Appendix

A. Ethical Considerations

Our work points to potential privacy threats that may occur when parameter-efficient fine-tuning (PEFT) is applied under the federated learning (FL) setup. Since (to the best of our knowledge) privacy concerns under PEFT based FL applications are under-explored, our observations suggest an important challenge that local data can be revealed if no additional defense mechanism is applied. Users involved in training might be oblivious to these risks. As a malicious server can deploy such attacks by merely poisoning model parameters, it is crucial to explore robust verification algorithms to examine the authenticity of the models received from the server. Furthermore, defense strategies such as differential privacy under the PEFT setting can prevent the server from observing the true local gradients with a small impact on utility. We hope that our work will motivate new research directions towards certifiable privacy, integrity, and authenticity guarantees for PEFT mechanisms.

B. Algorithm

We provide the pseudocode of our proposed attack, PEFTLeak, in Algorithm 1.

Algorithm 1: PEFTLeak

Input: Pretrained model $\mathbf{w}_F$, adapter parameters $\mathbf{w}_A$, adapter gradients $\frac{\partial \mathcal{L}_i}{\partial \mathbf{w}_A}$ of user i (victim)

Output: Recovered patches $\mathbf{x}^{(t,m)}$ for $t \in [N], m \in [M]$ of user i, where N is the total number of patches and M is the number of images in the batch

// **Server: Poisoning pretrained model, $\mathbf{w}_F$**

// *(Position encoding vectors)*

Select $\mathbf{E}_{pos}^{(n)} \sim \mathcal{N}(0, \sigma)$ for $n \in \{0, \dots, N\}$

// *(Linear embedding matrix)*

Set $\mathbf{E}$ in (6) to $0.5\mathbf{I}_D$

// *(MSA layer parameters)*

Set $\mathbf{W}_Q^h, \mathbf{W}_K^h, \mathbf{W}_V^h = \mathbf{I}_{D_h \times D_h}$ for head $h \in [L]$ ▷ Equation (9)

Set $\mathbf{b}_Q^h, \mathbf{b}_K^h, \mathbf{b}_V^h = \mathbf{0}$ for head $h \in [L]$ ▷ Equation (10)

Set $\mathbf{W}_{MSA} = \mathbf{I}_{D \times D}$ ▷ Section 4.2

// *(MLP layer parameters)*

Design weights $\mathbf{W}_{MLP,1}, \mathbf{W}_{MLP,2}$ according to (28), (29)

Design biases $\mathbf{b}_{MLP,1} = \gamma \mathbf{1}_{4D}$, $\mathbf{b}_{MLP,2} = -\gamma \mathbf{1}_D$ ▷ Section 4.5

// *(LN1 and LN2 layer parameters)*

Set weights $\mathbf{w}_{LN1}, \mathbf{w}_{LN2} = \sigma \mathbf{1}_D$ ▷ Sections 4.1, 4.4

Set biases $\mathbf{b}_{LN1}, \mathbf{b}_{LN2}$ to $\mathbf{0}_D$ ▷ Sections 4.1, 4.4

Send $\mathbf{w}_F$ to the users ▷ sent once prior to training

// **Server: Poisoning global adapter, $\mathbf{w}_A$**

Set weights in down-projection layer to $\mathbf{E}_{pos}^{(t)}$ for target position $t \in [N]$ ▷ Section 4.3

Design biases in down-projection layer according to (19)

Set weights and biases in up-projection layer to 0 ▷ Section 4.3

Send $\mathbf{w}_A$ to the users ▷ sent in each training round

// **User i: Local training**

Compute loss $\mathcal{L}_i(\mathbf{w}_F, \mathbf{w}_A)$ for batch of images

Compute gradient $\frac{\partial \mathcal{L}_i}{\mathbf{w}_A}$

Send $\frac{\partial \mathcal{L}_i}{\mathbf{w}_A}$ to the server ▷ sent in each training round

// **Server: Reconstruction from gradients**

Recover embeddings $\mathbf{y}^{(t,m)}$ for $t \in [N], m \in [M]$ ▷ Equation (24)

Recover patch $\mathbf{x}^{(t,m)}$ for $t \in [N], m \in [M]$ ▷ Equation (25)

Return $\mathbf{x}^{(t,m)}$ for $t \in [N], m \in [M]$ ▷ recovered patches

C. Additional Experiments

In this section, we provide additional experimental results for our proposed framework. Unless stated otherwise, for all the experiments below, we use a batch size 32, bottleneck dimension $r = 64$ and ViT-B/16 architecture in accordance with the experiments in Section 5.

