On the Efficiency of Privacy Attacks in Federated Learning

Nawrin Tabassum[1], Ka-Ho Chow[2], Xuyu Wang[3], Wenbin Zhang[3], Yanzhao Wu[3]*
[1] Ahsanullah University of Science and Technology
[2] University of Hong Kong
[3] Florida International University
nawrin.cse@aust.edu, kachow@cs.hku.hk, {xuywang, wenbin.zhang, yawu}@fiu.edu

Abstract

Recent studies have revealed severe privacy risks in federated learning, represented by Gradient Leakage Attacks. However, existing studies mainly aim at increasing the privacy attack success rate and overlook the high computation costs for recovering private data, making the privacy attack impractical in real applications. In this study, we examine privacy attacks from the perspective of efficiency and propose a framework for improving the Efficiency of Privacy Attacks in Federated Learning (EPAFL). We make three novel contributions. First, we systematically evaluate the computational costs for representative privacy attacks in federated learning, which exhibits a high potential to optimize efficiency. Second, we propose three early-stopping techniques to effectively reduce the computational costs of these privacy attacks. Third, we perform experiments on benchmark datasets and show that our proposed method can significantly reduce computational costs and maintain comparable attack success rates for state-of-the-art privacy attacks in federated learning. We provide the codes on GitHub at https://github.com/mlsysx/EPAFL.

1. Introduction

Federated Learning (FL) follows a decentralized approach to train Machine Learning (ML) models by transmitting gradients and avoiding sharing private user data. These methods have gained huge popularity in recent years in the computer vision domain due to their privacy-preserving capabilities for training vision models. They are especially useful in cases where training data is highly sensitive and should not be shared over the internet, e.g., healthcare [10, 25] and finance [19, 33] data. In practice, FL models [13, 20] are trained on edge devices (e.g., smartphones, IoT devices), where only the gradients are uploaded to the central server and aggregated to update the global model.

*Corresponding author.

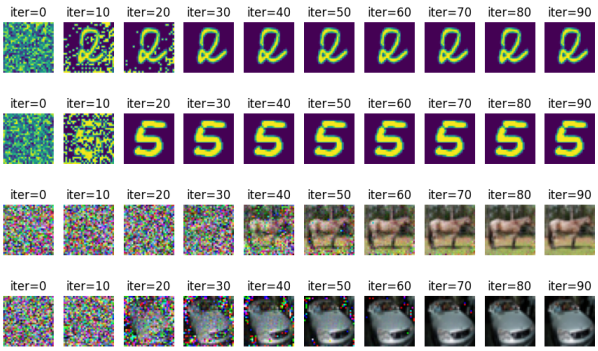

Figure 1. Visualization of Privacy Attack

The updated global model will be sent back to the client devices for the next round of training. However, several recent studies have revealed that FL is vulnerable to various security risks and privacy attacks, such as data poisoning attack [26, 32], model poisoning attack [5], model hijacking [2, 23], GAN-based attack [40], backdoor attack [1], and gradient leakage attack [43]. In particular, the gradient leakage attack has received growing attention [29, 41, 43]. Though training data are not shared directly in FL, the input data and labels can still be extracted from the shared gradients. This phenomenon is known as the gradient leakage attack, which puts the privacy guarantee of FL into question. The gradient leakage attacks initialize dummy data and calculate the dummy gradients from these data. Then the distance between actual and dummy gradients is minimized through an iterative process to update the dummy data so that the dummy data will gradually become similar to the private data. Following this approach, the gradient leakage attack was first proposed in [43] to reconstruct training data from shared gradients. Subsequently, an improved gradient attack was shown in [41] where both data and labels can be reconstructed with very high accuracy.

Most existing studies [29–31, 41, 43] on gradient leakage attacks mainly focus on achieving a higher Attack Success Rate (ASR) and/or better reconstruction quality of private

training data. Gradient leakage attacks primarily leverage a pre-defined fixed number of iterations to recover private training data, lacking the adaptability to improve the privacy attack efficiency. For example, Figure 1 visualizes the privacy attack process on two image examples from MNIST [15] (first two rows) and two image examples from CIFAR-10 [14] (bottom two rows) datasets for 100 iterations. We observe that different numbers of iterations are required to successfully recover different images. It incurs unnecessary computational costs to perform privacy attacks for the entire pre-defined number of iterations, potentially rendering these attacks ineffective under time constraints. On the other hand, efficient privacy attacks can also facilitate the quick design and evaluation of defense strategies. However, very few studies have studied the efficiency challenges of privacy attacks. Several open challenges remain to be addressed: (1) how to optimize the privacy attack efficiency without compromising the attack effectiveness and (2) how to determine the adequate number of iterations required to successfully recover private training data.

