum_{t=0}^{T}\eta_t^2 + B\sum_{t=0}^{T}\eta_t\rho_m^{(t)}$$
$$\leq L_m\left(\mathbf{\Theta}_m^{(0)}\right) + A\sum_{t=0}^{T}\eta_t^2 + B\sum_{t=0}^{T}\eta_t\rho_m^{(t)},$$

where A, B are some positive constants. Now we get our final results:

$$\inf_{t\leq T}\mathbb{E}\left[\left\|\nabla L_m\left(\mathbf{\Theta}_m^{(t)}\right)\right\|^2\right] \leq \frac{1}{\sum_{t=0}^{T}\eta_t}\sum_{t=0}^{T}\eta_t\left\|\nabla L_m\left(\mathbf{\Theta}_m^{(t)}\right)\right\|^2$$
$$\leq \mathcal{O}\left(\frac{1}{\sum_{t=0}^{T}\eta_t}\right) + \mathcal{O}\left(\frac{\sum_{t=0}^{T}\rho_m^{(t)}\eta_t}{\sum_{t=0}^{T}\eta_t}\right) + \mathcal{O}\left(\frac{\sum_{t=0}^{T}\eta_t^2}{\sum_{t=0}^{T}\eta_t}\right).$$

It is obvious that $\frac{1}{\sum_{t=0}^{T}\eta_t} \to 0$ and $\frac{\sum_{t=0}^{T}\eta_t^2}{\sum_{t=0}^{T}\eta_t} \to 0$ if $T \to \infty$. As for $\frac{\sum_{t=0}^{T}\rho_m^{(t)}\eta_t}{\sum_{t=0}^{T}\eta_t}$, according to the Cauchy-Schwartz inequality, we have

$$\sum_{t=0}^{T}\rho_m^{(t)}\eta_t = \sum_{t=0}^{T}\sqrt{\rho_m^{(t)}}\left(\sqrt{\rho_m^{(t)}}\eta_t\right)$$
$$\leq \sqrt{\left(\sum_{t=0}^{T}\rho_m^{(t)}\right)\left(\sum_{t=0}^{T}\rho_m^{(t)}\eta_t^2\right)}$$
$$\leq \sqrt{\left(\sum_{t=0}^{T}\rho_m^{(t)}\right)\left(\sum_{t=0}^{T}\rho_m^{(t)}\right)\left(\sum_{t=0}^{T}\eta_t^2\right)}$$
$$< \infty.$$

Hence, we also have $\frac{\sum_{t=0}^{T}\rho_m^{(t)}\eta_t}{\sum_{t=0}^{T}\eta_t} \to 0$ if $T \to \infty$. □

C Experimental Details and Additional Results

We measure the Epoch Time in Fig. 3 on the Jetson TX2 platform, while we run all the other experiments on a sever with a single NVIDIA TITAN RTX GPU and Intel Xeon Gold 6254 CPU.

Hyperparameters for FL We partition the whole FMNIST and CIFAR-10 training set onto $N = 1000$ clients, with a training-validation ratio of $0.8 : 0.2$. We set the number of local training iterations as $\tau = 50$ for FMNIST and $\tau = 40$ for CIFAR-10, with the same local batch size $B = 10$. The maximal number of communication round is 1000 for FMNIST and 4000 for CIFAR-10. We use the initial learning rate $\eta_0 = 0.01$ for both FMNIST and CIFAR-10, while we halve the learning rate at $(400, 600, 800)$-th rounds of FMNIST experiments, and at $(2000, 3000)$-th rounds of CIFAR-10. We use SGD with weight decay 0.0001 and momentum 0.9 for both FMNIST and CIFAR-10.

Hyperparameters for AT Following Zizzo et al. [39], We adopt $\epsilon_0 = 0.15$ and $\alpha_0 = 0.02$ for FMNIST, and we use $\epsilon_0 = 8/255$ and $\alpha_0 = 2/255$ for CIFAR-10. For the 2-module FADE, we use $\epsilon_1 = 0.045$ and $\alpha_1 = 0.006$ for CNN-7 (2/2), and $\epsilon_1 = 0.01$ and $\alpha_1 = 0.003$ for VGG-11 (2/2). For the 3-module FADE, we use $\epsilon_1 = 0.015$ and $\alpha_1 = 0.004$ for VGG-11 (2/3), and $\epsilon_2 = 0.01$ and $\alpha_2 = 0.003$ for VGG-11 (3/3). For all the models and modules, we use PGD with $T = 10$ iterations for training and $T = 20$ iterations for testing.

Auxiliary Output Model For all the auxiliary output models we use in CNN-7 and VGG-11, we first apply a 2×2 Maxpooling on the input feature, and then we use a fully connected layer to get the early-exit loss.

C.1 Epoch Time and the Number of Parameters

Table 3 provides the exact running time of each local epoch (An epoch contains 50 iterations of PGD-10 AT, with batch size $B = 10$) and the number of parameters of each module. The epoch time is measured on Jetson TX2 paltform.

Table 3: Epoch time and the number of parameters in each module.

Module	Epoch Time (s)	# Params
CNN-7	8.123	574,240
CNN-7 (1/2)	5.477	61,344
CNN-7 (2/2)	4.917	518,656
VGG-11	48.427	9,747,136
VGG-11 (1/2)	29.117	4,504,256
VGG-11 (2/2)	19.777	5,248,000
VGG-11 (1/3)	14.377	970,432
VGG-11 (2/3)	15.560	3,544,064
VGG-11 (3/3)	19.777	5,248,000

C.2 Learning Curves of 3-Module Training

The learning curves for "FADE (100% 3-Module)", "FADE (20% Joint + 80% 3-Module)" and "FADE (20% Joint + 30% 2-Module + 50% 3-Module)" are given in Fig. 6. Although FADE with 100% 3-Module clients does not achieve the same robustness as fully joint AT, FADE with mixed clients can recover the adversarial accuracy of that in joint training.

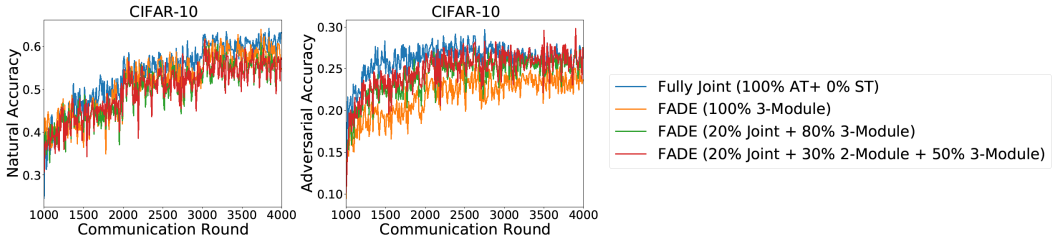

Figure 6: Learning curves of training schemes with 3-module FADE.