C.1. Recovered Images for CIFAR-10

In Fig. 9, we demonstrate the recovery of a batch of 32 images from the gradient for the CIFAR-10 dataset. As we observe, 106-out-of-128 image patches, i.e., 82.8% of the patches are recovered.

C.2. Comparison with the Optimization-Based Baseline

We now describe the details of the optimization-based gradient inversion attack baseline. To the best of our knowledge, there are no successful optimization-based attack baselines under the PEFT setting. Reference [67] studied the performance of the optimization-based attack from [70] for PEFT and observed that it was not successful under the PEFT setup. Attack from [20] improves over [70] in terms of the reconstruction performance by taking the direction of the gradient into consideration. Essentially, the goal is to find a batch of images $\mathbf{X}$ that minimize the cosine distance between the true gradient and the predicted gradient,

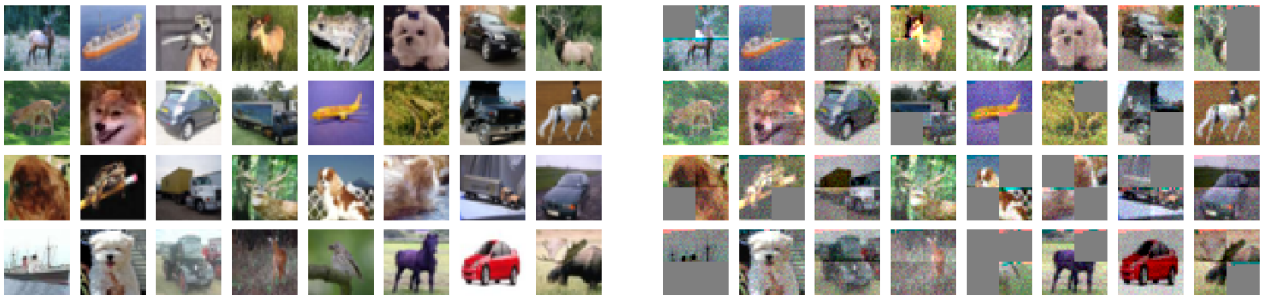
$$\mathbf{X}^* = \arg\min_{\mathbf{X}} \mathcal{F}(\mathbf{X}) \tag{30}$$

such that,

$$\mathcal{F}(\mathbf{X}) \triangleq 1 - \frac{\langle \Delta\mathbf{g}, \Delta\mathbf{g}^{pred} \rangle}{\|\Delta\mathbf{g}\| \|\Delta\mathbf{g}^{pred}\|} + TV(\mathbf{X}) \tag{31}$$

where $\Delta\mathbf{g}$ is the actual gradient received from the victim user, $\Delta\mathbf{g}^{pred}$ is the predicted gradient from training on dummy images. The total variation regularization $\text{TV}(\cdot)$ is used as a standard image prior to ensure the smoothness of the recovered image. We note that this attack considers adversaries with limited capability, who do not adopt any malicious tampering with the protocol, such as changing the model parameters or architecture.

We applied this attack to our PEFT setting and studied how well this gradient matching algorithm performs



(a) Original images (b) Recovered

Figure 9. CIFAR-10 (recovered images for a batch of 32 images).

(a) Recovered from [20] (b) Recovered (PEFTLeak)

Figure 10. Comparison with optimization-based benchmark from [20] (TinyImageNet).

(a) Recovered from [20] (b) Recovered (PEFTLeak)

Figure 11. Comparison with optimization-based benchmark from [20] (CIFAR-10).

