A PROOFS

A.1 Proof of Lemma 1

Lemma 1. *If l_m is μ_m-strongly convex on $\boldsymbol{z}_m$ and module m is (ϵ_{m-1}, c_m)-robust in l_m, we have*

$$\max_{\|\boldsymbol{\delta}_{m-1}\|\leq\epsilon_{m-1}} \|\Delta \boldsymbol{z}_m\|_2 \leq \frac{\|\nabla_{\boldsymbol{z}_m} l_m(\boldsymbol{z}_m, y)\|_2}{\mu_m} + \sqrt{\frac{2c_m}{\mu_m} + \frac{\|\nabla_{\boldsymbol{z}_m} l_m(\boldsymbol{z}_m, y)\|_2^2}{\mu_m^2}}.$$

Proof. With the strong convexity and the (ϵ_{m-1}, c_m)-robustness of l_m, we have

$$\begin{aligned}
&(\nabla_{\boldsymbol{z}_m} l_m(\boldsymbol{z}_m, y))^T \Delta \boldsymbol{z}_m + \frac{\mu_m}{2}\|\Delta \boldsymbol{z}_m\|_2^2 \\
&\leq l_m(\boldsymbol{z}_m + \Delta \boldsymbol{z}_m, y) - l_m(\boldsymbol{z}_m, y) \leq c_m \\
\Rightarrow &\left\|\Delta \boldsymbol{z}_m + \frac{\nabla_{\boldsymbol{z}_m} l_m(\boldsymbol{z}_m, y)}{\mu_m}\right\|_2 \\
&\qquad \leq \sqrt{\frac{2c_m}{\mu_m} + \frac{\|\nabla_{\boldsymbol{z}_m} l_m(\boldsymbol{z}_m, y)\|_2^2}{\mu_m^2}} \\
\Rightarrow &\|\Delta \boldsymbol{z}_m\|_2 \leq \frac{\|\nabla_{\boldsymbol{z}_m} l_m(\boldsymbol{z}_m, y)\|_2}{\mu_m} \\
&\qquad + \sqrt{\frac{2c_m}{\mu_m} + \frac{\|\nabla_{\boldsymbol{z}_m} l_m(\boldsymbol{z}_m, y)\|_2^2}{\mu_m^2}}.
\end{aligned}$$

□

A.2 Proof of Lemma 2

Lemma 2. *If the early exit loss l_m has β_m-smoothness and (ϵ_m, c_m)-robustness on $\boldsymbol{z}_m$, the joint loss l has β'_m-smoothness and (ϵ_m, c_M)-robustness on $\boldsymbol{z}_m$, we have*

$$\begin{aligned}
&\|\nabla_{\boldsymbol{w}_m} l - \nabla_{\boldsymbol{w}_m} l_m\|_2 \\
&\leq \left\|\frac{\partial \boldsymbol{z}_m}{\partial \boldsymbol{w}_m}\right\|_2 \sqrt{2(c_m + c_M)(\beta_m + \beta'_m)}.
\end{aligned}$$

Proof. We define $h_m(\boldsymbol{z}_m) = l(\boldsymbol{z}_m) - l_m(\boldsymbol{z}_m)$, which has $(\beta_m + \beta'_m)$-smoothness and $(\epsilon_m, c_m + c_M)$-robustness on $\boldsymbol{z}_m$. Thus $\forall \boldsymbol{\delta}_m$ with $\|\boldsymbol{\delta}_m\| \leq \epsilon_m$, we have

$$\begin{aligned}
&(\nabla_{\boldsymbol{z}_m} h_m(\boldsymbol{z}_m))^T \boldsymbol{\delta}_m - \frac{\beta_m + \beta'_m}{2}\|\boldsymbol{\delta}_m\|_2^2 \\
&\leq h_m(\boldsymbol{z}_m + \boldsymbol{\delta}_m) - h_m(\boldsymbol{z}_m) \leq c_m + c_M.
\end{aligned}$$