In this paper, we present a holistic framework to improve the Efficiency of Privacy Attacks in Federated Learning (EPAFL). *First*, we examine the computational costs of various privacy attacks to identify prospects for improving the efficiency of these attacks. *Second*, we provide different early stopping techniques in EPAFL to dynamically adjust the number of iterations required for reconstructing private data, which significantly reduces the data reconstruction costs. *Third*, we perform comprehensive experiments on two benchmark datasets (MNIST [15] and CIFAR-10 [14]), which demonstrates that EPAFL can effectively optimize the efficiency of privacy attacks.

2. Related Work

2.1. Federated Learning

Traditional centralized machine learning typically trains models on a single server using private data. This practice not only entails high computation requirements of the server but also poses high risks of leaking private data. With the increasing size and cost of deep learning models and the introduction of stringent privacy regulations, distributed machine learning [4, 12, 13, 16, 20, 24, 36, 37], represented by Federated Learning (FL) [13, 16, 20], emerged to meet these demands, where the computational power of multiple computing nodes (e.g., edge devices) are utilized for training a global ML model. In FL, the local models are trained on client devices using private data and only the gradients are uploaded to the server. Upon receiving these gradients, the server aggregates them and updates the global model following an aggregation algorithm [17, 18, 21, 22, 27]. The updated global model is then distributed to the client devices to continue the next round of local training. As training data is not directly shared with the server, FL preserves the confidentiality of sensitive private data. This approach has been adopted in several real-world cases to protect data privacy, such as building predictive models for medical diagnosis [3, 9], analyzing customer data for credit card fraud detection [38], and training personalized models for next-word prediction in keyboards [7].

2.2. Privacy Attacks in Federated Learning

Recent studies have investigated privacy risks in FL and showed that the privacy of training data can be easily compromised through shared gradients [29, 41, 43]. Gradient leakage attack is first proposed in [43] to recover private training data by minimizing the L_2 distance between actual and dummy gradients. An improved gradient leakage attack is introduced in [41] where labels are also reconstructed with very high accuracy. Another privacy attack is proposed in [40] where the input images are initialized using GAN models to keep dummy inputs closer to the actual inputs. The generalizability of these privacy attacks can be improved by utilizing different distance loss functions [6, 28]. For example, [6] leverages the cosine similarity to measure the gradient difference between actual and dummy data, and [28] introduces a weighted Gaussian kernel-based distance function. A few studies have made efforts to design new loss functions, such as adding regularization terms [29, 39]. In addition, a recursive gradient attack is proposed in [42], which performs better than the optimization-based attacks. Though most of the aforementioned studies use a single input to demonstrate privacy attacks, a few studies also suggest methods for the reconstruction of batch data [8, 39], which enhance the single-input attacks and show that private training data can be recovered even after utilizing a very large batch size.

These existing studies have been focused on increasing the attack success rate for improving the gradient leakage attacks. None of these studies have evaluated the attack performance in terms of the reconstruction costs. Although these privacy attack methods can maintain high reconstruction quality, we observe that they can be further optimized by reducing the computational costs.

3. Problem Statement

3.1. Federated Learning

The objective of Federated Learning (FL) is to train a model in a distributed manner without sharing the private training data, where only the gradients are transmitted to the central server [20]. A global model is built by the server from the gradients shared by the clients. At round r, the server first sends the global model F_W^r to a randomly selected subset of clients K. Each of these clients has its own local dataset $\{(X_k, Y_k)\}_{k=1}^K$. The clients calculate their local gradients

$\nabla_{W_k} L$ of the loss function L on their local dataset (X_k, Y_k), update their local models F_{W_k}, and send the gradients to the server. Finally, the server aggregates the gradients and updates the global model for the next round of FL training as Formula 1 shows.

$$F_W^{r+1} = F_W^r - \alpha \sum_{k=1}^{K} \frac{n_k}{n} \nabla_{W_k} L \tag{1}$$

where α is the global learning rate, n_k represents the number of samples in (X_k, Y_k) from client k, and n is the total number of samples from all the selected clients.