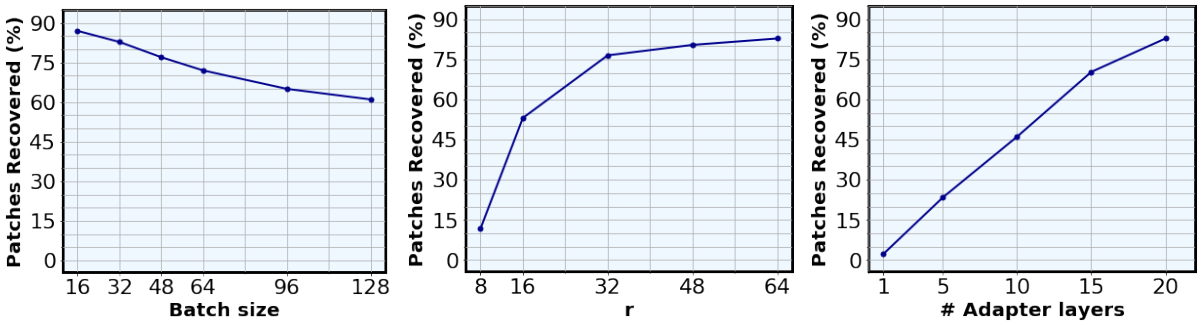
Architecture	ViT-B/16	ViT-L/16	ViT-B/32	
% **Patches recovered**	81	81	20.2 (naive)	79.6 (improved)

Table 3. Reconstruction for a batch of 32 images (TinyImageNet).

by leveraging the adapter gradients only. For this, we run the experiments for the images in Figs. 5a (in our main paper) and 9a from TinyImageNet and CIFAR-10 datasets (CIFAR-100 results were already provided in Fig. 8 in our main paper.) We demonstrate our results in Figs. 10 and 11, where we present the images reconstructed by the optimization-based attack vs. PEFTLeak. As we observe from Figs. 10 and 11, the optimization-based attack fails to reconstruct any of the images in the batch whereas PEFTLeak recovers most of the images with high fidelity.

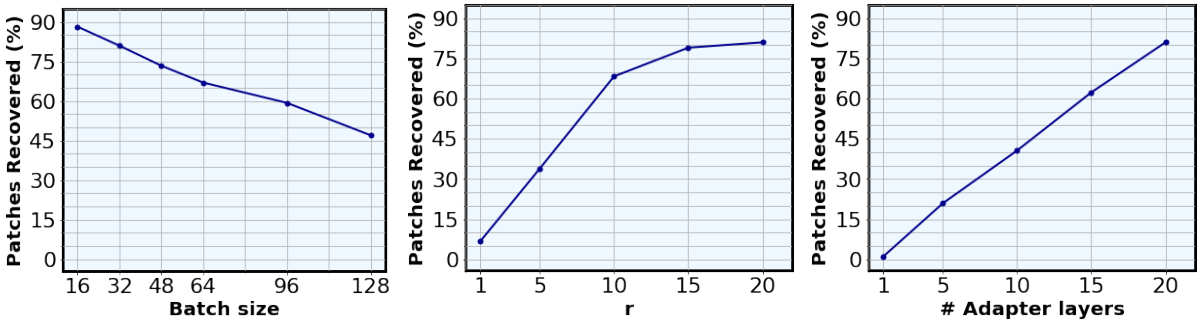
C.3. Different Model Architectures

Table 3 shows our results for ViT-L/16 and ViT-B/32 with a batch size of 32. We observed that for a fixed embedding dimension D, more encoders (ViT-L/16) can speed up our attack. ViT-L/16 (24 encoders) recovers an image in just 2 rounds, compared to 4 rounds for ViT-B/16 (12 encoders). When the number of encoders is fixed, we observed an interesting relation between D and patch size P. In ViT-B/32, each $(P, P) = (32, 32)$ patch flattens to a $P^2C = 3072$-dimensional vector (C channels). If $D \geq P^2C$, as in ViT-B/16 ($P = 16, D = 768$) and ViT-L/16 ($P = 16, D = 1024$), all pixels can be recovered. In ViT-B/32, $D < P^2C$, limiting naive recovery to $D = 768$ pixels. A simple solution is then to recover an average pixel from each $(2, 2)$ region, yielding a lower resolution recon-



(a) Varying batch size (b) Varying r (c) Varying # adapter layers

Figure 12. Percentage of patches recovered with varying batch size, bottleneck dimension, and number of adapter layers used within a single training round (CIFAR-10).



