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

ACM Conference on Computer and Communications Security (CCS) 2025

Tianyu Qi, Lei Xue, Yufeng Zhan, Xiaobo Ma
qity9@mail2.sysu.edu.cn

School of Cyber Science and Technology
Sun Yat-sen University

October 14, 2025

Outline

Introduction

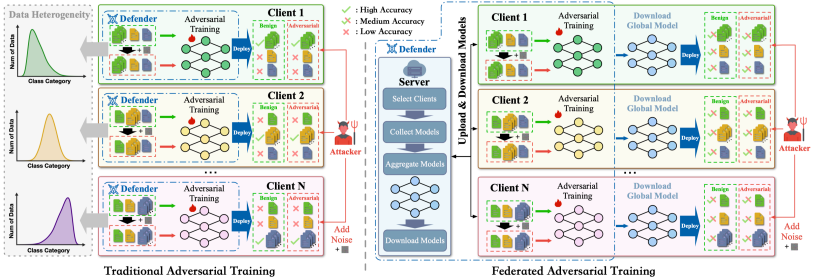


Figure: Challenges of adversarial training in multi-client scenario.

Backgrounds:

- Edge advances enable **on-device deployment** of large pre-trained models (latency, privacy).
- Local models face intensified probing and adaptive **adversarial attacks**, elevating reliability/security risk.

Recent challenges:

- **Adversarial training**: Non-IID client data hinders adversarial training, leading to degraded performance.
- **Federated adversarial training**: Large model size results in low training efficiency.

Introduction

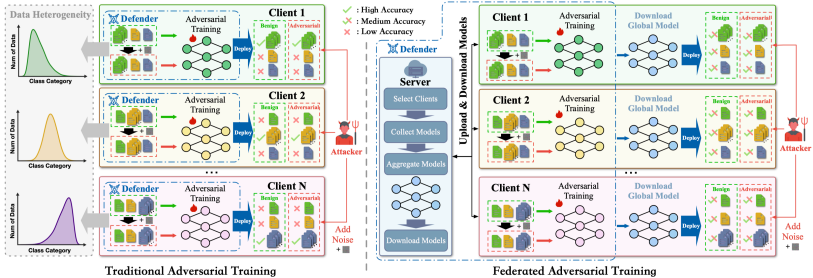


Figure: Challenges of adversarial training in multi-client scenario.

Our Contributions

- ***How to design a personalized defense framework?*** (Heterogeneous data distributions)
- ***How to enhance both adversarial robustness and benign accuracy?*** (Trade-off model performance)
- ***How to balance efficiency for clients?*** (Deployment on edge devices)

Outline

Preliminaries

Federated adversarial training (FAT):

- Federated learning on local device:

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

- Federated adversarial training on local device:

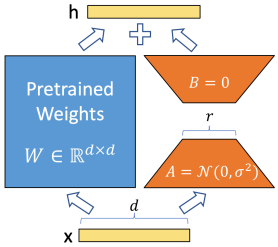
$$f_i(w_i, x, x^{adv}, y) =$$

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

- Aggregation on server:

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

Low-rank adaption (LoRA):



- Prediction based on pretrained backbone w^P, LoRA w^L, and clasffier w^C:

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

Preliminaries

Threat model:

Attacker categories

- **White-box attacks**: Access to all parameters.
- **Grey-box attacks**: Access to pre-trained parameters.

Attacker criteria

- **Inconspicuousness**: Perturbations go unnoticed by humans.
- **Impactfulness**: Compromise downstream tasks

Defender characteristics

- **Data Privacy**: Safeguard each client's privacy and security.
- **Heterogeneity**: Support personalized models in diverse scenarios.
- **Efficiency**: Accommodate the limited computational resources of clients.

Preliminaries

Impact of adversarial training with LoRA:

- We use ResNet-18 and ViT(ViT-T/16, ViT-B/16, ViT-L/16), applying the TRADES for adversarial fine-tuning via LoRA on CIFAR-10.
- Metrics: Adversarial Robustness, Benign Accuracy, Time, Parameter Size, GPU Memory.