3.2. Gradient Leakage Attack

Federated Learning may not guarantee data privacy even without sharing private training data. The gradient leakage attack is an optimization-based privacy attack that can fully reconstruct private training data from shared gradients [29, 41, 43]. After receiving the gradients, the adversary (e.g., an honest-but-curious server) can initialize dummy data, and match the dummy data to the real private training data by minimizing the distance between real and dummy gradients. The adversary minimizes the following objective (Formula 2) to closely approach the actual private data (x, y) with the dummy data (x', y').

$$\underset{x',y'}{min} ||\nabla_W L(F_W(x'), y') - \nabla_W L(F_W(x), y)||^2 \tag{2}$$

During the attack, the dummy gradients are optimized through an iterative process to minimize the loss, which measures the gradient difference between the dummy and private data. Our EPAFL framework can effectively monitor the loss and adapt the actual number of iterations I for performing privacy attacks, which can effectively reduce the total execution time cost.

3.3. Threat Model

The goal of the gradient leakage attack is to steal the private training data of the clients from the gradients. We assume that the server is honest in following the FL protocols but curious to access clients' private training data. The adversary (server) receives the local gradients after local training and updates the global model. However, it may attempt to perform gradient leakage attacks on the shared gradients to reconstruct clients' private training data. It initializes random dummy data and puts the dummy data into the global model. Then the corresponding dummy gradients are calculated. The adversary optimizes the distance-based loss function between the dummy gradients and the shared gradients from private data to iteratively update dummy data to approximate the actual private data. We found that the privacy attack converges fast with only a small number of iterations, far from reaching the pre-defined total number of iterations, leaving a huge potential to optimize the privacy attack efficiency.

4. Methodology

4.1. Attack Method

The goal of the privacy attack is to reconstruct private training data sample $x \in X$ and label $y \in Y$ from the shared gradients. The gradient $\nabla_W L$ is calculated and shared after training the local model F_W on (x, y). The model parameters W are optimized using the training loss function L. We first randomly initialize the dummy input data x'. Then we predict the label y' to recover a private training sample x. We follow the method suggested in [41] for finding the true label. The sign of the gradient corresponding to the true label logit in the output layer is opposite to the sign of the gradient of other output layer logits. For a neural network with l layers, the label y' can be predicted in Formula 3.

$$y' = i \quad s.t. \quad \nabla W_l^i . \nabla W_l^j \leq 0 \, , \forall j \neq i \tag{3}$$

where ∇W_l^i is the gradient of the weight W_l on the ith logit of the output layer, corresponding to label i.

Then we calculate the dummy gradients as Formula 4 shows.

$$\nabla W' = \nabla_W L(F_W(x'), y') \tag{4}$$

Upon obtaining the dummy gradients, the distance between the actual gradient and the dummy gradient is iteratively minimized to gradually update the dummy data to approach the real private data. A distance-based loss function is utilized to facilitate this process. In this study, we use the Euclidean distance between ∇W (shared gradients on private data) and $\nabla W'$ (dummy gradients) as the distance function $Dist$ in Formula 5.

$$Dist(\nabla W', \nabla W) = ||\nabla W' - \nabla W||^2 \tag{5}$$

The following objective (Formula 6) is adopted to recover the private training data.

$$\underset{x',y'}{min} \, Dist(\nabla W', \nabla W) \tag{6}$$

Here, the dummy data can be updated by using an optimizer, such as SGD with a learning rate η, to reconstruct the private training sample x as shown in Formula 7.

$$x' = x' - \eta \frac{\partial Dist(\nabla W', \nabla W)}{\partial x'} \tag{7}$$

4.2. Overview of EPAFL

The main objective of our EPAFL framework is to enhance the efficiency of privacy attacks so that private data can be reconstructed in fewer iterations and in less execution time. Figure 1 visualizes the privacy attack results on four image samples. Here, we observe that private data can be successfully reconstructed in much fewer iterations than the pre-defined 100 iterations. We also plot the loss

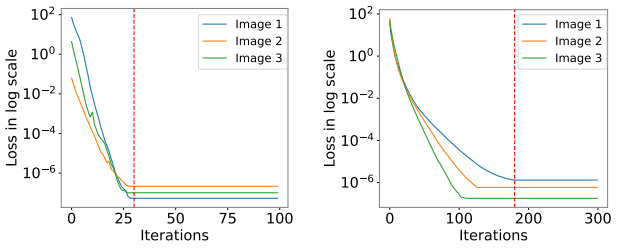

Figure 2. Loss vs. Iterations for Three MNIST (left) and Three CIFAR-10 (right) Samples

