Sylva: Tailoring Personalized Adversarial Defense in Pre-trained Models via Collaborative Fine-tuning

Tianyu Qi
Sun Yat-sen University
Shenzhen, China
qity9@mail2.sysu.edu.cn

Lei Xue
Sun Yat-sen University
Shenzhen, China
xuelei3@mail.sysu.edu.cn

Yufeng Zhan
Beijing Institute of Technology
Beijing, China
yu-feng.zhan@bit.edu.cn

Xiaobo Ma
Xi'an Jiaotong University
Xi'an, China
xma.cs@xjtu.edu.cn

ABSTRACT

The growing adoption of large pre-trained models in edge computing has made deploying model inference on mobile clients both practical and popular. These devices are inherently vulnerable to direct adversarial attacks, which pose a substantial threat to the robustness and security of deployed models. Federated adversarial training (FAT) has emerged as an effective solution to enhance model robustness while preserving client privacy. However, FAT frequently produces a generalized global model, which struggles to address the diverse and heterogeneous data distributions across clients, resulting in insufficiently personalized performance, while also encountering substantial communication challenges during the training process.

In this paper, we propose *Sylva*, a personalized collaborative adversarial training framework designed to deliver customized defense models for each client through a two-phase process. In Phase 1, *Sylva* employs LoRA for local adversarial fine-tuning, enabling clients to personalize model robustness while drastically reducing communication costs by uploading only LoRA parameters during federated aggregation. In Phase 2, a game-based layer selection strategy is introduced to enhance accuracy on benign data, further refining the personalized model. This approach ensures that each client receives a tailored defense model that balances robustness and accuracy effectively. Extensive experiments on benchmark datasets demonstrate that *Sylva* can achieve up to 50× improvements in communication efficiency compared to state-of-the-art algorithms, while achieving up to 29.5% and 50.4% enhancements in adversarial robustness and benign accuracy, respectively.

CCS CONCEPTS

• **Security and privacy**; • **Computing methodologies** → **Artificial intelligence**;

KEYWORDS

Pre-trained Models, Personalized federated learning, Adversarial training, Fine-tuning

ACM Reference Format:
Tianyu Qi, Lei Xue, Yufeng Zhan, and Xiaobo Ma. 2025. Sylva: Tailoring Personalized Adversarial Defense in Pre-trained Models via Collaborative Fine-tuning. In *Proceedings of 2025 ACM SIGSAC Conference on Computer and Communications Security (CCS '25).* ACM, New York, NY, USA, 20 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

With the rapid advancement of large language models (LLM), large-scale pre-trained models have garnered widespread attention across various fields, including computer vision [11], autonomous driving [66], and healthcare [22], etc. Fine-tuning pre-trained models for downstream tasks has gradually established itself as a novel learning paradigm [19, 20, 35]. Meanwhile, the increasing computational power of edge devices has facilitated the localized deployment of the pre-trained models, unlocking their potential for various applications on devices [37, 68].

However, recent studies have revealed substantial security risks associated with deploying pre-trained models on edge devices. The typically weak protection mechanisms of these devices render their various permissions susceptible to unauthorized access, exposing sensitive resources to potential exploitation [74]. Furthermore, the parameters of the machine learning model stored locally on edge devices are often inadequately secured, increasing the risk of theft or manipulation [9]. On the other hand, many pre-trained model weights and architectures are open-source, allowing attackers to easily obtain original parameters and designs from public repositories. This accessibility enables the development of more sophisticated attack strategies, such as embedding backdoors or executing poisoning attacks on fine-tuned models for downstream tasks [21, 31]. Such vulnerabilities pose significant threats to the reliability and security of edge-based machine learning applications, highlighting an urgent need for robust defense mechanisms.

To mitigate these challenges, adversarial training (AT) has become a widely utilized defense strategy [25, 40, 70]. By incorporating benign and adversarial data during the training process, this approach enhances model robustness while striving to maintain accuracy. Despite its effectiveness, traditional adversarial training faces notable limitations when applied in real-world scenarios. As

CCS '25, October 13-17, 2025, Taipei, Taiwan

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

shown on the left side of Fig. 1, the challenges of adversarial training stem from several key factors. First, the limited data available to local clients often hinders effective adversarial training, leading to suboptimal performance or increased risks of overfitting during fine-tuning for downstream tasks. Second, the data across different clients is non-independent and identically distributed (non-IID). For instance, in autonomous driving, data collected under extreme weather conditions is scarce [41], while in healthcare, datasets from different hospitals exhibit significant variability [65]. This variability exposes weaknesses that attackers can exploit, such as rare weather conditions in autonomous driving or underrepresented patient groups in healthcare. Additionally, privacy concerns prohibit uploading client data to the cloud for centralized training, further exacerbating these challenges.

In multi-client environments, federated adversarial training (FAT) has proven to be a powerful approach for enhancing model robustness in multi-client environments while preserving privacy [78]. As large pre-trained models gain prominence, fine-tuning through FAT is expected to become a dominant trend, enabling devices to contribute their limited data via periodic synchronization and knowledge sharing. Despite its potential, most existing studies have concentrated on building generalized global models to address the challenges of non-IID data [6, 27, 75]. However, in the presence of significant data heterogeneity, such global models often fail to deliver effective defense tailored to individual clients. Their strong generalization, while advantageous in some contexts, limits their ability to adapt to unique client-specific data distributions, resulting in diminished performance, as illustrated on the right side of Fig. 1. Additionally, the integration of FAT with large pre-trained models introduces communication overhead, which may significantly impact training speed, particularly in resource-constrained environments. These limitations expose a glaring gap in current methodologies and raise an urgent question: Can we design a personalized adversarial training framework that provides robust and tailored defense models for each client while preserving the training efficiency on edge devices?

This study aims to address three critical challenges in adversarial fine-tuning pre-trained models in multi-client environments: (1) **Designing a personalized defense framework**: Given the heterogeneous data distributions across clients, a collaborative adversarial training framework is required to provide each client with a tailored defense model while ensuring data privacy. (2) **Enhancing both adversarial robustness and benign accuracy**: Developing a training paradigm to improve the adversarial robustness of models while maintaining high accuracy on benign data is crucial to ensure reliable performance in diverse and challenging scenarios. (3) **Balancing efficiency for clients**: Achieving improved performance on resource-constrained edge devices requires addressing communication and training efficiency challenges. By achieving these objectives, each client can obtain a model that is both robust and highly accurate, tailored specifically to its unique data distribution. Such models excel in downstream tasks under typical conditions while providing strong, personalized defenses against adversarial attacks targeting individual clients.

To address the challenges above, we propose *Sylva*, a collaborative framework that tailors pre-trained models with robust defense capabilities through personalized adversarial fine-tuning for edge devices. *Sylva* is designed to fine-tune models in a way that simultaneously enhances robustness and accuracy for each client, even in environments with heterogeneous data distributions. To validate the feasibility of our approach, we first conduct preliminary experiments leveraging the low-rank adaptation (LoRA) algorithm for personalized adversarial fine-tuning in real-world scenarios. These experiments not only demonstrate the potential of LoRA in adversarial training but also expose the critical limitations of existing defense strategies. Then we present *Sylva*'s two-phase training framework. In the first phase, personalized adversarial training divides the model into two modules: the LoRA module and the classifier module. The parameters of the LoRA module are shared and aggregated across clients to enhance the generalization ability of the model backbone for adversarial feature extraction. Meanwhile, the classifier module is fine-tuned locally on each client, enabling personalized improvements in classifying adversarial samples based on each client's unique data distribution. We propose a novel adaptive class-balanced dynamic weighted loss for handling data heterogeneity and a ball-tree-based aggregation algorithm to accelerate adversarial training convergence. In the second phase, we employ a Shapley value-based cooperative game approach to enhance the accuracy of benign data for each client. We identify and selectively freeze the optimal layers within each model, striking a critical balance that maximizes accuracy of benign data while minimizing any compromise in robustness. This strategic layer adjustment ensures that each client's model achieves a higher degree of personalization, effectively navigating the trade-off between benign accuracy and adversarial robustness. The main contributions of this paper can be summarized as follows:

- To our best knowledge, we propose the first personalized collaborative adversarial training framework for pre-trained models, enabling client-specific defenses while preserving privacy and minimizing overhead on edge devices.
- We introduce a novel training paradigm for LoRA-based adversarial fine-tuning, integrating adaptive optimization strategies with a ball-tree-based model aggregation and a newly designed loss, to enhance robust generalization under heterogeneous data distributions.
- We propose a cooperative game-based layer freezing method for pre-trained models, leveraging a novel value function and Monte Carlo sampling to optimize the trade-off between benign data accuracy and model robustness.
- We conduct experiments on widely used datasets and pre-trained models, comparing *Sylva* with popular defenses under various attack scenarios, demonstrating its effectiveness in personalized defense.

2 PRELIMINARIES

2.1 Federated Adversarial Training

Federated learning is an effective training algorithm for protecting client data privacy [34, 42, 53]. In a distributed system with N clients and a central server, each client i has its private dataset $(x, y) \in \mathcal{D}_i$, where x represents the benign samples, and y denotes the ground truth. The server initializes the model, and client i downloads the model, denoted as w_i and trains it using local data. The training

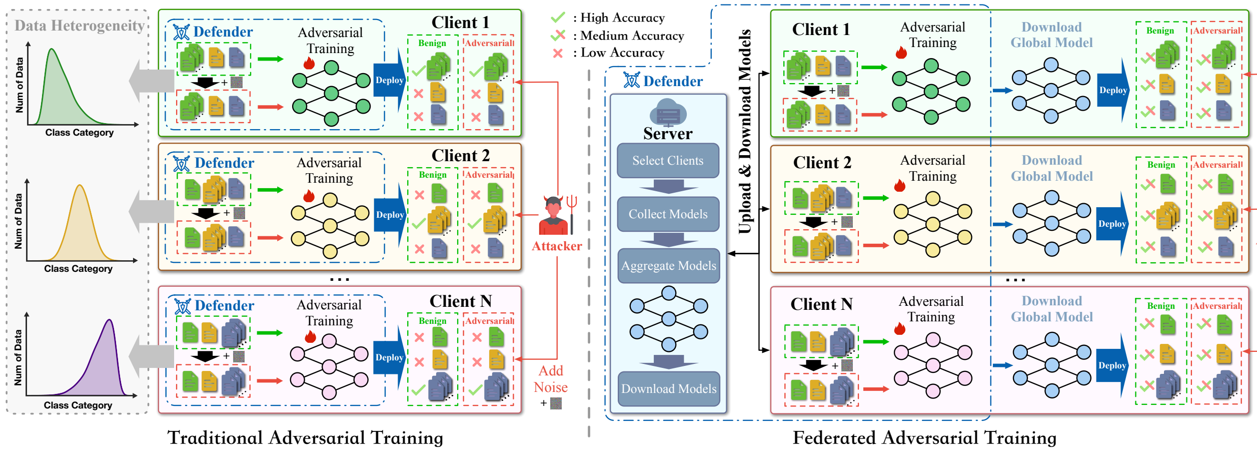


Figure 1: Challenges of adversarial training in multi-client scenario. (1) Left: Local traditional adversarial training performs well for classes with abundant data but struggles with those that are rare or unseen. (2) Right: Federated adversarial training addresses these issues by improving generalization across clients. However, it lacks the necessary personalization to meet the specific needs of individual clients and their unique data distributions.

objective for each client can be expressed as

$$f_i(w_i) = \frac{1}{|\mathcal{D}_i|} \sum_{(x,y)\in\mathcal{D}_i} f_i(w_i, x, y), \tag{1}$$

where $|\mathcal{D}_i|$ denotes the size of the dataset for client i.

For federated adversarial training, the local training objective is to enable the model to recognize both benign samples and adversarial samples, thereby improving its robustness through adversarial training. For a benign sample x, we add noise σ to generate an adversarial sample $x^{adv} = x + \sigma$. The goal of generating robust adversarial samples is to find the noise that maximizes the model loss $\mathcal{L}(w_i, x + \sigma, y)$. The local training objective for clients in federated adversarial training can be expressed as

$$f_i(w_i, x, x^{adv}, y) = \min \mathbb{E}_{x\sim\mathcal{D}_i}\left[\max_{\|x^{adv}-x\|_\infty \le \delta} \mathcal{L}(w_i, x, x^{adv}, y)\right]. \tag{2}$$

After training locally for a certain number of rounds, clients upload their models to the server, where parameter aggregation is performed using the data size of each device as a weight, represented as

$$w_g = \sum_{i=1}^{N} \frac{|\mathcal{D}_i| w_i}{\sum_{i=1}^{N} |\mathcal{D}_i|}. \tag{3}$$

Subsequently, clients can download the global model w_g for further local training. This process is repeated until final convergence.

2.2 Low-Rank Adaptation

Fine-tuning pre-trained models in mobile computing is challenging due to limited GPU memory, which often makes training all model parameters impractical. Low-rank adaptation (LoRA) [20] addresses this issue by freezing pre-trained model parameters and introducing lightweight trainable layers into transformer modules, significantly reducing computational overhead.

During the fine-tuning process, the updated parameters exhibit a low intrinsic rank. For a LoRA fine-tuned model $w \in \mathbb{R}^{r_{in}\times C}$, where r_{in} represents the input dimension, and C denotes the total number of prediction classes. The entire model can be divided into two parts: the backbone $w^B \in \mathbb{R}^{r_{in}\times r_{out}}$ and the classifier $w^C \in \mathbb{R}^{r_{out}\times C}$. The backbone's parameters can be represented as the combination of the pre-trained model parameters $w^P \in \mathbb{R}^{r_{in}\times r_{out}}$ and the LoRA module parameters $w^L \in \mathbb{R}^{r_{in}\times r_{out}}$. The LoRA parameters w^L can be expressed as a product of two low-rank matrices $w^{LA} \in \mathbb{R}^{r_{in}\times r}$ and $w^{LB} \in \mathbb{R}^{r\times r_{out}}$. We define the overall mapping of the model as $\mathcal{F}(\cdot)$, where the pre-trained model mapping and LoRA module mapping are $\mathcal{F}^P(\cdot)$ and $\mathcal{F}^L(\cdot)$, and the classifier mapping is $\mathcal{F}^C(\cdot)$. For a sample x the model output can be expressed as

$$\hat{y} = \mathcal{F}(x) = \mathcal{F}^C(\mathcal{F}^P(x) + \mathcal{F}^L(x)). \tag{4}$$

During training, the parameters of w^P remain frozen, and only w^L is updated, achieving a performance comparable to full-parameter fine-tuning.

2.3 Threat Model

We analyze the security risks associated with deploying large pre-trained models in distributed multi-client scenarios, including perspectives from both attackers and defenders.

2.3.1 *Attacker.* We posit that deploying models on client devices introduces potential system-level vulnerabilities, which attackers can exploit to gain access to client permissions and compromise the model's inference process through adversarial attacks.

Depending on the level of access an attacker has to the client model, these attacks can be classified into two categories:

- **White-box attacks**. The attacker has full access to all parameters of the deployed model and can leverage this information to generate highly effective adversarial samples. This scenario forms the basis for most existing attack methodologies.
- **Grey-box attacks**. The attacker knows the pre-trained model from which the client's task is fine-tuned and can access the original pre-trained model parameters (e.g., publicly platforms like Hugging Face) but lacks access to the fine-tuned parameters deployed on the client.