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

Discovery 1

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

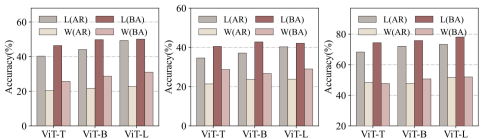
Preliminaries

Impact of the personalized framework:

RQ

- *Can we split each client model into adversarial and personalized components?*

- We simulate a non-IID environment on CIFAR-10, STL-10, and GTSRB using Dirichlet distribution.
- We use TRADES for federated adversarial fine-tuning via LoRA.
- Two strategies: uploading whole parameters (W) and uploading only LoRA (L).



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

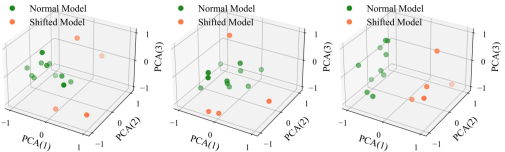
Discovery 2

- *Aggregating only the backbone (LoRA) and personalizing the classifier preserves generalized feature extraction while enhancing the classifier's sensitivity to heterogeneity.*

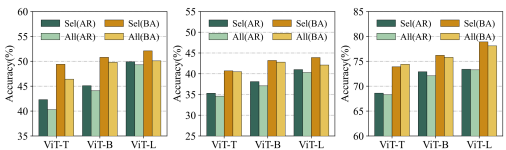
Preliminaries

Impact of the aggregation method:

- LoRA parameters may be impacted by non-IID data.
- We apply PCA to the LoRA parameters, reducing them to 3 dimensions.
- Two aggregation strategies: aggregate all models (All) and aggregate only closed models (Sel).



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



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

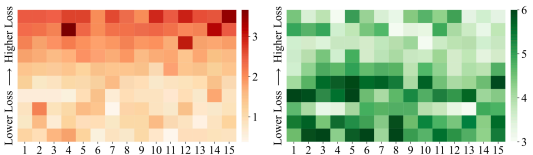
Discovery 3

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

Preliminaries

Limitations of the robustness-accuracy trade-off:

- Freezing low-robustness layers and retraining improves accuracy.
- Inter-layer interactions must be considered.
- Evaluate multiple layer combinations by summing and ranking losses.



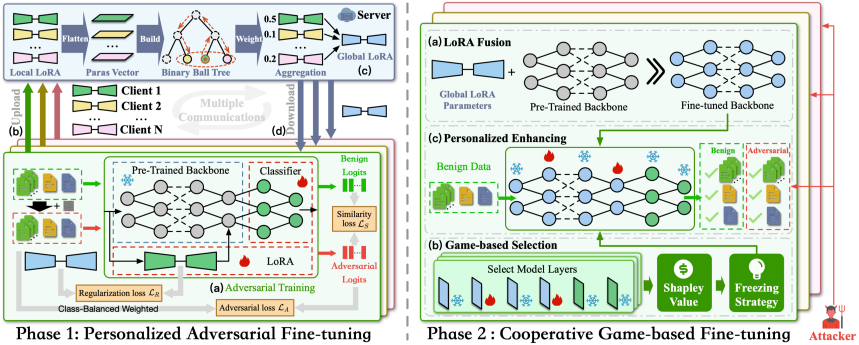
(a) Adversarial robustness (b) Benign accuracy

Discovery 4

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

Outline

Methodology



Overview:

- **Phase 1**: Federated LoRA fine-tuning yields a robust shared backbone and personalized classifiers.
- **Phase 2**: Shapley-guided layer freezing boosts accuracy with robustness preserved.

Methodology: Phase 1

Adaptive Class-balanced Dynamic Weighted Loss:

- The class imbalance weight:

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

- The dynamic smoothing weight:

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

- The smoothed class weights are normalized:

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

- The adversarial training loss:

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

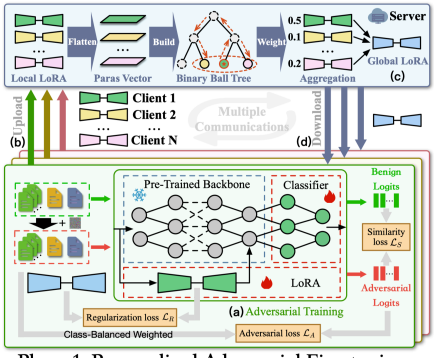


Figure: Overview of Phase 1.