Algorithm 1 Privacy Attack with Hybrid Early Stopping

Input: $F(x, W)$: Differentiable model, ∇W: Gradients obtained from local training on private data (x, y), M: Monitored metrics distance loss, N: Maximum number of iterations, T: Threshold, P: Patience, η: Attack learning rate

Output: Reconstructed training sample (x', y')

Initialize: $x' \leftarrow \mathcal{N}(0, 1)$

Calculate the dummy label:
$y' \leftarrow j \quad s.t. \quad \nabla W_l^j . \nabla W_l^k \leq 0 \, , \forall k \neq j$
for i from 1 to N **do**
  Calculate the dummy gradient:
  $\nabla W' \leftarrow \nabla_W L(F_W(x'), y')$
  Calculate the distance between dummy gradient $\nabla W'$ and actual gradient ∇W:
  $Dist(\nabla W', \nabla W) \leftarrow ||\nabla W' - \nabla W||^2$
  Update dummy data:
  $x' \leftarrow x' - \eta \frac{\partial Dist(\nabla W', \nabla W)}{\partial x'}$
  Apply threshold-based early stopping:
  $early\_stop \leftarrow Is\text{-}Below\text{-}Threshold(M, T)$
  if $early\_stop == true$ **then**
    $I \leftarrow i$
    break
  end if
  Apply plateau-based early stopping:
  $early\_stop \leftarrow Is\text{-}Trapped\text{-}On\text{-}Plateau(M, P)$
  if $early\_stop == true$ **then**
    $I \leftarrow i$
    break
  end if
end for
return (x', y')

curves in Figure 2. We find that the loss will stop improving after a certain number of iterations as marked by the red dashed lines. This indicates that the privacy attack process can be terminated before the pre-defined number of iterations. The key challenge here is how to determine when to stop the privacy attack, which motivates us to investigate early-stopping techniques in this study. Early-stopping

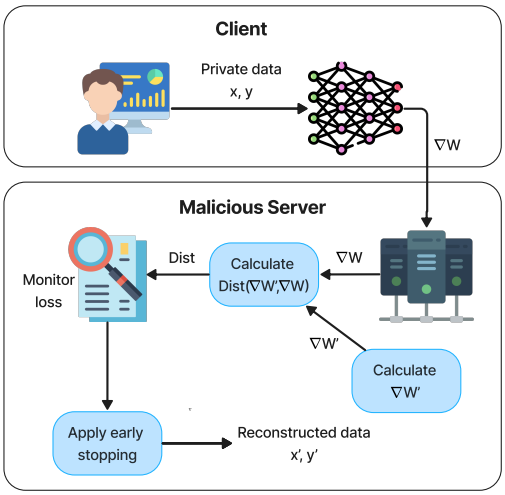

Figure 3. Architecture Overview of EPAFL

techniques hold the potential to terminate the privacy attack procedure as soon as the loss stops improving and reduce the execution time costs of privacy attacks. Figure 3 presents the architecture overview of the proposed framework EPAFL, which leverages status monitors, e.g., monitoring data reconstruction loss, and applies various early stopping techniques to significantly enhance the Efficiency of Privacy Attacks in FL.

We propose an efficient privacy attack method in Algorithm 1. Here, various early stopping strategies are employed based on the distance loss to establish effective conditions for terminating the privacy attack early without compromising the quality of reconstructed data. EPAFL monitors the loss values to determine whether to run the privacy attack for the pre-defined fixed number of iterations N or stop the attack process after the required number of iterations I. In this section, we primarily describe three different early stopping techniques that we have investigated using EPAFL: (1) threshold-based technique, (2) plateau-based technique, and (3) their combination.