Table 1: Adversarial training with/without LoRA

Metrics	Methods	ResNet18	ViT-T	ViT-B	ViT-L
AR↑ (%)	w-LoRA	48.22	**58.72**	**61.14**	**64.69**
	w/o-LoRA	**54.36**	53.28	59.02	62.18
BA↑ (%)	w-LoRA	59.35	**63.27**	**69.73**	**75.91**
	w/o-LoRA	**62.05**	60.14	64.37	70.66
Time↓ (10^3s/epoch)	w-LoRA	**0.31**	**0.84**	**1.60**	**4.90**
	w/o-LoRA	0.38	1.13	1.90	6.38
Paras↓ (M)	w-LoRA	**0.51**	**1.95**	**2.06**	**2.29**
	w/o-LoRA	11.16	7.37	84.84	292.14
Mem↓ (G)	w-LoRA	**0.98**	**0.89**	**3.14**	**6.63**
	w/o-LoRA	1.25	1.04	4.86	11.43

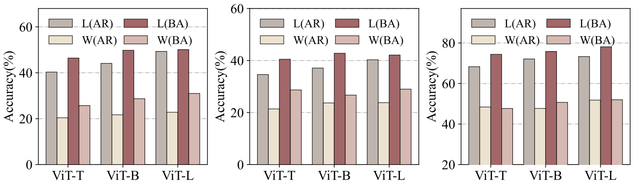
The objective of the attacker in both scenarios is to exploit the accessible model parameters to execute adversarial attacks on the client's downstream tasks. The adversarial samples generated must meet two critical criteria: (1) **Inconspicuousness**. The perturbations in the adversarial samples should be subtle enough to go unnoticed by humans. (2) **Impactfulness**. The adversarial attacks must effectively compromise the model's downstream tasks, even when standard defense mechanisms are in place.

2.3.2 Defender. We propose involving defenders in the fine-tuning process of client models for downstream tasks to enhance robustness. Collaborative fine-tuning is assumed to take place within a distributed framework based on a parameter server (PS) architecture [26]. In this setup, the defender operates at the server level, providing macro-level oversight and regulation of client activities to ensure robust and secure model training.

Defenders should possess the following three characteristics:

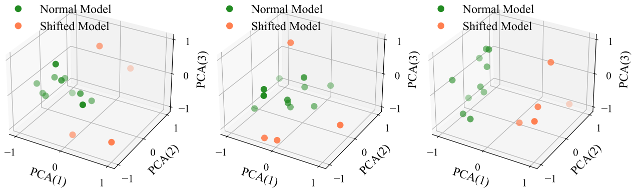
- **Data Privacy**. The defense mechanism must safeguard the privacy and security of each client's private data, ensuring no compromise occurs during the training or deployment process.
- **Heterogeneity**. Clients operate in diverse scenarios, requiring the defense approach to support personalized models that account for individual preferences and adversarial capabilities.
- **Efficiency**. To accommodate the limited computational resources of clients and mitigate potential network congestion, the defense strategy should prioritize efficiency, minimizing both memory usage and communication overhead.

In the FAT scenario, it is considered that for clients with similar data distributions, there are dedicated and advanced secure devices specifically used for data collection and defense training. For example, in autonomous driving, secure vehicles can be utilized to collect driving data and train defense models, ensuring robust security during the training process. Upon completing the training phase, the models are deployed to other vehicles with similar distribution but lower security capabilities, such as those within the same geographic region, to perform inference tasks. The FAT training paradigm ensures that attackers can only target the inference process on deployed devices, as the secure vehicles used during the training phase, equipped with robust defenses, remain inaccessible and protected from attacks.



(a) CIFAR-10 (b) STL-10 (c) GTSRB

Figure 2: Performance of personalized LoRA in adversarial training under heterogeneous environments



(a) CIFAR-10 (b) STL-10 (c) GTSRB

Figure 3: PCA embedding of the LoRA model after federated adversarial training

2.4 Preliminary Analysis of Existing Issues

In this section, we explore the critical issues in fine-tuning pre-trained models for multi-client attack-defense systems. To address these issues, we design experiments to validate their impact, offering essential insights for the development of *Sylva*. Specifically, we raise four key research questions:

- **Q1**: How can the computational resources of edge devices be effectively leveraged for adversarial defense training?
- **Q2**: How can personalized framework be tailored to meet the unique requirements of each client?
- **Q3**: How can valuable knowledge be effectively extracted from clients with diverse data distributions?
- **Q4**: How can the trade-off between adversarial robustness and benign accuracy be systematically optimized?

We conduct preliminary experiments to explore the above questions, highlighting our discoveries for each experiment and identifying remaining challenges that guide the design of *Sylva*.

2.4.1 Impact of Adversarial Training with LoRA. Adversarial training via full model training has demonstrated effectiveness, but it incurs high training costs, rendering it inefficient for edge devices with limited computational resources. To address this, we conduct experiments to evaluate the effectiveness of adversarial fine-tuning using LoRA.

We utilize the widely-used ResNet [16] and ViT [11] (ViT-T/16, ViT-B/16, ViT-L/16) pre-trained models, applying the TRADES algorithm [70] for adversarial fine-tuning via LoRA on the CIFAR-10 dataset [24]. ResNet and ViT-T are trained for 50 epochs, while ViT-B and ViT-L are trained for 100 epochs. The test environment is simulated according to the setup described in Section 4. The experimental results are evaluated using five metrics: *Adversarial Robustness* (AR), assessed by accuracy on adversarial samples; *Benign Accuracy* (BA), assessed by accuracy on benign samples;

Time, which tracks the time required per training epoch; *Parameter Size* (Paras), which quantifies the number of model parameters fine-tuned during training; *GPU Memory* (Mem), the GPU required for adversarial training with a fixed batch size.

The experimental results, presented in Table 1, demonstrate that adversarial fine-tuning using LoRA significantly reduces training time by up to 23.2% compared to full model fine-tuning. Moreover, LoRA requires only 0.7% of the parameters needed for full model training, saving approximately 30.5% of GPU memory during training and making it highly efficient in resource-constrained edge devices. Additionally, as model size increases, both AR and BA improve with LoRA-based adversarial fine-tuning. This aligns with findings in other tasks [20, 56], where as the model size increases, the low-rank eigenvalues in the model's mapping space capture most of the energy. Consequently, low-rank fine-tuning enhances adaptation to downstream tasks while preserving the model's overall feature extraction capabilities.

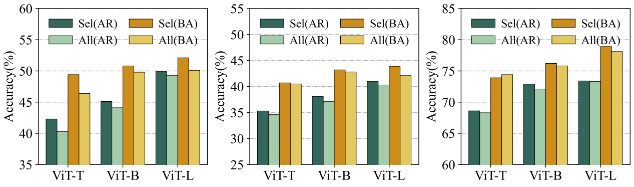
Discovery 1: LoRA-based adversarial fine-tuning preserves or even improves defense effectiveness while drastically reducing computational overhead, making it ideal for edge devices.

However, a critical challenge arises when scaling from single-client to multi-client training, as it requires designing a collaborative adversarial defense framework using LoRA to fully leverage its advantages.

2.4.2 Impact of the Personalized Framework. In multi-client environments, federated adversarial training has been widely adopted. However, most existing approaches rely on training a global model, which becomes ineffective when data distributions across devices are heterogeneous. This limitation arises from attackers designing tailored attack strategies based on the data distribution of downstream tasks. As a result, each client's model requires personalized defense mechanisms. This raises a crucial question: Can we decompose each client's model into two distinct components—one focused on learning generalized adversarial features, and the other on capturing personalized classification outcomes?

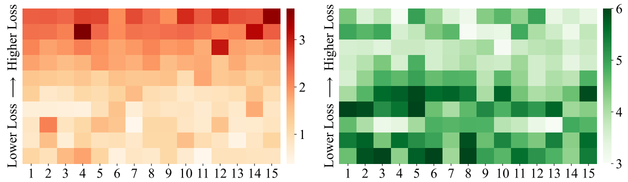
Given the powerful feature extraction capabilities of pre-trained model backbones, we believe that their feature extraction ability remains effective for adversarial examples after adversarial fine-tuning. To explore this, we conduct an experiment simulating two clients, where the training and testing data for each client are split according to the same Dirichlet distribution, thereby simulating a unique non-IID environment for each client [29]. We utilize three different ViT models and implement a federated adversarial training strategy, applying TRADES for local fine-tuning through LoRA, on the CIFAR-10 [24], STL-10 [7], and GTSRB [59] datasets. The federated adversarial training involve 5 local training epochs followed by parameter upload and cloud aggregation, repeated for 15 rounds. For the model upload step, we compare two strategies: uploading whole parameters (W) versus uploading only the LoRA parameters of the backbone while excluding the classifier parameters (L). The latter approach enables the training of a shared global backbone while allowing each client to personalize its classifier.

As shown in Fig. 2, the personalized aggregation approach (L) achieves superior performance in both BA and AR metrics on average for each client. This is attributed to our framework's ability to learn generalized features while tailoring the classifier. In contrast, the whole parameter aggregation approach (W) performs poorly, as aggregation in a heterogeneous environment induces model drift, hindering convergence.



(a) CIFAR-10 **(b) STL-10** **(c) GTSRB**

Figure 4: Comparison of aggregation methods for adversarial training in heterogeneous environments



(a) Adversarial robustness **(b) Benign accuracy**

Figure 5: Comparison of aggregation methods for adversarial training in heterogeneous environments

Discovery 2: Aggregating only the backbone (LoRA) and personalizing the classifier preserves generalized feature extraction while enhancing the classifier's sensitivity to heterogeneity, ensuring both global generalization and personalized defense.

Despite the effectiveness of this personalized framework, a new training paradigm is urgently required to further enhance robust generalization and adaptability within the LoRA-based collaborative adversarial training framework.

2.4.3 Impact of the Aggregation Method. Many studies have shown that data heterogeneity in federated adversarial training causes model drift on client devices, with direct aggregation slowing convergence [6, 27, 75]. We believe that, in the personalized federated adversarial framework, LoRA parameters may also be impacted by non-IID data, leading to drifts that undermine the adversarial generalization of the fine-tuned backbone.

We simulate a system with 15 clients, where each client's heterogeneous data follows a Dirichlet distribution [29]. After 10 epochs of local adversarial training, we apply PCA algorithm to the LoRA parameters, reducing them to 3 dimensions, as shown in Fig. 3. The results reveal drifts in some local LoRA models due to data heterogeneity. We design two aggregation strategies: one aggregates all models (All), while the other selectively aggregates only those whose PCA-compressed parameters are close, excluding models with significant drifts (Sel). After the same federated adversarial fine-tuning, the averaged test results for each client, shown in Fig. 4, demonstrate that only aggregating normal models yields higher AR and BA metrics. This strategy removes models with excessive drift, thereby accelerating the generalization and convergence of the adversarial backbone.

Discovery 3: Data heterogeneity can cause LoRA model drift, and models with excessive drift, when aggregated, can enhance convergence speed and reduce adversarial training performance.

However, the drifted model may contain unique knowledge not present in others. Discarding it outright could expedite convergence but risks losing valuable information, ultimately undermining the effectiveness of adversarial training. Moreover, compressing all LoRA parameters using PCA is computationally intensive, with efficiency significantly declining as the number of clients scales. Thus, a more refined aggregation approach is needed.

2.4.4 Limitations of the Robustness-Accuracy Trade-off. In adversarial training within pre-trained models, the key challenge is to maximize benign accuracy while maintaining adversarial robustness. The method in [77] enhances benign accuracy by first computing the loss for each model layer using a set of adversarial samples. The top-p layers with the lowest loss—indicating minimal sensitivity to robustness—are selected for unfreezing, while the others remain frozen. The model is then trained on benign data, enhancing benign accuracy without sacrificing robustness.

However, the method has a critical flaw: the p selected layers may not remain optimal after joint selection due to layer interactions affecting overall performance. To verify this, we conduct the following experiment: for the robust models trained on the 15 clients, we measure the loss of each layer as in [77]. We then select multiple combinations of p layers, unfreezing them while freezing the others, and retrained the model using benign data.

Fig. 5 shows accuracy changes for adversarial and benign samples after fine-tuning. The x-axis represents clients, and the y-axis represents the sum of losses from each layer, arranged in ascending order. In Fig. 5a, the trend in robustness aligns with expectations: selecting p layers with lower loss (i.e., lower robustness sensitivity) has a lesser impact on robustness, while layers with higher loss negatively affect it. However, an intriguing observation is that the combination yielding the lowest loss does not always produce the best results, as layer interactions can alter the final outcome. Meanwhile, Fig. 5b shows significant variance in benign accuracy improvement, indicating the need for a trade-off: the selection with the lowest robustness loss does not always result in the greatest benign accuracy gain.

Discovery 4: Jointly selecting robustness-insensitive layers for fine-tuning effectively enhances benign accuracy, but the interactions among layers prevent achieving the optimal outcome.

Therefore, it is crucial to design an effective and efficient fine-tuning method to identify the appropriate layers, aiming to minimize the reduction in robustness while maximizing benign accuracy improvement.

3 METHODOLOGY

In this section, we present the framework of the *Sylva* algorithm, offering a comprehensive explanation of its key components and the overall training process.

3.1 Overview

Fig. 6 provides an overview of the framework and training process of the *Sylva* algorithm, which is divided into two distinct phases.

In the Phase 1, personalized adversarial fine-tuning, the algorithm is implemented within a federated learning framework, where each client employs the LoRA fine-tuning method to personalize adversarial training. Each client fine-tunes its pre-trained model using our proposed novel loss function, updating only the LoRA and classifier modules. After local training rounds, LoRA parameters are uploaded to the cloud, where a ball-tree-based method assigns weights for aggregation and updates. Clients then download the aggregated LoRA model, iterating this process until convergence. Through this method, each client ultimately obtains a shared global backbone, which incorporates the aggregated LoRA modules. This backbone is capable of generalizing robust feature extraction from adversarial samples. Simultaneously, each client retains its personalized classifier module, which ensures targeted defense against adversarial attacks specific to its own data distribution.

The Phase 2, cooperative game-based fine-tuning, aims to enhance the benign accuracy of models on standard downstream tasks while preserving their robustness. For the personalized models trained by each client in Phase 1, we assume they already exhibit sufficient robustness against adversarial samples. However, their inference performance on local benign data can still be further improved. To achieve this, each client independently and in parallel fine-tunes its model using its local benign dataset. In this phase, we introduce a value function and adopt a cooperative game-based approach leveraging Shapley values to identify the optimal layers of the large model to freeze during training. This strategic layer selection maximizes the value function, allowing the models to improve their accuracy on benign data while incurring minimal loss of robustness. The outcome is a refined client-specific model that balances robust feature extraction with enhanced performance on standard tasks.

3.2 Personalized Adversarial Fine-tuning

We introduce the training details of Phase 1, including the design of the novel LoRA-based training paradigm, the optimization of the aggregation algorithm, and the workflow of training process.