(a) Varying batch size (b) Varying r (c) Varying # adapter layers

Figure 13. Percentage of patches recovered with varying batch size, bottleneck dimension, and number of adapter layers used within a single training round (TinyImageNet).

Figure 14. Recovered images from different model architectures.

struction. Fig. 14 illustrates this for a recovered sample.

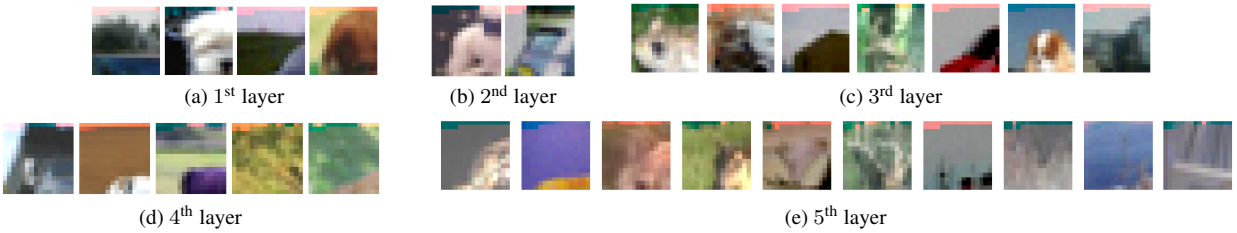
C.4. Ablation Study

Varying batch size. We next demonstrate the reconstruction performance with varying batch size, bottleneck dimension and number of adapter layers for CIFAR-10 and TinyImageNet dataset (CIFAR-100 results were provided in Section 5 in our main paper). In Figs. 12a and 13a, we observe that even for batch sizes as large as $64, 96, 128$, a notable amount of the patches are recovered.

Varying bottleneck dimension. We next report the reconstruction rate for varying r, the bottleneck dimension within each adapter layer. Higher value of r implies that more neurons are available in each adapter layer that can be leveraged for reconstruction. In Figs. 12b, 13b, we observe that as r increases, more patches are recovered.

Benefits of using multiple adapter layers. For the experiments in Figs. 4b, 5b and 9b, we have allocated 5 adapter layers for the reconstruction of patches from each position. As mentioned in Section 5, images from CIFAR-10 and CIFAR-100 datasets are divided into 4 patches. Therefore, for 4 patches, we utilize 20 adapter layers in total within a single training round. For TinyImageNet, each image is divided into 16 patches. The server aims to recover 4 patches from 20 adapter layers per training round. For this, the server sends malicious adapter parameters to recover patches from 4 target positions by leveraging the adapter gradients received from the user in each round. Hence, all the patches are recovered over 4 training rounds. In this regard, we next demonstrate the benefit of using multiple adapter layers in terms of attack success. In Figs. 12c and 13c, we report the percentage of patches recovered per training round. As we observe, more patches are recovered as more adapter layers are being utilized.

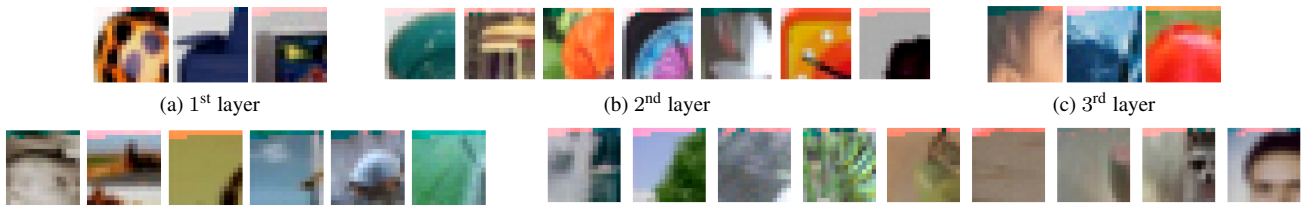
We further provide the illustration of the recovered patches in Figs. 15-20. Figs. 15, 16, and 17, demonstrate the recovery of the patches from the first position, i.e., top-left patch of the images from Figs. 9a, 4a and 5a. As de-



(a) 1st layer (b) 2nd layer (c) 3rd layer

(d) 4th layer (e) 5th layer

Figure 15. Recovered patches from the first position using multiple adapter layers (CIFAR-10).