We take the maximum of the left hand side with $\boldsymbol{\delta}_m^* = \frac{\nabla_{\boldsymbol{z}_m} h_m(\boldsymbol{z}_m)}{\beta_m + \beta'_m}$, and we can get

$$\begin{aligned}
&\frac{\|\nabla_{\boldsymbol{z}_m} h_m(\boldsymbol{z}_m)\|_2^2}{2(\beta_m + \beta'_m)} \leq c_m + c_M \\
\Rightarrow &\|\nabla_{\boldsymbol{z}_m} h_m(\boldsymbol{z}_m)\|_2 \leq \sqrt{2(c_m + c_M)(\beta_m + \beta'_m)}.
\end{aligned}$$

Table 5. Performance, memory, and storage I/O Bandwidth of the devices for training on CIFAR-10.

Device	Performance	Memory	I/O Bandwidth
GTX 1650m	3.1 TFLOPS	4 GB	16 GB/s
TX2	1.3 TFLOPS	4 GB	1.5 GB/s
KCU1500	0.2 TFLOPS	2 GB	2 GB/s
VC709	0.1 TFLOPS	2 GB	1.5 GB/s
Radeon HD 6870	2.7 TFLOPS	1 GB	16 GB/s
Quadro M2200	2.1 TFLOPS	4 GB	1.5 GB/s
A12 GPU	0.5 TFLOPS	4 GB	1.5 GB/s
Geforce 750	1.1 TFLOPS	1 GB	16 GB/s
Grid K240q	2.3 TFLOPS	1 GB	16 GB/s
Radeon RX 6300m	3.7 TFLOPS	2 GB	16 GB/s

Table 6. Performance, memory, and storage I/O Bandwidth of the devices for training on Caltech-256.

Device	Performance	Memory	I/O Bandwidth
Radeon RX 7600	21.8 TFLOPS	8 GB	16 GB/s
Radeon RX 6800	16.2 TFLOPS	16 GB	16 GB/s
Arc A770	19.7 TFLOPS	16 GB	16 GB/s
Quadro P5000	5.3 TFLOPS	16 GB	1.5 GB/s
RTX 3080m	19.0 TFLOPS	8 GB	16 GB/s
RTX 4090m	33.0 TFLOPS	16 GB	16 GB/s
A17 GPU	2.1 TFLOPS	8 GB	1.5 GB/s
GTX 1650m	3.1 TFLOPS	4 GB	16 GB/s
TX2	1.3 TFLOPS	4 GB	1.5 GB/s
P104 101	8.6 TFLOPS	4 GB	16 GB/s

With the chain rule and $\|\nabla_{\boldsymbol{w}_m} l - \nabla_{\boldsymbol{w}_m} l_m\|_2 \leq \left\|\frac{\partial \boldsymbol{z}_m}{\partial \boldsymbol{w}_m}\right\|_2 \|\nabla_{\boldsymbol{z}_m} l - \nabla_{\boldsymbol{z}_m} l_m\|_2$, we prove the lemma. □

B EXPERIMENT DETAILS

B.1 Device Details

Considering the different memory and performance requirements for training on CIFAR-10 (small images) and Caltech-256 (large images), we collect two device pools for CIFAR-10 (Table 5) and Caltech-256 (Table 6) respectively. Meanwhile, we multiply *degrading factors* to the peak memory and performance to simulate the real-time available memory and performance of each client with different co-running runtime applications, such as 4k-video playing and object detection (Tian et al., 2022). Specifically, the degrading factor for memory is uniformly sampled from $[0, 0.2]$, and the factor for performance is uniformly sampled from $[0, 1.0]$.

B.2 Baselines

We compare FedProphet with joint federated adversarial learning (jFAT) (Zizzo et al., 2020), knowledge-distillation federated adversarial training (FedDF-AT (Lin et al., 2020), FedET-AT (Cho et al., 2022)), partial-training federated adversarial training (HeteroFL-AT (Diao et al., 2020), FedDrop-AT (Wen et al., 2022), FedRolex-AT (Alam et al., 2022)), and Federated Robustness Propagation (FedRBN) (Hong et al., 2023).

Table 7. The model partition of VGG16 with $R_{\min} = 60$ MB. We show the memory requirement for training with SGD and the FLOPs of one forward propagation.