3.2.1 Adaptive Class-balanced Dynamic Weighted Loss. Current adversarial training algorithms predominantly focus on designing loss functions that balance the relationship between benign data and adversarial samples, seeking an optimal trade-off between accuracy and robustness. Building on this, we aim to design a novel loss function specifically adapted to LoRA fine-tuning, incorporating a class-weighting strategy that accounts for heterogeneous data distributions, enhancing adversarial training for each client's unique local data.

We address the class imbalance problem by utilizing the number of samples for each class. Specifically, on each client i, the class imbalance weight can be calculated based on its local dataset $\mathcal{D}_i$ as

$$h_{i_c}^{Base} = \frac{1-\gamma}{1-\gamma^{n_{i_c}}}, \tag{5}$$

where c represents a specific class, and n_{i_c} denotes the number of samples belonging to class c on client i. γ is a hyperparameter,

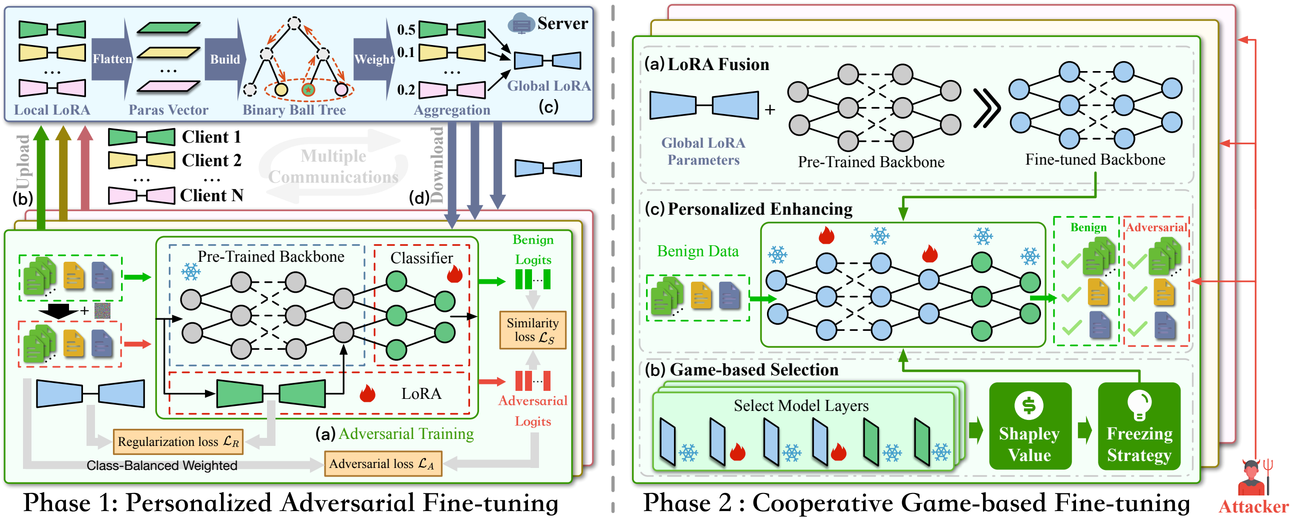


Figure 6: The overview of *Sylva*. (1) Left: Phase 1—Clients and the cloud collaborate in personalized federated adversarial fine-tuning to obtain a generalized backbone and personalized classifier. (2) Right: Phase 2-Clients apply a game-based algorithm to fine-tune layers, enhancing robustness while improving accuracy.

typically set in the range $[0.5, 0.99]$. We assume there are C total classes. The method reduces weights for frequent classes while emphasizing underrepresented ones.

However, pre-trained models are primarily trained on benign data and may lack sufficient sensitivity to adversarial samples. Moreover, when client data distributions differ significantly from the pre-training data, directly applying the aforementioned weight scheme in adversarial fine-tuning can lead to convergence challenges. To improve the model's generalization in extracting adversarial features across classes, we introduce a dynamic smoothing weight scheme to stabilize the training process. The weight is defined as

$$h_{i_c}^{Ada}(t) = \epsilon \cdot h_{i_c}^{Ada}(t-1) + (1-\epsilon) \cdot h_{i_c}^{Base}(t), \tag{6}$$

where ϵ is a smoothing hyperparameter in the range $[0, 1]$, and t represents the local training epoch. The smoothed class weights are normalized as

$$h_{i_c}(t) = \frac{h_{i_c}^{Ada}(t)}{\sum_{c=1}^{C} h_{i_c}^{Ada}(t)}. \tag{7}$$

As local adversarial training progresses, the smoothed weights gradually approach the preset class weights. The corresponding adversarial training loss can be expressed as

$$\mathcal{L}_A(w_i, x^{adv}, y) = \min \sum_{(x,y)\in\mathcal{D}_i} h_{i_y} \cdot \mathcal{L}_{CE}(\mathcal{F}(w_i, x^{adv}), y), \tag{8}$$

where $\mathcal{L}_{CE}(\cdot)$ denotes the cross-entropy loss to improve the robustness, h_{i_y} represents the smoothed class weight for the class y of the sample x at the current training epoch.

Furthermore, it is crucial to ensure that adversarial training does not significantly compromise the model's accuracy on benign data, requiring the model to maintain sensitivity to benign inputs. This requirement is quantified by measuring the similarity between the model's output vectors for benign and adversarial samples. To enforce it, we incorporate a KL-divergence loss term, defined as

$$\begin{aligned} \mathcal{L}_S(w_i, x, x^{adv}) &= \mathcal{L}_{KL}(\mathcal{F}(w_i, x), \mathcal{F}(w_i, x^{adv})) \\ &= \sum \text{softmax}(\mathcal{F}(w_i, x)) \cdot \log\left(\frac{\text{softmax}(\mathcal{F}(w_i, x))}{\text{softmax}(\mathcal{F}(w_i, x^{adv}))}\right). \end{aligned} \tag{9}$$

For the backbone module of pre-trained models, our optimization objective is to improve its generalization in feature extraction for adversarial samples. Meanwhile, the classifier module focuses on personalized classification performance. To account for client heterogeneity, we introduce a regularization loss to prevent excessive LoRA updates and preserve backbone generalization. The loss function is formulated as

$$\mathcal{L}_R(w_i^L, w_g^L) = \left\| w_i^L - w_g^L \right\|_2^2, \tag{10}$$

where w_i^L and w_g^L represent the LoRA parameters of client i and the global aggregated LoRA parameters, respectively. Finally, by combining Eq. (8), Eq. (9) and Eq. (10), we weight the above loss terms to obtain the adversarial training loss as

$$\mathcal{L} = \mathcal{L}_A + \lambda_1 \mathcal{L}_S + \lambda_2 \mathcal{L}_R. \tag{11}$$

Each client conducts local adversarial training based on Eq. (11), with updates restricted to the LoRA and classifier modules. Clients upload LoRA parameters after fixed epochs for global aggregation on the server.

3.2.2 Ball-tree-based Aggregation. Upon receiving the LoRA parameters from each client, the server faces model heterogeneity due to variations in local data distributions. Since the primary goal of adversarial training for the pre-trained model's backbone, fine-tuned with LoRA, is to generalize feature extraction from adversarial samples, models trained on highly biased client data could potentially disrupt the global model. Uniform aggregation of model weights across all clients may, therefore, hinder convergence efficiency.

We adopt a ball-tree-based [10] retrieval method to assign aggregation weights to each client model. The LoRA model parameters received from each client are flattened into vectors, represented as

$$\mathbf{G} = \{\mathcal{G}(w_1^L), \mathcal{G}(w_2^L), \dots, \mathcal{G}(w_N^L)\}, \tag{12}$$

where $\mathcal{G}(\cdot)$ is the flatten function. We treat each $\mathcal{G}(w_i^L)$ as a node and construct a ball tree for the entire vector $\mathbf{G}$. The ball tree algorithm is a spatial indexing method that organizes multidimensional data in a tree structure. Each non-leaf node represents a hypersphere enclosing a subset of data, facilitating efficient nearest neighbor searches by recursively partitioning the data. For client i , we can use the ball tree to identify the k client model vectors $\{\mathcal{G}(w_{j_1}^L), \mathcal{G}(w_{j_2}^L), \dots, \mathcal{G}(w_{j_k}^L)\}$ that are most similar to its parameters, and compute the distances between them

$$d_{ij_m} = \|\mathcal{G}(w_i^L) - \mathcal{G}(w_{j_m}^L)\|_2, \quad m = 1, 2, \dots, k \tag{13}$$

Then, gaussian weighting is applied to these distances to obtain the aggregation weight for each client model, denoted as

$$q_i = \frac{\sum_{m=1}^{k} \exp\left(-\frac{d_{ij_m}}{\sigma^2}\right)}{\sum_{i=1}^{N}\sum_{m=1}^{k} \exp\left(-\frac{d_{ij_m}}{\sigma^2}\right)}. \tag{14}$$

The weighting scheme favors clients with stable data, centrally located in high-dimensional space and exhibiting stronger robust generalization, while assigning lower weights to those with more extreme, dispersed distributions. Finally, by combining Eq. (3), the server can obtain the aggregated LoRA model as

$$w_g^L = \sum_{i=1}^{N} \frac{q_i |D_i| w_i}{\sum_{i=1}^{N} q_i |D_i|}. \tag{15}$$

Our aggregation computes weights over high-dimensional parameters, where naive nearest neighbor search incurs $O(N^2)$ complexity. Leveraging the ball tree algorithm reduces this to $O(N \log N)$ by efficiently capturing distance relations, enabling scalability to large-scale scenarios.

3.3 Cooperative Game-based Fine-tuning

After Phase 1, each client fuses the global LoRA with the pre-trained model to obtain a robustly generalized backbone. Adding a personalized classifier enables effective adversarial defense. However, we believe that benign accuracy can be further improved without sacrificing robustness.

3.3.1 Shapley Game for Layers. In Phase 2, each client independently trains using its own data in parallel, further fine-tuning its personalized model. Our goal is to select p layers from the L layers of the robust model for training on benign data, while freezing the remaining layers, to maximize accuracy improvements and minimize robustness degradation. This problem is framed as a cooperative game, with Shapley values [57] used to quantify the marginal contribution of each layer and optimize the overall value function.

Let $\mathbf{L}$ represent the set of model layers, and $\mathbf{S}$ denote one subset of layers selected for training while freezing the rest. We define two sensitivity losses as

$$\mathcal{L}_{rob}(\mathrm{S}) = \mathcal{L}_{CE}(\mathcal{F}(w_i^{\{\mathrm{S}\}}, x^{adv}), y) - \mathcal{L}_{CE}(\mathcal{F}(w_i, x^{adv}), y), \tag{16}$$

$$\mathcal{L}_{acc}(\mathrm{S}) = \mathcal{L}_{CE}(\mathcal{F}(w_i^{\{\mathrm{S}\}}, x), y) - \mathcal{L}_{CE}(\mathcal{F}(w_i, x), y), \tag{17}$$

to quantify the impact of training $\mathbf{S}$: $\mathcal{L}_{rob}$ measures changes in loss for adversarial samples, and $\mathcal{L}_{acc}$ measures changes in loss for benign samples. A smaller $\mathcal{L}_{rob}$ indicates that the selected layers are less sensitive to robustness, meaning fine-tuning them minimally affects robustness. Conversely, a larger $\mathcal{L}_{acc}$ suggests that the selected layers have potential for further optimization on benign data. Based on these metrics, we define the value function in the cooperative game as

$$v(\mathrm{S}) = \mathcal{L}_{acc}(\mathrm{S}) - \beta \cdot \mathcal{L}_{rob}(\mathrm{S}), \tag{18}$$

where β is a hyperparameter used to adjust the value in the cooperative game. Subsequently, we can calculate the Shapley value for each layer l as its average marginal contribution, denoted as

$$\phi_l = \sum_{\mathrm{S} \subseteq \mathbf{L} \setminus \{l\}} \frac{|\mathrm{S}|!(L - |\mathrm{S}| - 1)!}{L!} \left[v(\mathrm{S} \cup \{l\}) - v(\mathrm{S})\right]. \tag{19}$$

This parameter accounts for each participating layer's marginal contribution across all possible combinations of participants and computes the average. Finally, we select p layers, denoted as $\mathbf{P} = \{l_1, l_2, \dots, l_p\}$, where $\phi_{l_1} \geq \phi_{l_2} \geq \cdots \geq \phi_{l_p}$. By freezing the other layers and fine-tuning only these selected layers using benign data, the model's accuracy can be effectively improved.

3.3.2 Monte Carlo Sampling Optimization. While the cooperative game approach leveraging Shapley values effectively identifies the optimal layers for fine-tuning to enhance the model's benign accuracy, its computational complexity is prohibitively high. Specifically, for each layer, it requires evaluating all subsets of the remaining layers and computing the value function, resulting in a complexity of $O(2^L \times L)$. To address this, we plan to adopt Monte Carlo sampling to optimize computational cost.

We randomly generate B permutations of the model layers, denoted as $\pi^{(1)}, \pi^{(2)}, \dots, \pi^{(B)}$. For each permutation $\pi^{(b)}$, we identify the position of layer l and define the set of all layers preceding l in that permutation as $S_l^{(b)}$. The marginal contribution of layer l when added to $S_l^{(b)}$ is then represented as

$$\phi_l \approx \frac{1}{B} \sum_{b=1}^{B} \left[v(\mathrm{S}_l^{(b)} \cup \{l\}) - v(\mathrm{S}_l^{(b)})\right]. \tag{20}$$

As the number of samples B increases, the calculated contribution values converge to the results of the Shapley value game. The computational complexity is reduced to $O(B \times L)$, with the error decreasing at a rate of $O(1/\sqrt{B})$ as B grows.

3.4 The Workflow of *Sylva*

Algorithm 1 outlines the *Sylva* defense algorithm. Each client i holds its private benign dataset $\mathcal{D}_i$. A global model w_g and a local model w_i, both utilizing LoRA for fine-tuning, are constructed and deployed.