(a) 1st layer (b) 2nd layer (c) 3rd layer

(d) 4th layer (e) 5th layer

Figure 16. Recovered patches from the first position using multiple adapter layers (CIFAR-100).

scribed in Section 4, the weight and bias parameters in the adapter layers are designed such that patches from the target position can be recovered by leveraging the adapter gradients. Patches from all other positions will be filtered out by the activation function. We observe that by utilizing multiple adapter layers, we recover most of the target patches for this position. Moreover, in Figs. 18, 19, and 20, we demonstrate the recovered patches from the same target position for $r = 8$ in comparison with $r = 64$. As we observe, more patches are retrieved from the adapter gradients when r is increased from 8 to 64.

C.5. Robustness Against Defense Mechanisms

Fig. 21 presents the attack performance against potential defense mechanisms, including noise addition [1], pruning (top-K) [4, 35] and stochastic quantization [3]. Attack performance is measured in terms of average LPIPS score [65] between recovered and ground-truth images. In Fig. 21a, we vary the standard deviation of added Gaussian noise with respect to the gradient norm.

C.6. Attack to FedAvg

We next consider the FedAvg setup, where each user performs multiple rounds of local training before sending the gradient to the server. We again leverage the activation structure from [17] (proposed for the FedAvg setting) in the down-projection layer within each adapter. At each global training round, each user performs local training for 5 epochs, and sends the local gradient to the server. We demonstrate the reconstructed images in Fig. 22b, and observe that image patches can be recovered with high fidelity.

C.7. Reconstruction on Additional Images

In Fig. 23, we further demonstrate the reconstructed images from a larger batch size. For this, we consider the images from CIFAR-100 dataset for a batch of size 64. As we observe in Fig. 23, successful reconstruction of 75% of the patches is obtained from the adapter gradients.

σ	1	2	3	5	10
Gaussian	12	30.4	52.3	77.3	85.9
Laplacian	12.5	35.1	57.8	70	92.9

Table 4. % patches recovered with different σ and distributions.

C.8. Attack Detectability

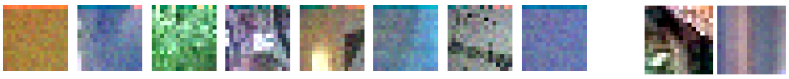
Our attack leverages the fact that users implicitly trust the server for the pretrained model and fine-tuning parameters. However, our malicious design may cause the users to question the integrity of the server. As described in Section 4.3, to recover patches from a target position, our attack sets the weight rows to be identical in the first linear layer of the adapter modules. To make this design more stealthy, the server can introduce non-malicious weight rows and biases in-between. Moreover, for position encoding, any distribution can be used if they meet the criteria outlined in (12) and Section 4.1. Table 4 shows our results with lower standard deviation σ across multiple distributions to improve stealth. Even with a σ as small as 3, our attack can recover 57.8% of the patches (batch size 32, CIFAR-100).

C.9. Reconstructed Images from the ImageNet Dataset

In Fig. 24, we present sample images from ImageNet.

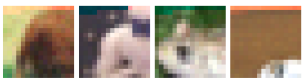
(a) 2nd layer

(b) 3rd layer

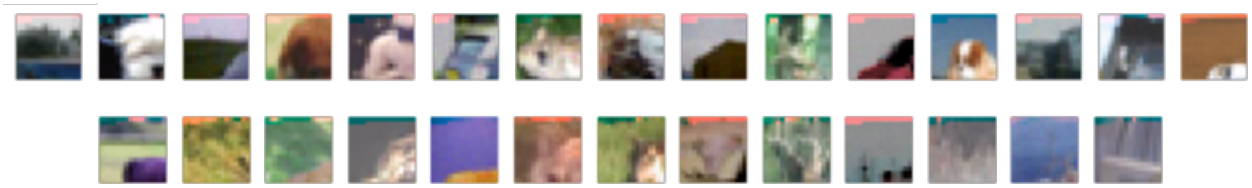


(c) 4th layer (d) 5th layer

Figure 17. Recovered patches from the first position using multiple adapter layers (TinyImageNet). None of the patches are recovered from the 1st layer gradients.