4.2.1 Threshold-based Early Stopping

An intuitive method for early stopping is to terminate the privacy attack procedure when the reconstruction loss is below a threshold. The function $Is\text{-}Below\text{-}Threshold(M, T)$ takes two arguments: the monitored metric value M, such as the loss value, and a threshold value T, and checks if the monitored metric is below the predefined threshold T. If $M < T$, the privacy attack will be terminated. Otherwise, the privacy attack will continue until reaching the pre-defined number of iterations N. The threshold value T should be chosen carefully as

Algorithm 2 Is-Trapped-On-Plateau

procedure Is-Trapped-On-Plateau(M, P)
Input: M: Monitored metrics, P: Patience
Output: Early stop decision
Initialize:
$wait \leftarrow 0$
$plateau\_start \leftarrow false$
$early\_stop \leftarrow false$
$M_{best} \leftarrow \infty$

```
for i from 1 to P do
    if M_i < M_best then
        M_best ← M_i
        wait ← 0
        plateau_start ← false
    else
        wait ← wait + 1
        plateau_start ← true
    end if
end for
if wait == P and plateau_start == true then
    early_stop ← true
end if
return early_stop
end procedure
```

very small values of T may not stop the privacy attack procedure timely. On the other hand, a very large threshold (T) may reduce the execution time at the cost of poor quality of the reconstructed data.

4.2.2 Plateau-based Early Stopping

Plateau detection is another effective technique for applying early stopping and enhancing the efficiency of deep learning training [11, 34, 35]. Plateau-based early stopping monitors the loss value for multiple consecutive iterations. Given that the loss values tend to oscillate, the loss value may not be stable during the privacy attack procedure. Hence, we set a patience value P which indicates how many consecutive iterations the loss will be monitored for before stopping the privacy attack process. If there is no improvement in the loss for several consecutive iterations, the privacy attack will be terminated. Algorithm 2 shows the procedure for the plateau-based early-stopping. It takes two arguments: the monitored metric value M (e.g., the loss value) and patience P. When the loss M_i in iteration i becomes larger than or equal to the lowest loss M_{best}, the algorithm starts monitoring the loss for P successive iterations, e.g., for 10 iterations, and checks if the loss is trapped on a plateau. If the loss stagnates, indicating no improvements in data reconstruction, the privacy attack will be terminated. Otherwise, the privacy attack will continue until the next detection of loss plateau or reaching the pre-defined number of iterations. The patience value P should be chosen carefully as very small values of P may stop the privacy attack too early and impair the data reconstruction quality. On the other hand, very large values of P may result in high execution costs.

4.2.3 Hybrid Early Stopping

In our experiments, we observe that for unsuccessful privacy attacks, the loss value often stagnates at a higher value than the threshold, which may not be terminated early by only using the threshold-based method. On the other hand, when the loss is already below the pre-defined threshold with a loss very close to 0, the plateau-based method still requires P iterations before terminating the privacy attack process, leading to unnecessary execution time costs. Hence, we combine both threshold-based and plateau-based early-stopping techniques to form a hybrid method. This hybrid early-stopping method keeps track of the loss and terminates the privacy attack procedure when any early-stopping condition is met. Algorithm 1 shows this hybrid early stopping technique. Integrating two early-stopping techniques can harness the advantages of both threshold-based and plateau-based early-stopping techniques and significantly enhance privacy attack efficiency.

5. Experimental Analysis

5.1. Experimental Setup

We implement EPAFL with PyTorch and perform experiments on two benchmark image datasets, MNIST dataset [15] of image size 28×28 and CIFAR-10 dataset [14] of image size 32×32. We use the following configurations for performing the experiments. The learning rate η=1.0 for the LBFGS optimizer. We utilize an untrained LeNet [15] to conduct the experiments. To initialize the weights of the neural network, we follow a uniform distribution with values ranging from -0.5 to 0.5. The dummy data are generated using a Gaussian distribution. We randomly choose a subset of 100 samples from the dataset and perform gradient leakage attacks on those samples. The maximum number of iterations N is set as 300, implying that if the reconstruction process does not stop earlier, the privacy attack will run for 300 iterations for each sample.

5.2. Threshold-based Early Stopping

First, we perform experiments by setting different values as the threshold for the loss. In this case, the privacy attack will be stopped after the loss goes below a certain threshold T. In the first step, we take a small number of samples and perform gradient attacks on them to empirically choose the appropriate threshold values for the experiments. To study