In Phase 1, personalized adversarial fine-tuning is implemented with the following steps:

- Initialize the server model w_g, and allow clients to download the LoRA model's parameters w_g^L from the server. Each client then constructs its local model w_i. (Line 4-7)
- Each client generates adversarial examples from its local dataset $\mathcal{D}_i$ using the PGD method. Then, the client trains locally for T_1 epochs using Eq. (11), only updating its LoRA parameters w_i^L and classifier parameters w_i^C. (Line 8-14)

Algorithm 1 *Sylva*'s Adversarial Training Process

```
1:  **Input**: Local training epoch $T_1$, aggregation epoch $T_2$, local benign data $\mathcal{D}_i$, global model $w_g(w_g^L, w_g^P, w_g^C)$, and local model $w_i(w_i^L, w_i^P, w_i^C)$, where $i \in \{1, 2, \dots, N\}$.
2:  **Output**: Personalized, robust, high-accuracy client models.
3:  *In Phase 1:*
4:  **Initialize**: Server's model $w_g$ and clients' models $w_i$.
5:  **for** 1 **to** $T_2$ **do**
6:      **for** client $i \in \{1, 2, \dots, N\}$ **parallel do**
7:          Client $i$ downloads the global LoRA model as $w_i^L$. Combine $w_i^L$, $w_i^P$ and $w_i^C$ to form $w_i$;
8:          **for** 1 **to** $T_1$ **do**
9:              **for** $x \in \mathcal{D}_i$ **do**
10:                 $x_{adv} = \text{PGD}(w_i, x)$;
11:                 Train and calculate loss $\mathcal{L}$ with Eq. (11);
12:                 $w_i^L, w_i^C \leftarrow \text{SGD}(w_i, \mathcal{L})$;
13:             **end for**
14:         **end for**
15:         Each client upload LoRA model $w_i^L$ to server;
16:     **end for**
17:     Server aggregate LoRA model with Eq. (15);
18: **end for**
19: *In Phase 2:*
20: **for** client $i \in \{1, 2, \dots, N\}$ **parallel do**
21:     Fuse $w_i^L$, $w_i^P$, and $w_i^C$ to form the model $w_i$;
22:     **for** layer $l \in w_i$ **do**
23:         Calculate marginal contribution with Eq. (20);
24:     **end for**
25:     Select layers $\mathbf{P} = \{l_1, l_2, \dots, l_p\}$, $\phi_{l_1} \geq \phi_{l_2} \geq \dots \geq \phi_{l_p}$;
26:     Freeze layers except $\mathbf{P}$ and train model $w_i$ by $\mathcal{D}_i$;
27: **end for**
```

- Clients upload their updated w_i^L parameters to the server, which aggregates them using Eq. (15) to update the global LoRA parameters w_g^L. This process is repeated for T_2 rounds. (Line 15-18)

After completing Phase 1, each client possesses the shared global LoRA model w_i^L, the frozen pre-trained backbone w_i^P, and a personalized classifier w_i^C. Phase 2 involves cooperative game-based fine-tuning, which proceeds as follows:

- Each client fuses the updated w_i^L, w_i^P, and w_i^C from Phase 1 to construct the complete local model w_i. It then calculates the marginal contribution ϕ_l for each layer l using Eq. (20), quantifying the relative contribution of each layer. (Line 21-24)
- Based on the computed marginal contributions, the top-p layers with the highest contribution values are selected to form the set $\mathbf{P}$. (Line 25)
- Using the client's local benign data, all layers outside the set $\mathbf{P}$ are frozen, while the layers within $\mathbf{P}$ are fine-tuned to enhance the model's benign accuracy. (Line 26)

Through this workflow, each client obtains a personalized large model tailored to its data distribution, achieving high accuracy on benign data while effectively resisting adversarial attacks.

Table 2: Performance comparison of *Sylva* and baseline under different datasets and attack algorithms (ViT-B/16)

Datasets	Baselines	Benign(BA)	FGSM	PGD	SparseFool	PAP
CIFAR-10	FA-PGD-AT	39.25	45.89	42.50	48.02	44.12
	FA-TRADES	50.12	45.04	43.39	45.71	43.56
	FA-Gen-AF	53.42	46.95	45.73	45.53	45.39
	FP-PGD-AT	41.04	45.28	43.63	48.51	45.43
	FP-TRADES	52.85	47.45	45.42	46.02	44.46
	FP-Gen-AF	54.70	48.52	46.21	49.88	47.12
	DBFAT	53.56	45.63	44.94	48.93	48.41
	Per-Adv	48.28	44.52	42.61	44.31	44.40
	Per-LoRA	53.02	47.21	44.32	46.71	44.35
	Sylva(Ours)	**59.03**	**52.88**	**55.03**	**52.68**	**51.89**
STL-10	FA-PGD-AT	34.53	39.85	38.61	40.26	39.88
	FA-TRADES	42.65	37.65	36.50	39.80	39.20
	FA-Gen-AF	44.72	38.57	37.18	39.65	40.15
	FP-PGD-AT	36.48	38.58	37.21	41.32	39.74
	FP-TRADES	43.68	39.10	37.21	38.92	38.85
	FP-Gen-AF	45.83	41.01	38.53	41.83	41.23
	DBFAT	45.55	41.05	38.01	41.60	40.75
	Per-Adv	36.71	40.20	38.87	39.93	39.67
	Per-LoRA	44.37	40.84	38.11	41.09	40.66
	Sylva(Ours)	**49.11**	**43.78**	**41.48**	**44.05**	**44.20**
GTSRB	FA-PGD-AT	60.17	68.34	68.98	71.67	71.52
	FA-TRADES	74.38	67.88	68.68	70.65	72.33
	FA-Gen-AF	76.45	68.91	67.89	71.72	71.48
	FP-PGD-AT	62.53	68.85	68.92	71.93	71.83
	FP-TRADES	76.43	69.37	69.85	70.88	71.49
	FP-Gen-AF	79.55	69.51	69.90	71.56	73.15
	DBFAT	79.40	68.58	68.72	70.70	73.45
	Per-Adv	63.70	68.03	67.83	69.68	70.09
	Per-LoRA	74.84	68.70	68.34	71.01	70.78
	Sylva(Ours)	**80.49**	**72.63**	**72.40**	**73.78**	**74.30**
CIFAR-100	FA-PGD-AT	12.45	25.82	25.27	27.36	26.59
	FA-TRADES	24.08	24.88	23.35	25.20	25.44
	FA-Gen-AF	23.65	25.33	24.20	26.16	25.47
	FP-PGD-AT	13.71	25.70	25.25	28.00	26.79
	FP-TRADES	24.61	23.83	25.32	26.67	25.69
	FP-Gen-AF	25.07	25.45	25.34	27.46	26.63
	DBFAT	25.02	25.11	25.29	27.11	28.33
	Per-Adv	15.31	24.37	24.47	25.02	26.80
	Per-LoRA	24.85	25.38	24.92	27.13	27.06
	Sylva(Ours)	**26.95**	**27.63**	**26.23**	**28.95**	**28.70**

4 EXPERIMENTS

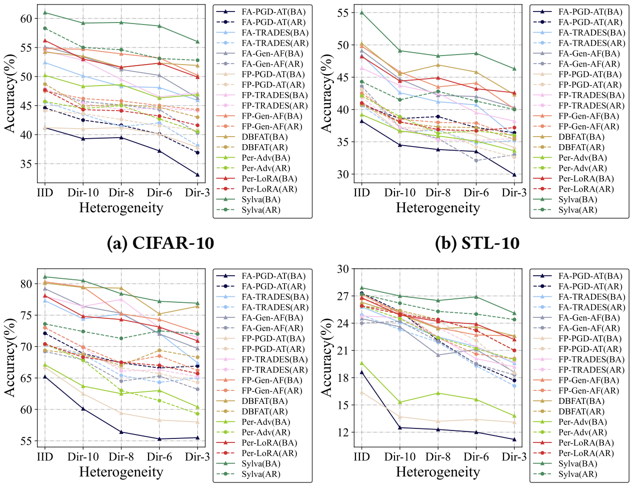
Throughout our experimental evaluation, we explore the performance of *Sylva* defense under diverse distributed environments and attack scenarios. We compare *Sylva* with baselines to highlight its personalized defense advantages and assess its feasibility and adaptability across different settings, addressing four key research questions.

- **RQ1**: How does *Sylva* perform against different attack algorithms in terms of robustness and accuracy?
- **RQ2**: How does *Sylva* handle varying heterogeneous data distributions across diverse clients?
- **RQ3**: How does *Sylva* achieve efficiency in communication and computation in distributed environments?
- **RQ4**: How do *Sylva*'s hyperparameters influence its overall robustness and accuracy performance?

4.1 Experimental Setup

We provide a brief overview of the experimental setup, with detailed specifics available in Appendix A.

Datasets and models. This study leverages datasets widely recognized in adversarial training, including CIFAR-10 [24], STL-10 [7], and GTSRB [59], as introduced in Section 2. To better align with tasks involving large models, the more complex CIFAR-100 dataset is also incorporated. For model selection, we utilize large



(a) CIFAR-10 **(b) STL-10**

(c) GTSRB **(d) CIFAR-100**

Figure 7: Comparison of adversarial training performance under different heterogeneity levels (ViT-B/16)

vision models based on transformers, specifically ViTs [11] pre-trained on the ImageNet dataset [55]. These models are categorized into three scales—ViT-T/16, ViT-B/16, and ViT-L/16—based on the number of transformer layers.

Attacks. We simulate both white-box and gray-box attacks using recent and widely adopted image attack algorithms. For white-box attacks, we employ the classic FGSM algorithm [14] and its variants, alongside the more robust PGD algorithm [40]. To explore subtle perturbations, we include the SparseFool attack [43], which modifies only a few pixels. For gray-box attacks, we utilize the PAP algorithm [5], which generates universal perturbations tailored for pre-trained models and remains effective against fine-tuned models without requiring knowledge of downstream tasks.

Baselines. We use several standard defense algorithms for adversarial training: PGD-AT [40], which employs the PGD algorithm to generate adversarial samples and enhance model robustness; TRADES [70], which optimizes the loss to balance accuracy on benign data and robustness against adversarial attacks; and Gen-AF [77], a fine-tuning approach for adversarial training on pre-trained models, maintaining robustness while improving accuracy. In the distributed scenario, we employ FedAvg [42] and FedProx [28], with FedProx addressing client data heterogeneity. We integrate them with various defense algorithms, establishing them as baseline approaches for comparison. Additionally, we incorporate DBFAT [72], designed specifically for non-IID settings, into our baseline approaches. We also include two personalized federated learning algorithms for comparison. [1] proposes addressing Byzantine attacks in distributed training by customizing the loss function for each client, which we refer to as Per-Adv. [63] introduces a method that computes LoRA-based trust weights among clients during large model fine-tuning to achieve personalized updates, which we denote as Per-LoRA. It is worth mentioning that we incorporate the TRADES loss into both personalized algorithms, thereby improving their alignment with the threat model and experimental tasks defined in this study.

Implementation. For data heterogeneity, client data is partitioned using a Dirichlet distribution [29] with 15 clients and a Dirichlet parameter of 10. In Phase 1, γ and ϵ are set to 0.9, λ_1 to 20, λ_2 to 0.001, and k, the number of clients for weight aggregation, is set to 5. In Phase 2, β is set to 5, B to 300, and p, the proportion of layers selected for training, to 3% of the total layers. The learning rate is 0.005. Federated adversarial training uses 5 local epochs, with ViT-T performing 30 cloud aggregation rounds, while ViT-B and ViT-L performing 50. Experiments are conducted on 4 NVIDIA A100 GPUs using the Ray framework [44] for multi-process client simulation. To assess *Sylva*'s efficiency in real-world scenarios, we conduct experiments on five real devices, which are used to simulate edge devices in various practical applications. These include three consumer-grade GPUs: GeForce RTX 4090, 3090, and 2080Ti. Additionally, we test two edge computing devices for autonomous driving. One is the GeForce RTX 3060 in the Apollo D-KIT Advanced [4], a kit for autonomous driving simulation, and the other is the Jetson AGX Orin [45], which is commonly used in real-world vehicles [46, 47].

4.2 Performance of Adversarial Training (RQ1)

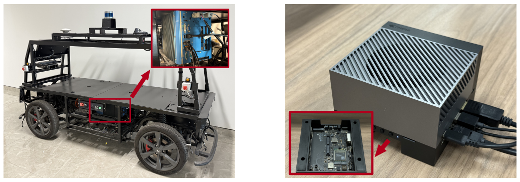
We perform adversarial training in a distributed scenario with 15 clients, following default settings. For each client, we evaluate performance by calculating the average accuracy on benign data (BA) and adversarial robustness (AR) across all clients under various attack algorithms. As baselines, we integrate local adversarial training with FedAvg (FA) and FedProx (FP).

We evaluate on four datasets, reporting average performance across all clients (Table 2). In heterogeneous settings, non-IID-aware baselines like FedProx variants and DBFAT show relatively better performance. *Sylva* consistently outperforms across diverse datasets, excelling on both benign and adversarial samples. On datasets like CIFAR-10, it shows substantial gains over both traditional federated adversarial training and personalized methods. Specifically, *Sylva* leads to an accuracy increase of up to 50.4% and a robustness enhancement of 29.5%, demonstrating its superior capability in enhancing both performance and resilience against adversarial attacks. When compared to the baselines of the two personalized algorithms, *Sylva* continues to deliver significant improvements, achieving an average increase of 48.3% in benign accuracy and 17.5% in robustness. This performance boost can be attributed to *Sylva*'s greater adaptability, as it effectively balances generalization across tasks with maximizing personalization for individual clients, thereby enhancing both accuracy and robustness. While the improvements on more complex datasets like CIFAR-100 are comparatively smaller, they remain significant, highlighting *Sylva*'s adaptability and effectiveness in varied scenarios. Furthermore, we evaluate *Sylva* on models of varying scales, with detailed results provided in Appendix B. *Sylva* consistently delivers strong performance across all model scales.

Answer to RQ1: By leveraging its personalized design, *Sylva* enables each client's model to effectively adapt to its specific data distribution, achieving superior accuracy and robustness even in highly heterogeneous environments.

Table 3: Detailed specifications of GPUs in real edge devices

	Type	Memory (GB)	Clock (MHz)	Bandwidth (GB/s)	FP16 (TFLOPS)
RTX 4090	GDDR6X	24	2235	1010	82.58
RTX 3090	GDDR6X	24	1395	936	35.58
RTX 2080-Ti	GDDR6	12	1350	768	30.14
RTX 3060	GDDR6	12	1320	360	12.74
AGX Orin	LPDDR5	32	930	204	10.65



(a) GeForce RTX 3060

(b) Jetson AGX Orin

Figure 8: Commonly used GPUs in autonomous driving scenarios

4.3 Performance of Different Heterogeneous Distributions (RQ2)

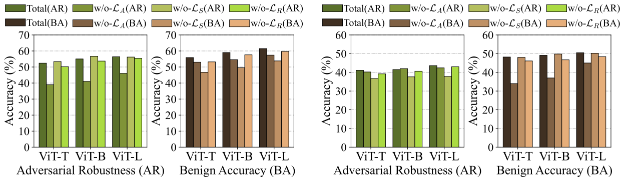
To evaluate the impact of varying levels of data heterogeneity on adversarial defense performance, we conduct experiments in five distinct scenarios: one with independent identical (IID) distribution and four with increasing heterogeneity levels determined by Dirichlet distribution parameters. Lower Dirichlet parameters indicate greater data heterogeneity, as detailed in Appendix A.4. For the attack algorithms, we use PGD and measure the benign accuracy and adversarial robustness metrics. The performance of *Sylva* and baseline models is assessed under these conditions across four datasets, with the results summarized in Fig. 7.

The results clearly show that as data heterogeneity increases, the overall performance of all algorithms declines, underscoring the significant influence of heterogeneity on training outcomes. Among the baselines, FedProx-based algorithms consistently outperform FedAvg-based ones, benefiting from their regularization mechanism that better accommodates heterogeneous data distributions. Although the two personalized baselines show good performance, they do not fully exploit the generalization potential of the model’s backbone, nor are their loss functions adequately aligned with the threat model for adversarial training. Consequently, they still fall behind *Sylva* in terms of effectiveness. Notably, *Sylva* consistently outperforms other methods and exhibits greater resilience to increasing heterogeneity. All experiments above are conducted using the ViT-B architecture, and results for other model scales are provided in Appendix C.