Methodology: Phase 1

- A KL-divergence loss to align benign and adversarial samples:

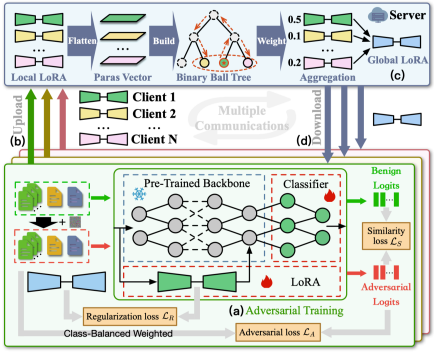
$$\mathcal{L}_S(w_i, x, x^{adv}) = \mathcal{L}_{KL}(\mathcal{F}(w_i, x), \mathcal{F}(w_i, x^{adv}))$$
$$= \sum \text{softmax}(\mathcal{F}(w_i, x)) \cdot \log\left(\frac{\text{softmax}(\mathcal{F}(w_i, x))}{\text{softmax}(\mathcal{F}(w_i, x^{adv}))}\right)$$

- A regularization loss to preserve generalization:

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

- Weight the above loss as:

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



Phase 1: Personalized Adversarial Fine-tuning

Figure: Overview of Phase 1.

Methodology: Phase 1

Ball-tree-based Aggregation:

- Flatten LoRA parameters to vectors:

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

- Identify the k vectors closest to i and compute the distance:

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

- Obtain the aggregation weights:

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

- The server weights the LoRA parameters:

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

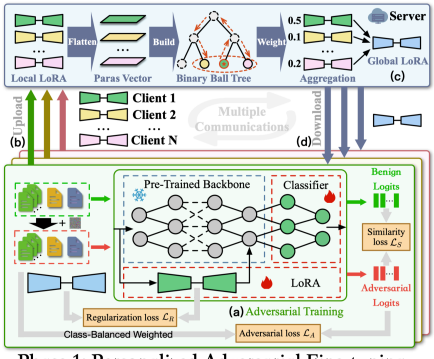


Figure: Overview of Phase 1.

Methodology: Phase 2

Shapley Game for Layers:

- Define two sensitivity losses as:

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

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

- The value function in the cooperative game:

$$v(\mathbf{S}) = \mathcal{L}_{acc}(\mathbf{S}) - \beta \cdot \mathcal{L}_{rob}(\mathbf{S})$$

- The Shapley value for each layer l:

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

- Select p layers $\mathbf{P} = \{l_1, l_2, \ldots, l_p\}$, where $\phi_{l_1} \geq \phi_{l_2} \geq \cdots \geq \phi_{l_p}$

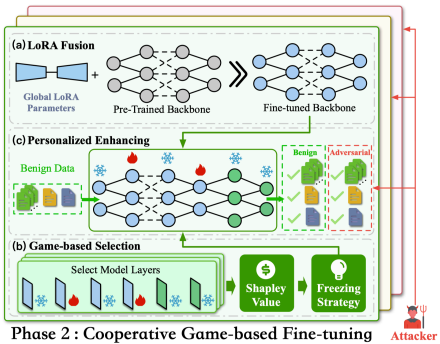


Figure: Overview of Phase 2.

Methodology: Phase 2

Monte Carlo Sampling Optimization:

- Randomly generate B permutations of the model layers, as $\pi^{(1)}, \pi^{(2)}, \ldots, \pi^{(B)}$.
- The marginal contribution is:

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

- Original complexity is $O(2^L \times L)$, reduced to $O(B \times L)$ using Monte Carlo Sampling, with the error decreasing at a rate of $O(1/\sqrt{B})$.