Module	Layer	Mem. Req.	FLOPs
1	Conv 1 Conv 2	55.8 MB	2.6 G
2	Conv 3 Conv 4 Conv 5	46.1 MB	4.9 G
3	Conv 6 Conv 7 Conv 8	50.4 MB	6.0 G
4	Conv 9	34.7 MB	2.4 G
5	Conv 10	33.1 MB	2.4 G
6	Conv 11 Conv 12	59.3 MB	1.2 G
7	Conv 13 Linear 1 Linear 2 Linear 3	36.1 MB	0.6 G

Table 8. The model partition of ResNet34 with $R_{\min} = 224$ MB. We show the memory requirement for training with SGD and the FLOPs of one forward propagation.

Module	Layer/Block	Mem. Req.	FLOPs
1	Conv	148.6 MB	3.9 G
2	BasicBlock 1	130.2 MB	7.5 G
3	BasicBlock 2	130.2 MB	7.5 G
4	BasicBlock 3 BasicBlock 4	197.9 MB	13.3 G
5	BasicBlock 5 BasicBlock 6 BasicBlock 7 BasicBlock 8	221.6 MB	28.1 G
6	BasicBlock 9 BasicBlock 10 BasicBlock 11 BasicBlock 12 BasicBlock 13	206.5 MB	37.1 G
7	BasicBlock 14 BasicBlock 15 BasicBlock 16 Linear	204.0 MB	20.6 G

(1) jFAT trains the whole model end-to-end, with memory swapping if a client does not have sufficient memory.

(2) In knowledge-distillation FL, each client selects the largest model that can be trained with the available memory from a group of models ({CNN3, VGG11, VGG13, VGG16} in CIFAR-10, {CNN4, ResNet10, ResNet18, ResNet34} in Caltech-256). The heterogeneous locally trained models are aggregated into the large global model by knowledge distillation with a small public dataset.

(3) In partial-training FL, each client trains a sub-model of the whole model by dropping out a certain percentage of neurons or filters in each layer. The percentage is set as $1 - R_k^{(t)}/R_{\max}$ where $R_{\max}$ is the memory requirement for training the whole model.

(4) FedRBN allows clients with insufficient memory to conduct standard training only. The robustness is transferred from the batch normalization statistics of the memory-sufficient clients who conduct adversarial training to those who conduct standard training.

B.3 Model Partition in FedProphet

According to Algorithm 1 and the minimal reserved memory in each setting, the VGG16 and ResNet34 are both partitioned into 7 modules as shown in Table 7 and Table 8.

B.4 Training Hyperparameters

Common Hyperparameters We conduct FL with $N = 100$ clients, and we randomly select $C = 10$ clients to participate in training at each communication round. To guarantee that each algorithm in Table 2 and Figure 7 can converge, the total numbers of communication rounds are set to 500 for jFAT and 1000 for other baselines. In each communication round, each selected client conducts $E = 30$ iterations of local SGD. The batch size is set to $B = 64$ on CIFAR-10 and $B = 32$ on Caltech-256, and the learning rates are $\eta_0 = 0.005$ and 0.001 for VGG16 and ResNet34 respectively. We apply a learning rate decay factor $\gamma = 0.994$ such that $\eta_t = \gamma^t \eta_0$ at communication round t. The momentum is set to be 0.9, and the weight decay is set to be 10^{-4} in all our settings.

Hyperparameters for Knowledge-distillation FL We partition around 10% of each dataset as the public dataset for knowledge distillation, namely, 5000 samples in CIFAR-10 and 2500 samples in Caltech-256. Following Cho et al. (2022), we set the iterations of distillation to be 128, with the same learning rate and batch size in the common hyperparameters.

Hyperparameters for FedProphet We use $\mu = 10^{-5}$ in Table 2, which is shown to be the optimal in Figure 8. We set $\gamma = 0.05$ and $\Delta\alpha = 0.1$ in all our experiments. We set the maximal number of communication rounds for each module to be 500, while we allow FedProphet to end the training of the current module early when the accuracy is not improved in the last 50 rounds.