Answer to RQ2: *Sylva*’s personalized framework enhances robustness by minimizing data heterogeneity, ensuring strong performance across varying heterogeneous environments.

4.4 Comparison of Adversarial Training Efficiency (RQ3)

Given that *Sylva* is designed for deployment in distributed systems with multiple edge devices, evaluating its efficiency in real-world environments is essential. To simulate a real distributed system, we employ three widely used GPUs—NVIDIA GeForce RTX 4090, 3090, and 2080-Ti—on edge devices, while leveraging a high-speed subnet server as the cloud. To align with real-world scenarios, we also explore the deployment of *Sylva* in autonomous driving, including tasks like road sign recognition, as described in our threat model. We test *Sylva* on both the autonomous driving simulation device, specifically the RTX 3060 in the Apollo D-KIT Advanced [4], and the widely used Jetson AGX Orin in real vehicles [45]. The specifications of the GPUs used are shown in Table 3. We present key GPU parameters, including memory size, clock frequency, bandwidth, and the 16-bit floating-point (FP16) tera floating-point operations per second (TFLOPS), which are crucial for evaluating computational performance, with TFLOPS serving as a key indicator of computational power. Based on these parameters, theoretical analysis indicates that the consumer-grade GPUs—RTX 4090, 3090, and 2080-Ti—deliver excellent performance. In autonomous driving scenarios, the RTX 3060 performs slightly less effectively, as the Apollo D-KIT Advanced is primarily designed for vehicle task simulation. In contrast, real-world vehicles are typically equipped with 2-4 Jetson AGX Orin units [46, 47], allowing parallel deployment to approach the computational power of the RTX 3090 per vehicle.



(a) CIFAR-10

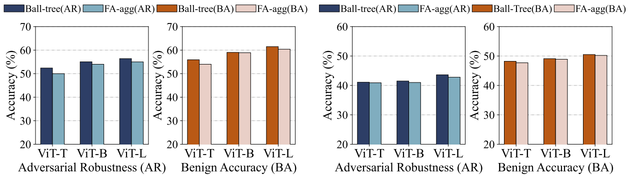
(b) STL-10

Figure 9: Impact of different loss modules on adversarial robustness and benign accuracy on CIFAR-10 and STL-10

In this setup, we assess *Sylva*’s efficiency by measuring its memory overhead (Mem), per-epoch training time (Time), and model upload communication time (Com) on edge devices. These metrics are compared against FA-TRADES (FAT) to provide a comprehensive evaluation. The results are shown in Table 4. Compared to traditional FAT algorithms, *Sylva* employs an efficient parameter fine-tuning approach for pre-trained models, offering significant reductions in memory usage, training time, and communication overhead. These benefits become more pronounced as model size increases. Specifically, *Sylva* reduces memory usage by up to 35.6%, training time by 26.7% on high-performance GPUs like the NVIDIA 4090, and up to 28.1% on resource-constrained devices such as the 2080-Ti. In terms of communication, *Sylva* achieves a 50× reduction in communication time, even with larger models and high-speed local networks, making it ideal for real-world distributed systems. Compared to Per-LoRA, *Sylva* shows no significant difference in memory usage, as both use LoRA for fine-tuning. However, *Sylva*’s unique personalization approach only requires the aggregation of the backbone’s LoRA parameters, while Per-LoRA also needs to upload the classifier’s parameters. This results in a communication savings of up to 64.5% for *Sylva*. Additionally, Per-LoRA’s trust weight computation adds complexity, while *Sylva*’s ball tree aggregation reduces complexity, leading to slightly faster training.

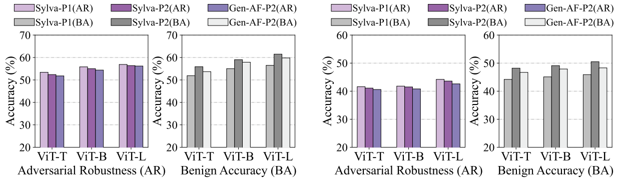
Table 4: Efficiency comparison of *Sylva* on different edge devices

		Mem↓ (G)	Time↓ ($\times 10^3$s)					Com↓ (s)				
			RTX 4090	RTX 3090	RTX 2080-Ti	RTX 3060	AGX Orin	RTX 4090	RTX 3090	RTX 2080-Ti	RTX 3060	AGX Orin
ViT-T	FAT	1.04	0.35	0.45	0.47	0.49	0.99	**0.7**	1.3	1.5	1.0	1.2
	Per-LoRA	0.89	0.31	0.32	0.33	0.32	0.82	0.9	0.7	**1.1**	1.1	1.0
	Sylva	**0.89**	**0.30**	**0.32**	**0.33**	**0.32**	**0.82**	0.8	**0.6**	1.2	**0.9**	**0.9**
ViT-B	FAT	4.86	0.40	0.47	0.49	0.53	1.13	15.0	13.5	13.8	14.1	13.8
	Per-LoRA	3.14	0.34	0.39	0.42	0.45	0.92	1.3	1.4	1.4	1.2	1.5
	Sylva	**3.14**	**0.33**	**0.37**	**0.41**	**0.45**	**0.91**	**0.8**	**0.9**	**0.8**	**0.7**	**0.8**
ViT-L	FAT	11.43	0.75	0.93	1.78	1.87	2.41	49.9	47.6	49.1	48.7	48.2
	Per-LoRA	6.63	0.57	0.88	1.29	1.39	1.71	3.1	2.9	3.2	3.4	3.0
	Sylva	**6.63**	**0.55**	**0.87**	**1.28**	**1.39**	**1.70**	**1.1**	**1.0**	**1.2**	**1.2**	**1.1**



(a) CIFAR-10 **(b) STL-10**

Figure 10: Impact of ball-tree-based aggregation algorithm on CIFAR-10 and STL-10



(a) CIFAR-10 **(b) STL-10**

Figure 11: Impact of different phases on the trade-off between robustness and accuracy for CIFAR-10 and STL-10

On the other hand, *Sylva* not only improves performance on high-end GPUs but also reduces training time and communication overhead on real edge devices. Experimental results closely match the computational power estimates derived from our theoretical analysis based on GPU parameters. Notably, for the Jetson AGX Orin, we only use a single unit for the experiments, but real-world vehicles typically use multiple units. Model or data parallelism could potentially double the training speed [12]. With 32GB of memory, the Orin can run *Sylva* alongside other autonomous driving tasks. Overall, *Sylva* outperforms baseline algorithms in real-world scenarios and is viable for practical deployment. In addition, we also test the communication efficiency when training with different numbers of clients, and the results are provided in Appendix D.

Answer to RQ3: *Sylva* significantly enhances distributed adversarial training by reducing memory usage, training time, and communication overhead, with its advantages growing as model size increases, making it highly efficient for real-world scenarios.

4.5 Ablation Experiments (RQ4)

4.5.1 Impact of Different Modules. We first evaluate the impact of our loss function, as illustrated in Fig. 9, by testing the total loss function and assessing adversarial robustness and benign accuracy without the $\mathcal{L}_A$, $\mathcal{L}_S$, and $\mathcal{L}_R$ components individually. The results clearly demonstrate that each component plays a positive role in enhancing overall performance. Among these, $\mathcal{L}_A$ and $\mathcal{L}_S$ prove to be particularly critical, as their absence leads to a substantial decline in both robustness and accuracy, underscoring their importance in the design of the loss function. We next evaluate the impact of the ball tree aggregation algorithm employed in Phase 1 of *Sylva* by comparing it with the traditional aggregation method used in FedAvg (FA-agg). As shown in Fig. 10, the proposed algorithm effectively mitigates model drift caused by data heterogeneity, resulting in superior performance in both adversarial robustness and benign accuracy. We further evaluate the effectiveness of the two-phase design on model robustness and benign accuracy. Phase 1 (P1) adversarial training generates a highly robust model, establishing a strong foundation for further refinement. In Phase 2 (P2), we compare *Sylva*'s algorithm with Gen-AF's Stage 2 method by training the robust models with their respective approaches. As shown in Fig. 11, Phase 2 achieves a substantial improvement in benign accuracy with only a minimal trade-off in adversarial robustness, clearly outperforming the Gen-AF approach. These results highlight the ability of *Sylva*'s two-phase design to effectively balance adversarial robustness and benign accuracy. The experiments discussed above are conducted on CIFAR-10 and STL-10. Additional results for GTSRB and CIFAR-100 are provided in Appendix E for further validation and analysis.

Table 5: The Impact of Hyperparameters in CIFAR-10

Para	Value	AR	BA	Para	Value	AR	BA
γ	0.9	55.03	59.03	β	20	56.87	57.27
	0.7	54.65	58.35		10	56.22	58.34
	0.5	54.70	57.25		5	55.03	59.03
	0.3	52.97	56.24		1	51.29	61.25
ϵ	0.9	55.03	59.03	B	50	49.24	53.98
	0.7	55.01	58.93		100	53.04	56.37
	0.5	54.31	58.35		300	55.03	59.03
	0.3	55.25	58.22		500	55.39	60.77
r	8	54.86	58.63	PGD Strength	3	48.04	61.26
	4	55.03	59.03		5	52.16	60.64
	2	55.42	58.36		10	55.03	59.03
	1	53.26	57.28		15	56.96	56.89

4.5.2 Impact of Different Hyperparameters. To investigate the impact of hyperparameters on *Sylva*, we conduct a comprehensive set of experiments on the ViT-B model using the CIFAR-10 dataset. This analysis evaluates the effects of key hyperparameters, including γ and ϵ in the Phase 1 loss function, β and B in Phase 2, the hidden space dimension r for LoRA, and the PGD strength employed in adversarial training. Table 5 presents the corresponding results for adversarial robustness and benign accuracy, providing a detailed comparison across different hyperparameter settings. The findings demonstrate that the default hyperparameter settings deliver strong overall performance, balancing robustness and accuracy effectively. Notably, smaller sampling sizes (B) in Phase 2 result in significantly worse outcomes, highlighting the importance of an adequate sampling strategy. In contrast, larger values of B show diminishing returns, with minimal performance gains and increased training time. Furthermore, the hidden space dimension r for LoRA shows minimal impact on the final results; even with $r = 1$, performance remains excellent. This observation aligns with LoRA's established efficiency and adaptability in traditional fine-tuning settings, further validating its applicability in adversarial training scenarios. Additionally, we evaluate the effects of these hyperparameters on three other datasets, as well as on scenarios with varying numbers of clients. The detailed results are provided in Appendix E.

Answer to RQ4: *Sylva* excels in balancing robustness and accuracy through its modular design, efficient aggregation, and adaptive hyperparameter tuning, validated across diverse datasets and scenarios.

5 DISCUSSION

This section explores four key directions for further improving *Sylva*, focusing on supporting heterogeneous deployment across devices with varying computational capacities, enhancing experimental design with larger-scale models, refining the threat model to include training-time attacks, and broadening its applicability to multi-modal training and diverse adversarial scenarios.

Adding Heterogeneous Model Support for Device Adaptation. The rapid advancement of AI models has driven industries to adopt increasingly powerful models for their applications. However, in industries such as the automotive sector, the long operational lifespan of vehicles (typically around 18 years [2]) leads to substantial variability in hardware capabilities across the system. As a result, not all vehicles are able to support the latest, most demanding models. To maximize performance, each vehicle should deploy the best model that its hardware can reliably support. Devices with varying computational capacities may require different model sizes, posing a challenge for maintaining the effectiveness of heterogeneous LoRA models fine-tuned via *Sylva*. Future work could explore methods for handling heterogeneous models within *Sylva*'s adversarial training, enhancing adaptability across diverse devices through dynamic scaling or pruning.

Expanding experiments with larger-scale models. Our algorithm has shown strong performance on traditional open-source ViT models. However, as model sizes continue to grow with the development of large-scale architectures, the robustness of massive vision models, such as LLaVA [32] and Qwen-VL [3] with hundreds of billions of parameters, remains an open question. Leveraging LoRA for fine-tuning, *Sylva* can be seamlessly integrated into these models, extending its applicability to unprecedented scales. While such experiments are resource-intensive, they represent a feasible and valuable direction for future exploration.

Strengthening the threat model. The current threat model assumes attacks occur during the inference phase in federated adversarial training. However, vulnerabilities may arise during training, even with secure devices, exposing models to potential risks. Addressing the detection and elimination of malicious clients during distributed training is a critical challenge. Integrating Byzantine defense techniques into *Sylva*'s Phase 1 offers a promising solution [13, 39, 48, 58], paving the way for future research to further enhance model robustness in this direction.

Adapting to multi-modal training and attacks. With the rapid evolution of large models in text [17] and multi-modal [54] applications, adversarial attack and defense algorithms are increasingly being developed to address their unique vulnerabilities [33, 76]. Integrating these defense mechanisms into *Sylva* can further enhance its adaptability, enabling efficient and task-specific adversarial training across diverse downstream applications. This integration represents a promising direction for expanding *Sylva*'s applicability and effectiveness.

6 RELATED WORKS

6.1 Adversarial Attack

Recent studies have focused on generating adversarial examples through subtle perturbations leading to misclassifications. Goodfellow et al. proposed FGSM [14], highlighting neural networks' vulnerability due to their linearity and introducing a simple adversarial training approach. Tramèr et al. improved robustness against black-box attacks with RFGSM [62], which integrates perturbations from multiple models. Madry et al. introduced PGD [40], providing robust optimization with security guarantees. Liu et al. developed EOTPGD [36], a Bayesian framework incorporating randomness for greater robustness. Croce et al. proposed APGD [8], a parameter-free ensemble attack for evaluating defense strategies. Su and Modas et al. introduced the One Pixel Attack [60] and Sparsefool [43], optimizing minimal pixel changes to reduce computational overhead. Ban et al. presented PAP [5], leveraging Low-Level Layer Lifting Attacks (L4A) to manipulate neuron activations in pre-trained models' lower layers. Zhou et al. introduced AdvCLIP [76], generating downstream-agnostic adversarial examples with cross-modal pre-trained encoders.

6.2 Adversarial Defence in Federated Learning

Federated learning (FL) enables distributed model training while preserving user privacy [34, 42, 69], but faces challenges due to statistical heterogeneity in device data distributions [38, 51, 73]. To overcome the limitations of global models in non-IID scenarios, personalized federated learning approaches have been proposed to train client-specific models [64, 71]. Many works leverage parameter-efficient methods in FL for LLMs, including adapter tuning [52], prompt learning [15], and localized adjustments [61]. With the continuous development of edge computing, many works have implemented adversarial attacks and defenses based on the FL framework [23, 49, 67]. Zizzo et al. investigated federated adversarial training, comparing model performance in both centralized and distributed FL settings [78]. Li et al. provided a convergence proof for this federated adversarial learning approach [30]. Zhou et al. addressed the aggregation error in FL, breaking it down into bias and variance [75]. Additionally, Chen et al. explored certified defenses against adversarial examples in FL, providing an alternative direction for enhancing robustness [6]. Zhang et al. proposed DBFAT [72], which adjusts sample weights based on PGD steps for generating adversarial examples. Hong et al. introduced a strategy propagating adversarial robustness from resource-rich to resource-poor users via batch normalization [18]. Qi et al. developed FairVFL, a privacy-preserving framework enhancing fairness in vertical federated learning through adversarial learning [50].