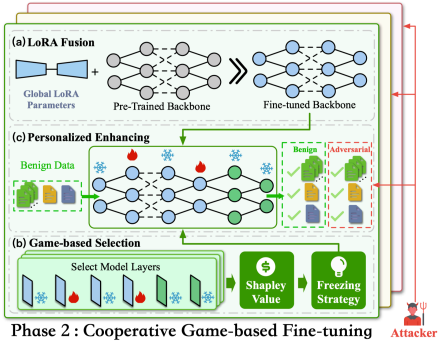


Figure: Overview of Phase 2.

Outline

Experiments

Experimental Setup:

- **Datasets and models.** CIFAR-10, STL-10, GTSRB, CIFAR-100; ViT-T/16, ViT-B/16, ViT-L/16.
- **Attacks.** FGSM, PGD, SparseFool, PAP.
- **Baselines.** Standard defense: PGD-AT, TRADES, Gen-AF; Distributed scenario: FedAvg, FedProx, DBFAT, Per-Adv, Per-LoRA.
- **Implementation.** Simulation: 4 NVIDIA A100 GPUs using Ray; Real world: GeForce RTX 4090, 3090, and 2080Ti, Apollo D-KIT, Jetson AGX Orin.

4 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?

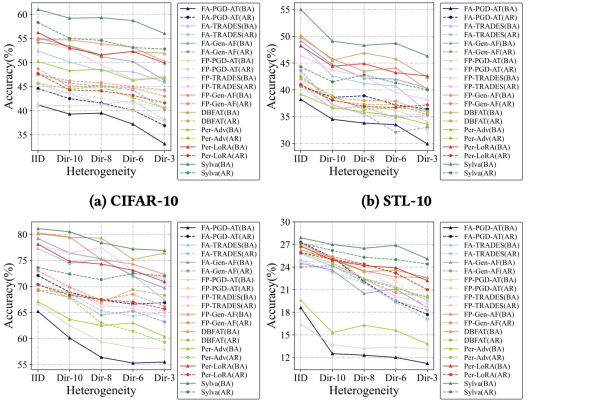
Experiments

Performance of adversarial training:

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**

Performance of different heterogeneous distributions:



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

(c) GTSRB (d) CIFAR-100

Experiments

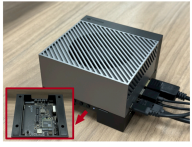
Comparison of adversarial training efficiency:

- Real-world GPU devices.

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

- Efficiency in real-world scenarios.

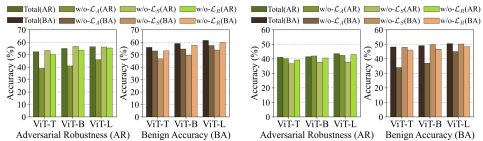
Table 4: Efficiency comparison of *Sylva* on different edge devices

		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**

Experiments

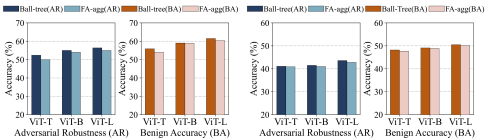
Ablation Experiments:

- Impact of loss function modules.



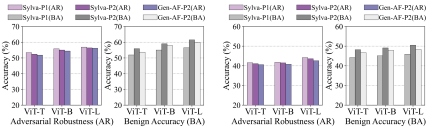
(a) CIFAR-10 (b) STL-10

- Impact of the ball tree aggregation.



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

- Impact of different phases.



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

- Impact of different hyperparameters.

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

Outline

Discussion and Conclusion

Discussion

- **Adding heterogeneous model support for device adaptation**: Different model size.
- **Expanding experiments with larger-scale models**: LLaVA or Qwen-VL.
- **Strengthening the threat model**: Threat during training.
- **Adapting to multi-modal training and attacks**: Different downstream tasks.

Conclusion

- **Novel contribution**: First adversarial defense for pre-trained models in distributed settings.
- **Two-Phase Framework**: Robustness via adversaria fine-tuning; Balance via game-based layer freezing.
- **Lightweight**: Outperforms SOTA with minimal overhead.

Thank you for your attention!
Questions?