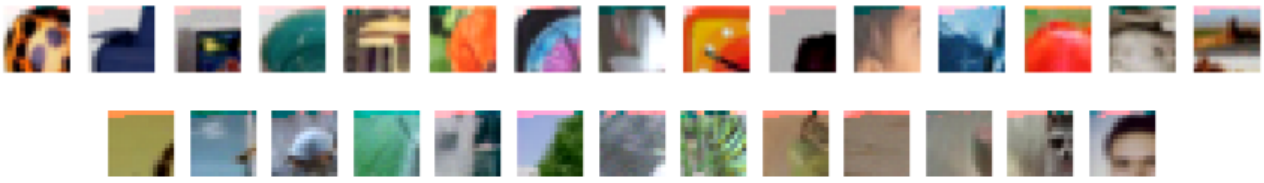
(a) $r = 8$



(b) $r = 64$

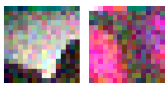
Figure 18. Impact of bottleneck dimension r on patch reconstruction (CIFAR-10).

(a) $r = 8$



(b) $r = 64$

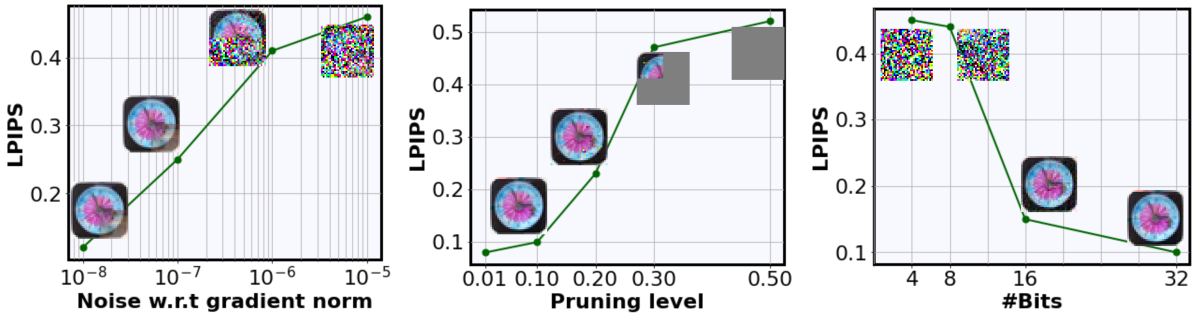
Figure 19. Impact of bottleneck dimension r on patch reconstruction (CIFAR-100).



(a) $r = 8$

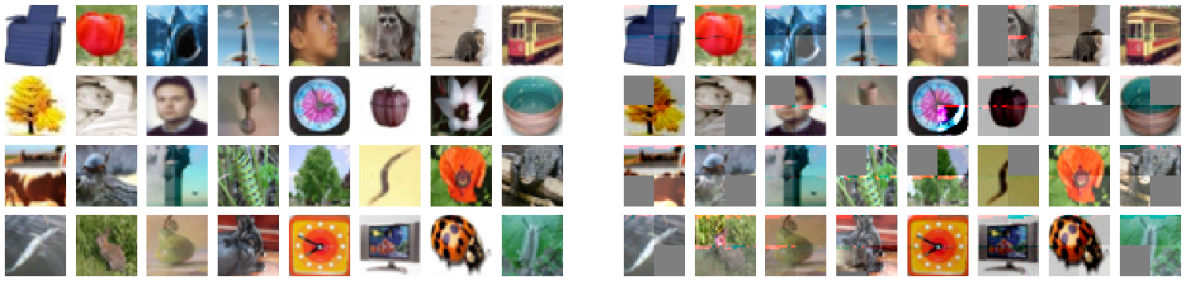
(b) $r = 64$

Figure 20. Impact of bottleneck dimension r on patch reconstruction (TinyImageNet).



(a) Noise (b) Pruning (c) Quantization

Figure 21. Performance against mitigation strategies (CIFAR-100, batch size 32). Lower LPIPS denotes better reconstruction.



(a) Original images (b) Recovered

Figure 22. Recovered images for FedAvg with 5 local training rounds (CIFAR-100).

(a) Original images (b) Recovered (PEFTLeak)

Figure 23. Recovered images for a batch of size 64 (CIFAR-100).

(a) Ground-truth (b) Recovered

Figure 24. Recovered images (ImageNet).