7 CONCLUSION

In this paper, we introduce *Sylva*, the first adversarial defense algorithm tailored for pre-trained models in distributed settings. *Sylva* delivers robust and accurate models even with imbalanced data distributions. Through preliminary experiments, we identify key challenges and highlight the advantages of efficient parameter fine-tuning. Building on this, we propose a two-phase training framework: Phase 1 utilizes a novel loss function and personalized federated fine-tuning for enhanced robustness, while Phase 2 applies a game-based layer freezing strategy to balance robustness and accuracy. Experiments demonstrate that *Sylva* not only outperforms state-of-the-art adversarial defenses in terms of robustness and accuracy but also does so with minimal computational overhead, making it suitable for edge deployment.

REFERENCES

[1] Youssef Allouah, Abdellah El Mrini, Rachid Guerraoui, Nirupam Gupta, and Rafael Pinot. 2024. Fine-tuning personalization in federated learning to mitigate adversarial clients. In *Proceedings of the Neural Information Processing Systems*. 100816–100844.

[2] ASME. 2025. Battery-Powered EVs Now Match Gas Cars in Lifespan. https://www.asme.org/topics-resources/content/new-study-finds-battery-powered-evs-now-match-gas-cars-in-lifespan.

[3] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966* (2023).

[4] Baidu. 2022. Apollo D-KIT Advanced. https://apollo.baidu.com/community/apollo_d_kit.

[5] Yuanhao Ban and Yinpeng Dong. 2022. Pre-trained adversarial perturbations. In *Proceedings of the Neural Information Processing Systems*. 1196–1209.

[6] Chen Chen, Yuchen Liu, Xingjun Ma, and Lingjuan Lyu. 2022. Calfat: Calibrated federated adversarial training with label skewness. In *Proceedings of the Neural Information Processing Systems*. 3569–3581.

[7] Adam Coates, Andrew Ng, and Honglak Lee. 2011. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*. 215–223.

[8] Francesco Croce and Matthias Hein. 2020. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *Proceedings of the International Conference on Machine Learning*. 2206–2216.

[9] Zizhuang Deng, Kai Chen, Guozhu Meng, Xiaodong Zhang, Ke Xu, and Yao Cheng. 2022. Understanding real-world threats to deep learning models in android apps. In *Proceedings of the Conference on Computer and Communications Security*. 785–799.

[10] Mohamad Dolatshah, Ali Hadian, and Behrouz Minaei-Bidgoli. 2015. Ball*-tree: Efficient spatial indexing for constrained nearest-neighbor search in metric spaces. *arXiv preprint arXiv:1511.00628* (2015).

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of the International Conference on Learning Representations*.

[12] Shiqing Fan, Yi Rong, Chen Meng, Zongyan Cao, Siyu Wang, Zhen Zheng, Chuan Wu, Guoping Long, Jun Yang, Lixue Xia, et al. 2021. DAPPLE: A pipelined data parallel approach for training large models. In *Proceedings of the Principles and Practice of Parallel Programming*. 431–445.

[13] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. 2020. Local model poisoning attacks to byzantine-robust federated learning. In *Proceedings of the USENIX Security Symposium*. 1605–1622.

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

[15] Tao Guo, Song Guo, Junxiao Wang, Xueyang Tang, and Wenchao Xu. 2023. Promptfl: Let federated participants cooperatively learn prompts instead of models-federated learning in age of foundation model. *IEEE Transactions on Mobile Computing* 23, 5 (2023), 5179–5194.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 770–778.

[17] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556* (2022).

[18] Junyuan Hong, Haotao Wang, Zhangyang Wang, and Jiayu Zhou. 2023. Federated robustness propagation: sharing adversarial robustness in heterogeneous federated learning. In *Proceedings of the Association for the Advancement of Artificial Intelligence*. 7893–7901.

[19] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the International Conference on Machine Learning*. 2790–2799.

[20] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *Proceedings of the International Conference on Learning Representations*.

[21] Jinyuan Jia, Yupei Liu, and Neil Zhenqiang Gong. 2022. Badencoder: Backdoor attacks to pre-trained encoders in self-supervised learning. In *Proceedings of the Symposium on Security and Privacy*. 2043–2059.

[22] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. 2021. Highly accurate protein structure prediction with AlphaFold. *Nature* 596, 7873 (2021), 583–589.

[23] Torsten Krauß and Alexandra Dmitrienko. 2023. Mesas: Poisoning defense for federated learning resilient against adaptive attackers. In *Proceedings of the Conference on Computer and Communications Security*. 1526–1540.

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

[25] Saehyung Lee, Hyungyu Lee, and Sungroh Yoon. 2020. Adversarial vertex mixup: Toward better adversarially robust generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 272–281.

[26] Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. 2014. Scaling distributed machine learning with the parameter server. In *Proceedings of the Operating Systems Design and Implementation*. 583–598.

[27] Qinbin Li, Bingsheng He, and Dawn Song. 2023. Adversarial collaborative learning on non-iid features. In *Proceedings of the International Conference on Machine Learning*. 19504–19526.

[28] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated optimization in heterogeneous networks. In *Proceedings of the Machine Learning and Systems*. 429–450.

[29] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. 2020. On the convergence of fedavg on non-iid data. In *Proceedings of the International Conference on Learning Representations*.

[30] Xiaoxiao Li, Zhao Song, and Jiaming Yang. 2023. Federated adversarial learning: A framework with convergence analysis. In *Proceedings of the International Conference on Machine Learning*. 19932–19959.

[31] Hongbin Liu, Jinyuan Jia, and Neil Zhenqiang Gong. 2022. PoisonedEncoder: Poisoning the unlabeled pre-training data in contrastive learning. In *Proceedings of the USENIX Security Symposium*. 3629–3645.

[32] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. In *Proceedings of the Neural Information Processing Systems*. 34892–34916.

[33] Kai Liu, Xin Liu, An Yang, Jing Liu, Jinsong Su, Sujian Li, and Qiaoqiao She. 2020. A robust adversarial training approach to machine reading comprehension. In *Proceedings of the Association for the Advancement of Artificial Intelligence*. 8392–8400.

[34] Lumin Liu, Jun Zhang, Shenghui Song, and Khaled B Letaief. 2022. Hierarchical federated learning with quantization: Convergence analysis and system design. *IEEE Transactions on Wireless Communications* 22, 1 (2022), 2–18.

[35] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. In *Proceedings of the Association for Computational Linguistics*. 61–68.

[36] Xuanqing Liu, Yao Li, Chongruo Wu, and Cho-Jui Hsieh. 2018. Adv-bnn: Improved adversarial defense through robust bayesian neural network. *arXiv preprint arXiv:1810.01279* (2018).

[37] Zechun Liu, Changsheng Zhao, Forrest Iandola, Chen Lai, Yuandong Tian, Igor Fedorov, Yunyang Xiong, Ernie Chang, Yangyang Shi, Raghuraman Krishnamoorthi, et al. 2024. Mobilellm: Optimizing sub-billion parameter language models for on-device use cases. In *Proceedings of the International Conference on Machine Learning*.

[38] Mi Luo, Fei Chen, Dapeng Hu, Yifan Zhang, Jian Liang, and Jiashi Feng. 2021. No fear of heterogeneity: Classifier calibration for federated learning with non-iid data. In *Proceedings of the Neural Information Processing Systems*. 5972–5984.

[39] Zhuoran Ma, Jianfeng Ma, Yinbin Miao, Yingjiu Li, and Robert H Deng. 2022. ShieldFL: Mitigating model poisoning attacks in privacy-preserving federated learning. *IEEE Transactions on Information Forensics and Security* 17 (2022), 1639–1654.

[40] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *Proceedings of the International Conference on Learning Representations*.

[41] Aboli Marathe, Deva Ramanan, Rahee Walambe, and Ketan Kotecha. 2023. Wedge: A multi-weather autonomous driving dataset built from generative vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3318–3327.

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

[43] Apostolos Modas, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. 2019. Sparsefool: a few pixels make a big difference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9087–9096.

[44] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I Jordan, et al. 2018. Ray: A distributed framework for emerging AI applications. In *Proceedings of the Operating Systems Design and Implementation*. 561–577.

[45] NVIDIA. 2021. Jetson AGX Orin. https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin/.

[46] NVIDIA. 2024. AI Drives Future of Transportation at Asia's Largest Automotive Show. https://blogs.nvidia.com/blog/nvidia-partners-auto-china/.

[47] NVIDIA. 2024. Wave of EV Makers Choose NVIDIA DRIVE for Automated Driving. https://nvidianews.nvidia.com/news/wave-of-ev-makers-choose-nvidia-drive-for-automated-driving.

[48] Mustafa Safa Ozdayi, Murat Kantarcioglu, and Yulia R Gel. 2021. Defending against backdoors in federated learning with robust learning rate. In *Proceedings of the Association for the Advancement of Artificial Intelligence*. 9268–9276.

[49] Dario Pasquini, Danilo Francati, and Giuseppe Ateniese. 2022. Eluding secure aggregation in federated learning via model inconsistency. In *Proceedings of the Conference on Computer and Communications Security*. 2429–2443.

[50] Tao Qi, Fangzhao Wu, Chuhan Wu, Lingjuan Lyu, Tong Xu, Hao Liao, Zhongliang Yang, Yongfeng Huang, and Xing Xie. 2022. Fairvfl: A fair vertical federated learning framework with contrastive adversarial learning. In *Proceedings of the Neural Information Processing Systems*. 7852–7865.

[51] Tianyu Qi, Yufeng Zhan, Peng Li, Jingcai Guo, and Yuanqing Xia. 2023. Hwamei: A learning-based synchronization scheme for hierarchical federated learning. In *Proceedings of the International Conference on Distributed Computing Systems*. 534–544.

[52] Tianyu Qi, Yufeng Zhan, Peng Li, and Yuanqing Xia. 2024. Tomtit: Hierarchical Federated Fine-Tuning of Giant Models based on Autonomous Synchronization. In *Proceedings of the Conference on Computer Communications*. 1910–1919.

[53] Tianyu Qi, Yufeng Zhan, Peng Li, and Yuanqing Xia. 2025. Robin: An Efficient Hierarchical Federated Learning Framework Via a Learning-Based Synchronization Scheme. *IEEE Transactions on Cloud Computing* (2025).

[54] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the International Conference on Machine Learning*. 8748–8763.

[55] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision* 115 (2015), 211–252.

[56] Dayana Savostianova, Emanuele Zangrando, and Francesco Tudisco. 2024. Low-Rank Adversarial PGD Attack. *arXiv preprint arXiv:2410.12607* (2024).

[57] Lloyd S Shapley. 1953. A value for n-person games. *Contribution to the Theory of Games* 2 (1953).

[58] Virat Shejwalkar and Amir Houmansadr. 2021. Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning. In *Proceedings of the Network and Distributed System Symposium*.

[59] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. 2011. The German traffic sign recognition benchmark: a multi-class classification competition. In *Proceedings of the International Joint Conference on Neural Networks*. 1453–1460.

[60] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. 2019. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation* 23, 5 (2019), 828–841.

[61] Guangyu Sun, Umar Khalid, Matias Mendieta, Taojiannan Yang, Pu Wang, Minwoo Lee, and Chen Chen. 2024. Exploring parameter-efficient fine-tuning to enable foundation models in federated learning. In *Proceedings of the International Conference on Big Data*. 8015–8024.

[62] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. 2018. Ensemble adversarial training: Attacks and defenses. In *Proceedings of the International Conference on Learning Representations*.

[63] Nicolas Wagner, Dongyang Fan, and Martin Jaggi. 2024. Personalized collaborative fine-tuning for on-device large language models. In *Proceedings of the Conference on Language Modeling*.

[64] Jiaqi Wang, Xingyi Yang, Suhan Cui, Liwei Che, Lingjuan Lyu, Dongkuan DK Xu, and Fenglong Ma. 2024. Towards personalized federated learning via heterogeneous model reassembly. In *Proceedings of the Neural Information Processing Systems*. 29515–29531.

[65] Luyuan Xie, Manqing Lin, Tianyu Luan, Cong Li, Yuejian Fang, Qingni Shen, and Zhonghai Wu. 2024. MH-pFLID: Model heterogeneous personalized federated learning via injection and distillation for medical data analysis. In *Proceedings of the International Conference on Machine Learning*. 54561–54575.

[66] Honghui Yang, Sha Zhang, Di Huang, Xiaoyang Wu, Haoyi Zhu, Tong He, Shixiang Tang, Hengshuang Zhao, Qibo Qiu, Binbin Lin, et al. 2024. Unipad: A universal pre-training paradigm for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 15238–15250.

[67] Abbas Yazdinejad, Ali Dehghantanha, Hadis Karimipour, Gautam Srivastava, and Reza M Parizi. 2024. A robust privacy-preserving federated learning model against model poisoning attacks. *IEEE Transactions on Information Forensics and Security* 19 (2024), 6693–6708.

[68] Jinliang Yuan, Chen Yang, Dongqi Cai, Shihe Wang, Xin Yuan, Zeling Zhang, Xiang Li, Dingge Zhang, Hanzi Mei, Xianqing Jia, et al. 2024. Mobile foundation model as firmware. In *Proceedings of the International Conference on Mobile Computing and Networking*. 279–295.

[69] Yufeng Zhan, Peng Li, Zhihao Qu, Deze Zeng, and Song Guo. 2020. A learning-based incentive mechanism for federated learning. *IEEE Internet of Things Journal* 7, 7 (2020), 6360–6368.

[70] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. 2019. Theoretically principled trade-off between robustness and accuracy. In *Proceedings of the International Conference on Machine Learning*. 7472–7482.

[71] Jianqing Zhang, Yang Hua, Hao Wang, Tao Song, Zhengui Xue, Ruhui Ma, and Haibing Guan. 2023. Fedala: Adaptive local aggregation for personalized federated learning. In *Proceedings of the Association for the Advancement of Artificial Intelligence*. 11237–11244.

[72] Jie Zhang, Bo Li, Chen Chen, Lingjuan Lyu, Shuang Wu, Shouhong Ding, and Chao Wu. 2023. Delving into the adversarial robustness of federated learning. In *Proceedings of the Association for the Advancement of Artificial Intelligence*. 11245–11253.

[73] Shulai Zhang, Zirui Li, Quan Chen, Wenli Zheng, Jingwen Leng, and Minyi Guo. 2021. Dubhe: Towards data unbiasedness with homomorphic encryption in federated learning client selection. In *Proceedings of the International Conference on Parallel Processing*. 1–10.

[74] Hao Zhou, Haoyu Wang, Shuohan Wu, Xiapu Luo, Yajin Zhou, Ting Chen, and Ting Wang. 2021. Finding the missing piece: Permission specification analysis for android NDK. In *Proceedings of the International Conference on Automated Software Engineering*. IEEE, 505–516.

[75] Yao Zhou, Jun Wu, Haixun Wang, and Jingrui He. 2022. Adversarial robustness through bias variance decomposition: A new perspective for federated learning. In *Proceedings of the International Conference on Information & Knowledge Management*. 2753–2762.

[76] Ziqi Zhou, Shengshan Hu, Minghui Li, Hangtao Zhang, Yechao Zhang, and Hai Jin. 2023. Advclip: Downstream-agnostic adversarial examples in multimodal contrastive learning. In *Proceedings of the International Conference on Multimedia*. 6311–6320.

[77] Ziqi Zhou, Minghui Li, Wei Liu, Shengshan Hu, Yechao Zhang, Wei Wan, Lulu Xue, Leo Yu Zhang, Dezhong Yao, and Hai Jin. 2024. Securely fine-tuning pre-trained encoders against adversarial examples. In *Proceedings of the Symposium on Security and Privacy*. 3015–3033.

[78] Giulio Zizzo, Ambrish Rawat, Mathieu Sinn, and Beat Buesser. 2020. Fat: Federated adversarial training. In *Proceedings of the Neural Information Processing Systems Workshop*.

A DETAILS OF EXPERIMENTAL SETUP

A.1 Datasets and Models

In this experiment, we select several datasets that are commonly employed in adversarial training research. These datasets encompass a variety of characteristics, including differing numbers of categories, distinct application scenarios, and varying dataset scales. A detailed overview of each dataset is presented below:

- CIFAR-10 [24]: A dataset of 60,000 32×32 color images across 10 classes, with 50,000 for training and 10,000 for testing.
- STL-10 [7]: A dataset with 10 object classes, consisting of 5,000 training and 8,000 test images at 96×96 resolution, commonly used for unsupervised and semi-supervised learning benchmarks.
- GTSRB [59]: A traffic sign recognition dataset with over 50,000 images spanning 43 classes, serving as a benchmark for intelligent transportation systems.
- CIFAR-100 [24]: Similar to CIFAR-10 but with 100 classes, comprising 60,000 images (600 per class), offering a greater challenge in image classification.

The Vision Transformer (ViT) is a transformer-based model for image classification, treating images as sequences of patches. We use three pre-trained variants: ViT-Tiny (ViT-T/16), ViT-Base (ViT-B/16), and ViT-Large (ViT-L/16), differing in scale and complexity. Their configurations, summarized in Table 6, enable performance evaluation under varying computational demands.

A.2 Attacks

Building on the threat model outlined in Section 2, attacker methods can be broadly divided into white-box and gray-box attacks. In our experiments, we focus on four representative attack methods, detailed as follows:

- FGSM [14]: A white-box attack that perturbs inputs along the gradient direction of the loss function, efficiently generating small but impactful adversarial examples.
- PGD [40]: An iterative extension of FGSM, applying multiple small perturbations projected back into the allowed space for a more thorough adversarial optimization.
- SparseFool [43]: Focuses on creating sparse perturbations by minimizing pixel changes, offering effective attacks with high computational efficiency for high-dimensional data.
- PAP [5]: A gray-box attack targeting pre-trained models, effective even against fine-tuned models by leveraging low-level layer manipulations and noise augmentation for strong transferability.

The aforementioned attack algorithms are applied to a range of datasets, with one representative image selected from each dataset for visualization purposes. Results on ViT-B are presented in Fig. 12. These algorithms are straightforward to implement and effectively satisfy the inconspicuousness criteria outlined in the threat model, ensuring minimal perceptual differences while achieving their intended effects.

Table 6: Details of vision transformer models

Model	Layers	Hidden Size	Intermediate Size	Heads
ViT-T/16	12	192	768	3
ViT-B/16	12	768	3072	12
ViT-L/16	24	1024	4096	16

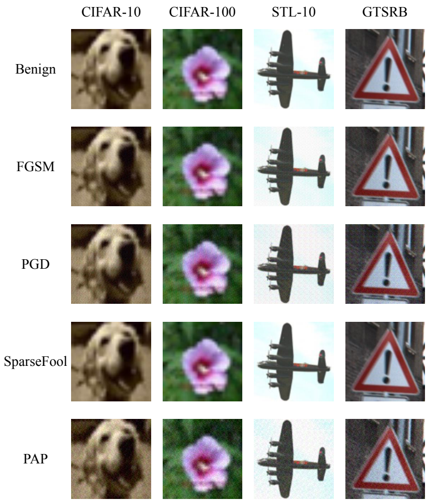


Figure 12: Visualization under different attack algorithms

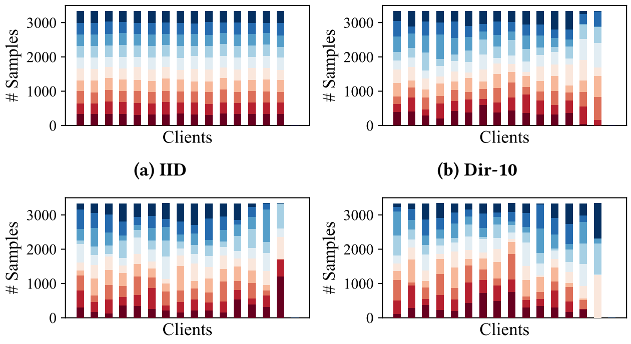
A.3 Baselines

To demonstrate the superiority of *Sylva*, we first examine commonly used and novel defense algorithms. For local defense, the following three algorithms are employed:

- PGT-AT [40]: PGD-AT employs the PGD algorithm to generate adversarial examples during training, thereby enhancing model robustness. By repeatedly exposing the model to adversarial perturbations, it improves the model's ability to resist various attacks effectively.
- TRADES [70]: TRADES achieves a balance between accuracy on benign data and robustness against adversarial attacks. It minimizes a carefully designed loss function that combines the standard classification loss with a penalty term measuring output discrepancies between benign and adversarial examples.
- Gen-AF [77]: Gen-AF improves model robustness through a two-stage fine-tuning approach for pre-trained models. It first optimizes the encoder using genetic regularization, followed by the selection and refinement of robust layers, leading to enhanced accuracy and robustness across diverse datasets.

In the distributed multi-client scenario addressed in this paper, we integrate commonly used federated learning algorithms with the aforementioned local defense strategies to enable effective deployment. The specific federated learning algorithms employed are as follows:

- FedAvg [42]: FedAvg is a federated learning algorithm where each client trains a local model and shares updates with the



(a) IID

(b) Dir-10

(c) Dir-6

(d) Dir-3

Figure 13: Visualization of Dirichlet distribution

server, which computes a weighted average to update the global model.

- FedProx [28]: FedProx extends FedAvg by adding a proximal term to the loss function, mitigating the impact of heterogeneous data across clients. This regularization ensures that local models remain close to the global model, improving convergence and robustness.
- DBFAT [72]: DBFAT enhances adversarial robustness in federated learning through local re-weighting and global regularization, improving both accuracy and robustness, especially in non-IID settings.
- Per-Adv [1]: Per-Adv enhances robustness against Byzantine attacks by introducing a personalized loss function, wherein each client interacts with others to collaboratively construct a robust interpolated objective.
- Per-LoRA [63]: Per-LoRA leverages LoRA-based fine-tuning to perform local updates of large models, while additionally computing trust weights from each client's LoRA parameters, which are used to guide the personalized update of the local model.

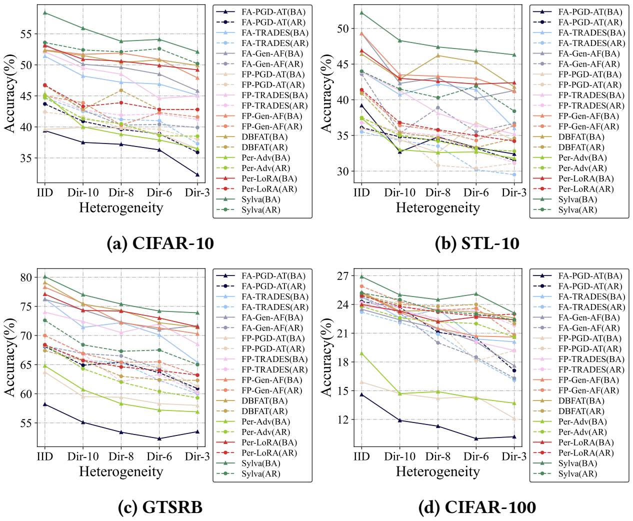
Specifically, we integrate the FedAvg and FedProx algorithms with the three aforementioned local defense algorithms, yielding the following configurations: FA-PGD-AT, FA-TRADES, and FA-Gen-AF, as well as FP-PGD-AT, FP-TRADES, and FP-Gen-AF. These combinations leverage the strengths of both federated optimization strategies and local defense mechanisms to enhance model robustness. For the two personalized algorithms, Per-Adv and Per-LoRA, we apply appropriate optimizations to adapt them to our specific tasks. We utilize the TRADES loss function for both algorithms to facilitate adversarial training. In the case of Per-Adv, we assume an adversary without Byzantine attacks to better align with the threat model outlined in this study. For Per-LoRA, we employ strategy 2, as proposed by the authors, to compute the trust weights.

A.4 Data Heterogeneity

We adopt the widely used data heterogeneity partitioning method based on the Dirichlet distribution [29], which effectively captures the non-IID nature of data distributions typically observed in real-world scenarios. For the CIFAR-10 dataset, we distribute the data among 15 clients with varying Dirichlet distribution parameters, ensuring that both the training and test sets for each client follow the same distribution. This method realistically simulates data

Table 7: Performance comparison of *Sylva* and baseline under different datasets and attack algorithms (ViT-T/16)

Datasets	Baselines	Benign(BA)	FGSM	PGD	SparseFool	PAP
CIFAR-10	FA-PGD-AT	37.54	41.82	40.95	46.10	42.23
	FA-TRADES	48.24	43.10	42.58	44.15	41.48
	FA-Gen-AF	50.12	44.68	42.74	43.59	42.61
	FP-PGD-AT	39.91	42.35	41.62	45.86	42.48
	FP-TRADES	49.58	44.90	42.69	43.53	42.10
	FP-Gen-AF	51.73	45.50	43.91	45.57	44.91
	DBFAT	51.45	43.62	42.80	45.35	44.50
	Per-Adv	39.96	41.93	41.32	46.50	42.77
	Per-LoRA	50.89	45.90	43.03	45.21	44.30
	Sylva(Ours)	**55.92**	**50.02**	**52.45**	**50.25**	**49.58**
STL-10	FA-PGD-AT	32.81	36.12	34.89	38.45	37.67
	FA-TRADES	40.31	35.56	34.88	37.91	37.40
	FA-Gen-AF	42.29	36.85	35.30	37.91	36.77
	FP-PGD-AT	34.41	36.48	35.21	39.35	37.89
	FP-TRADES	41.10	37.21	35.22	37.00	36.88
	FP-Gen-AF	43.10	38.92	36.67	39.60	38.53
	DBFAT	42.75	36.77	35.48	39.14	38.11
	Per-Adv	32.92	36.83	35.01	38.25	38.01
	Per-LoRA	43.06	38.90	36.82	39.25	38.44
	Sylva(Ours)	**48.21**	**42.37**	**41.12**	**43.82**	**42.78**
GTSRB	FA-PGD-AT	55.73	63.52	64.25	69.12	68.22
	FA-TRADES	71.15	64.10	65.28	67.39	68.52
	FA-Gen-AF	74.85	65.49	66.10	67.50	68.10
	FP-PGD-AT	59.15	64.38	65.45	68.23	67.90
	FP-TRADES	72.30	65.75	66.52	67.37	68.20
	FP-Gen-AF	75.92	66.10	66.48	67.75	**69.71**
	DBFAT	75.40	65.38	65.53	66.75	68.95
	Per-Adv	60.73	63.06	64.25	67.21	67.53
	Per-LoRA	74.33	65.80	65.71	67.05	68.32
	Sylva(Ours)	**76.90**	**68.75**	**68.90**	**69.93**	69.03
CIFAR-100	FA-PGD-AT	11.89	24.32	23.75	25.92	24.70
	FA-TRADES	22.60	23.52	22.11	23.77	23.91
	FA-Gen-AF	22.39	24.00	23.12	24.30	23.68
	FP-PGD-AT	14.02	24.45	24.18	26.12	24.90
	FP-TRADES	23.41	22.78	23.75	25.18	24.05
	FP-Gen-AF	23.18	24.10	23.99	25.95	25.30
	DBFAT	23.28	23.89	23.98	25.50	26.88
	Per-Adv	14.65	23.87	22.64	23.85	24.04
	Per-LoRA	23.27	24.20	23.75	24.89	25.06
	Sylva(Ours)	**24.92**	**25.12**	**24.82**	**26.25**	**27.15**



(a) CIFAR-10

(b) STL-10

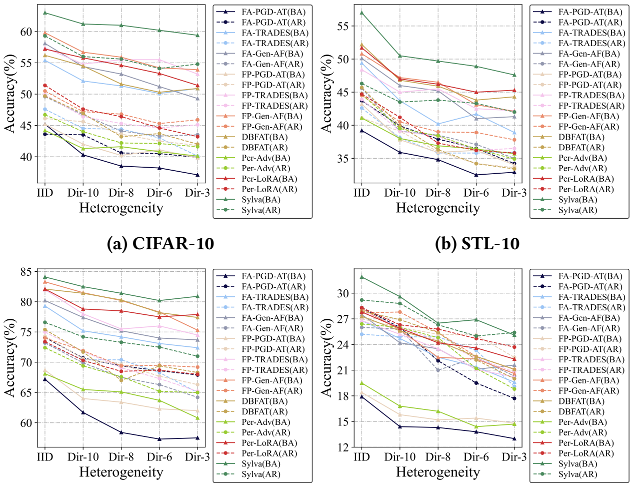
(c) GTSRB

(d) CIFAR-100

Figure 14: Comparison of adversarial training performance under different heterogeneity levels (ViT-T/16)

Table 8: Performance comparison of *Sylva* and baseline under different datasets and attack algorithms (ViT-L/16)

Datasets	Baselines	Benign(BA)	FGSM	PGD	SparseFool	PAP
CIFAR-10	FA-PGD-AT	40.50	47.21	43.89	49.45	45.58
	FA-TRADES	52.37	46.18	44.47	47.00	44.64
	FA-Gen-AF	54.75	48.17	46.87	47.27	46.78
	FP-PGD-AT	42.14	48.62	45.09	50.02	46.87
	FP-TRADES	54.17	48.02	46.68	48.78	46.56
	FP-Gen-AF	56.07	49.94	47.83	50.10	48.48
	DBFAT	54.92	47.01	46.13	50.20	49.72
	Per-Adv	41.34	47.35	44.02	48.35	45.66
	Per-LoRA	55.83	48.92	47.60	49.57	48.21
	Sylva(Ours)	**61.51**	**54.01**	**56.39**	**54.12**	**53.26**
STL-10	FA-PGD-AT	35.89	41.05	39.88	41.95	41.05
	FA-TRADES	43.70	39.15	38.11	41.80	40.93
	FA-Gen-AF	46.00	40.48	39.12	41.98	41.85
	FP-PGD-AT	37.78	41.77	39.14	42.51	41.69
	FP-TRADES	45.02	40.72	39.12	40.93	40.99
	FP-Gen-AF	47.20	41.03	40.49	42.12	43.02
	DBFAT	46.79	42.17	39.91	41.14	42.78
	Per-Adv	37.95	41.23	39.48	41.52	41.12
	Per-LoRA	46.95	41.78	41.20	42.05	42.11
	Sylva(Ours)	**50.51**	**45.01**	**43.52**	**46.47**	**45.42**
GTSRB	FA-PGD-AT	61.69	69.99	70.69	73.65	73.48
	FA-TRADES	75.24	69.58	70.40	72.34	74.15
	FA-Gen-AF	77.86	70.63	69.92	74.12	73.36
	FP-PGD-AT	63.98	71.57	70.66	73.62	72.71
	FP-TRADES	78.04	71.58	71.62	72.65	72.76
	FP-Gen-AF	81.14	72.81	71.83	73.95	75.07
	DBFAT	80.96	70.77	70.88	72.47	74.30
	Per-Adv	65.52	69.29	69.37	72.55	73.18
	Per-LoRA	78.83	71.52	70.25	72.36	74.01
	Sylva(Ours)	**82.51**	**74.39**	**74.20**	**75.64**	**75.84**
CIFAR-100	FA-PGD-AT	14.35	26.59	26.04	28.29	27.33
	FA-TRADES	24.73	25.50	24.93	25.95	26.14
	FA-Gen-AF	24.23	26.05	25.81	26.81	26.09
	FP-PGD-AT	15.82	26.47	26.04	28.86	27.46
	FP-TRADES	25.32	24.42	26.05	27.32	26.33
	FP-Gen-AF	25.70	26.11	27.89	28.15	27.30
	DBFAT	25.66	25.78	26.91	27.79	29.04
	Per-Adv	16.80	25.88	26.01	26.72	27.04
	Per-LoRA	25.97	25.99	26.32	27.85	28.03
	Sylva(Ours)	**29.62**	**28.33**	**28.87**	**29.77**	**29.44**



(a) CIFAR-10

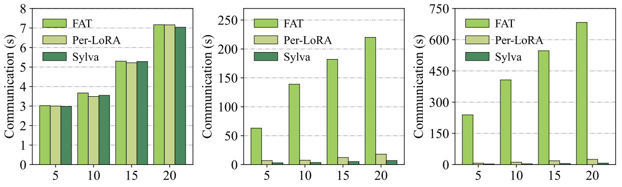
(b) STL-10

(c) GTSRB

(d) CIFAR-100

Figure 15: Comparison of adversarial training performance under different heterogeneity levels (ViT-L/16)

heterogeneity, providing a robust setup for evaluating federated learning algorithms in non-IID environments.



(a) ViT-T

(b) ViT-B

(c) ViT-L

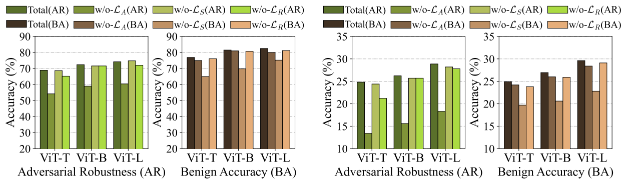
Figure 16: Relationship between the number of clients and communication time

The training set partitioning is visualized in Fig. 13, where the x-axis represents the clients and the y-axis shows the class sample counts, with different classes distinguished by colored bars. In the experiments, we use Dirichlet distribution parameters of 10, 6, and 3, along with an IID setup for comparison. Smaller Dirichlet parameters increase data heterogeneity, resulting in more imbalanced class distributions across clients. This approach effectively simulates realistic data heterogeneity by introducing varying class distributions, capturing natural class imbalances often encountered in real-world scenarios. It provides a more challenging evaluation setup, allowing for a comprehensive assessment of model robustness under diverse non-IID conditions, and tests the model's ability to generalize across clients with differing data distributions.

B ADDITIONAL RESULTS ON THE PERFORMANCE OF ADVERSARIAL TRAINING (RQ1)

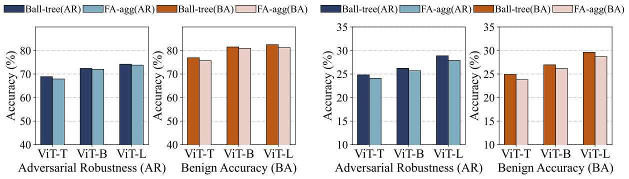
Table 2 provides a comprehensive comparison of *Sylva* against baseline models utilizing the ViT-B architecture across multiple datasets. The results emphasize *Sylva*'s superior performance in terms of both adversarial robustness and benign accuracy under a wide range of adversarial attack scenarios. Notably, *Sylva* consistently surpasses the baseline methods, demonstrating its capability to effectively manage diverse attack conditions while maintaining exceptional levels of adversarial reliability.

To further evaluate the adaptability of *Sylva*, we extend our analysis to models of varying sizes, specifically ViT-T and ViT-L, with detailed results presented in Table 7 and Table 8. The findings highlight that *Sylva* maintains strong defense capabilities across different model scales, delivering robust performance even as model complexity changes. *Sylva* outperforms both traditional distributed algorithms and personalized algorithms, consistently demonstrating superior performance in terms of both robustness and adaptability. Notably, while the defense capabilities of baseline methods often degrade significantly with smaller models, such as ViT-T, *Sylva* demonstrates exceptional adaptability. It consistently achieves an effective balance between adversarial robustness and benign accuracy, underscoring its versatility and resilience under varying conditions. These findings highlight the remarkable versatility of *Sylva*, demonstrating its suitability for a broad spectrum of model architectures and adversarial scenarios. Its consistent ability to deliver robust performance across diverse datasets and varying model scales underscores its potential as a dependable solution for adversarial training. This adaptability makes *Sylva* an effective choice for



(a) GTSRB **(b) CIFAR-100**

Figure 17: Impact of different loss modules on adversarial robustness and benign accuracy on GTSRB and CIFAR-100



(a) GTSRB **(b) CIFAR-100**

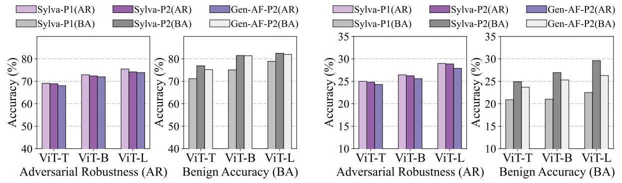
Figure 18: Impact of ball-tree-based aggregation algorithm on GTSRB and CIFAR-100

applications ranging from simple to highly complex environments, further cementing its value in enhancing model resilience.

C ADDITIONAL RESULTS ON THE PERFORMANCE UNDER DIFFERENT HETEROGENEITY LEVELS (RQ2)

In Section 4.3, we thoroughly analyze the adversarial defense performance of the ViT-B model under diverse heterogeneity conditions, with *Sylva* demonstrating remarkable defense capabilities and robustness throughout. To gain deeper insights into how models of varying scales respond to heterogeneity, we extend the evaluation to the ViT-T and ViT-L models, systematically assessing their defense performance under different levels of heterogeneity, as depicted in Fig. 14 and Fig. 15, respectively.

The results demonstrate that *Sylva* consistently outperforms its counterparts across various model sizes, effectively maintaining both accuracy and robustness, even in highly challenging heterogeneous environments where data distribution and computational resources vary significantly. Notably, smaller models, such as ViT-T, exhibit more rapid convergence during training, enabling them to achieve relatively stable performance despite the increasing heterogeneity of the data. This finding underscores *Sylva*'s adaptability in managing the intricate balance between model complexity and environmental variability. Moreover, compared to the other two personalized baseline algorithms, the smaller performance fluctuations observed under heightened heterogeneity indicate that *Sylva*'s personalized framework effectively mitigates the detrimental impacts of data heterogeneity by tailoring defense strategies to individual clients. These findings highlight the exceptional versatility of *Sylva*, making it an ideal choice for adversarial training across systems characterized by diverse heterogeneity and a broad spectrum of model sizes.



(a) GTSRB **(b) CIFAR-100**

Figure 19: Impact of different phases on the trade-off between robustness and accuracy for GTSRB and CIFAR-100

Table 9: The Impact of Hyperparameters in STL-10

Para	Value	AR	BA	Para	Value	AR	BA
γ	0.9	41.48	49.11	β	20	42.97	46.81
	0.7	41.26	48.83		10	42.50	47.25
	0.5	40.55	48.80		5	41.48	49.11
	0.3	40.12	48.05		1	40.79	50.02
ϵ	0.9	41.48	49.11	B	50	39.27	48.05
	0.7	40.96	49.05		100	41.10	47.92
	0.5	40.23	49.24		300	41.48	49.11
	0.3	40.57	48.76		500	41.76	50.21
r	8	41.72	49.34	PGD Strength	3	40.02	52.17
	4	41.48	49.11		5	41.07	50.81
	2	41.25	49.03		10	41.48	49.11
	1	40.27	48.65		20	42.52	47.33

D ADDITIONAL RESULTS ON EFFICIENCY COMPARISON ACROSS DIFFERENT SYSTEM SCALES (RQ3)

In Section 4.4, we emphasize the efficiency advantages of *Sylva* in real-world systems composed of diverse edge devices operating under adversarial training scenarios. To delve deeper into its communication efficiency, we perform additional experiments to assess the communication overhead in distributed systems of varying scales. By leveraging the edge devices described in Section 4.4, we simulate distributed systems with 5, 10, 15, and 20 clients, each performing adversarial training while concurrently communicating with the server. As the client count rises, network congestion emerges as a critical challenge, potentially undermining the timeliness and reliability of adversarial updates.

Fig. 16 presents a comparative analysis of communication time for distributed training across three distinct models under different client counts. The results clearly demonstrate a proportional increase in communication time as the number of clients grows, primarily due to heightened network congestion. Crucially, *Sylva* and Per-LoRA exhibit a significant advantage over traditional algorithms, particularly with larger models, by transmitting only the lightweight LoRA parameters. Since *Sylva* only transmits the backbone's LoRA parameters and keeps the classifier local to preserve personalization, it incurs lower communication overhead, making it more efficient for real-world deployments. This approach drastically reduces communication overhead, underscoring *Sylva*'s practicality and efficiency for deployment in real-world distributed systems. Its ability to mitigate communication challenges is especially valuable

Table 10: The Impact of Hyperparameters in GTSRB

Para	Value	AR	BA	Para	Value	AR	BA
γ	0.9	72.40	80.49	β	20	72.14	78.77
	0.7	72.34	79.25		10	72.80	80.04
	0.5	72.04	78.95		5	72.40	80.49
	0.3	71.83	78.24		1	70.92	81.94
ϵ	0.9	72.40	80.49	B	50	70.20	78.04
	0.7	72.37	78.37		100	71.29	80.82
	0.5	71.02	79.42		300	72.40	80.49
	0.3	71.25	78.10		500	72.43	79.95
r	8	72.66	79.42	PGD Strength	3	70.94	82.03
	4	72.40	80.49		5	71.02	81.72
	2	72.98	80.93		10	72.40	80.49
	1	71.62	78.39		20	73.97	77.92

Table 11: The Impact of Hyperparameters in CIFAR-100

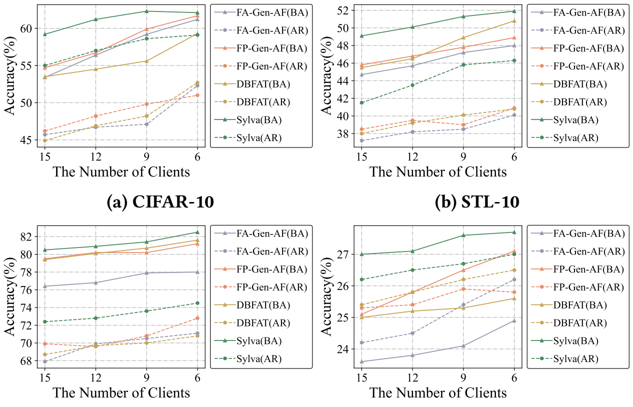
Para	Value	AR	BA	Para	Value	AR	BA
γ	0.9	26.23	26.95	β	20	26.54	25.10
	0.7	26.05	26.55		10	26.40	25.15
	0.5	26.09	26.72		5	26.23	26.95
	0.3	25.98	25.76		1	24.88	28.24
ϵ	0.9	26.23	26.95	B	50	25.87	25.90
	0.7	26.42	25.87		100	26.05	26.45
	0.5	25.87	25.58		300	26.23	26.95
	0.3	25.90	25.07		500	26.20	26.97
r	8	26.44	27.01	PGD Strength	3	25.04	28.03
	4	26.23	26.95		5	25.88	27.89
	2	26.30	27.04		10	26.23	26.95
	1	26.03	26.44		20	27.12	24.87

in scenarios involving large-scale models and high client densities, further cementing its suitability for such environments.

E ADDITIONAL ABLATION STUDY RESULTSN (RQ4)

In Section 4.5, we evaluate the impact of *Sylva*'s key modules on adversarial training models, focusing on CIFAR-10 and STL-10 datasets. This includes analyzing the loss function design, aggregation methods, and two-phase training framework, all critical to *Sylva*'s performance. To validate these findings, we extend experiments to GTSRB and CIFAR-100 datasets. The results, presented in Fig. 17, Fig. 18, and Fig. 19, confirm the consistent contributions of each module. These trends align with the main text, demonstrating *Sylva*'s robustness and strong generalization across datasets of varying complexity, highlighting its versatility in adversarial training scenarios.

We evaluate the impact of various hyperparameters, with detailed results shown in Table 9, Table 10, and Table 11. These findings align with those on CIFAR-10, revealing consistent trends across all datasets. Specifically, decreases in γ and ϵ lead to declines in both adversarial robustness and benign accuracy, underscoring their importance in maintaining model performance. In Phase 2, reducing β slightly improves benign accuracy but significantly reduces adversarial robustness, highlighting the trade-off between these metrics. These results stress the need for careful hyperparameter tuning to balance robustness and accuracy in adversarial training frameworks.



(a) CIFAR-10

(b) STL-10

(c) GTSRB

(d) CIFAR-100

Figure 20: Comparison of adversarial training performance under different number of clients (ViT-B/16)

To comprehensively evaluate the performance of the proposed framework, we conduct experiments under varying client numbers, as illustrated in Fig. 20. For comparison, we select several high-performing baseline algorithms, including FA-Gen-AF, FP-Gen-AF, and DBFAT, to ensure a robust analysis. In the multi-client scenario, the number of clients is systematically varied among 15, 12, 9, and 6, with experiments conducted on four distinct datasets to assess both adversarial robustness and benign accuracy comprehensively. The results reveal a clear trend: as the number of clients decreases, the overall performance improves, likely due to the reduced data heterogeneity associated with fewer clients. Notably, across all configurations, *Sylva* consistently outperforms the baseline algorithms, demonstrating superior results in both adversarial robustness and benign accuracy. These findings underscore the framework's effectiveness and adaptability, highlighting its capacity to excel across a diverse range of multi-client settings, irrespective of the dataset or client configuration.