Dataset	Threshold	ASR	MSE (Avg.)	SSIM (Avg.)	Reconstruction Time (s)	Iterations (Max)	Iterations (Min)	Iterations (Avg.)	Iterations (SD)
MNIST	0.001	0.65	0.001	0.9579	**125.26**	18	**5**	**7.4**	**2.154**
	0.0001	0.68	0.0001	0.9694	183.84	**15**	6	9.44	2.165
	0.00001	0.74	**2.955e-05**	0.977	281.85	24	8	12.72	3.411
	0.000001	**0.79**	3.576e-05	**0.9787**	371.03	34	9	15.15	4.984
CIFAR-10	0.001	0.14	0.0054	0.9354	**234.65**	**71**	**37**	**49**	**9.769**
	0.0001	0.55	0.001	0.9674	1525.31	124	54	78.22	16.466
	0.00001	**0.74**	0.0002	0.9874	3050.43	284	66	114.88	31.76
	0.000001	0.72	**4.434e-05**	**0.9956**	3763.76	300	82	180.42	81.833

Table 1. Privacy Attack Performance under Different Thresholds (Threshold-based Early Stopping)

Dataset	Patience	ASR	MSE (Avg.)	SSIM (Avg.)	Reconstruction Time (s)	Iterations (Max)	Iterations (Min)	Iterations (Avg.)	Iterations (SD)
MNIST	5	0.82	**1.833e-07**	0.9823	**313.68**	**45**	**20**	**26.44**	**4.758**
	10	0.80	9.371e-07	**0.983**	327.91	95	23	32.41	8.502
	15	**0.84**	3.526e-05	0.9824	361.38	63	28	37.35	6.062
CIFAR-10	5	0.69	**2.745e-05**	**0.997**	**2598.12**	**280**	106	**175.04**	**41.19**
	10	0.69	4.78e-05	0.9956	2974.91	300	**105**	180.01	49.35
	15	**0.71**	4.139e-05	0.996	4031.03	300	110	185.51	49.87

Table 2. Privacy Attack Performance under Different Patience Values (Plateau-based Early Stopping)

the impact of the threshold-based early stopping on Attack Success Rate (ASR), we check whether the SSIM (Structural Similarity Index Measure) is above 0.9. In this way, we ensure that the reconstruction quality is consistently maintained. A higher threshold results in a lower SSIM as the privacy attacks stop earlier before fully reconstructing the private training samples. On the other hand, a lower threshold results in higher computational costs. The privacy attack continues until the loss is very small, where the reconstruction time and number of iterations increase. Based on this observation, we choose four different threshold values to perform the experiments.

Table 1 shows the impact of varying the threshold values on the privacy attacks. We make three interesting observations. *First,* the attack success rate (ASR) is affected by the specific threshold value. We find that the ASR increases as the threshold decreases. For the MNIST dataset, the lowest ASR is 65%. However, the lowest ASR for CIFAR-10 is 14% when the threshold is set to 0.001. We attribute this observation to the high complexity and rich content of the CIFAR-10 dataset. As a large number of iterations are required to reconstruct CIFAR-10 samples, stopping too early may result in lower ASR. *Second,* the MSE (Mean Squared Error, computed as an average over all samples) is quite low for all thresholds, which indicates that the distance between actual and dummy data can be minimized even after applying threshold-based early stopping. The average SSIM is above 0.93 for all thresholds, implying that good reconstruction quality can be achieved within a small number of iterations. *Third,* applying a threshold for early stopping heavily influences the reconstruction time and number of iterations. For example, 79% of samples from the MNIST dataset can be successfully reconstructed in only 371.03 seconds and 15 iterations on average. The overall reconstruction time is very low, ranging from 125.26 to 371.03 seconds for MNIST and 234.65 to 3763.76 seconds for CIFAR-10. The results suggest that selecting a proper threshold can significantly improve privacy attack efficiency by reducing computational costs.

5.3. Plateau-based Early Stopping

In this set of experiments, we evaluate the performance of the plateau-based early stopping by detecting the loss plateau. If the loss does not improve for a few consecutive iterations (specified by patience P), we terminate the reconstruction process early to reduce the computational costs. We choose $\{5, 10, 15\}$ as patience values to investigate the impact of this early-stopping technique.

Table 2 presents the impact of different patience values on the computational costs and performance of the privacy attacks. We highlight two interesting observations below. *First,* the plateau-based early stopping performs better than the threshold-based technique (in Table 1) in terms of ASR and SSIM. For the MNIST dataset, the ASR is above 80% for all patience values, which indicates that this technique works well for reconstructing most of the samples. We observe that the ASR is significantly higher for both datasets when patience is set to 15. The SSIM is also very high

Patience	Threshold	ASR	MSE (Avg.)	SSIM (Avg.)	Reconstruction Time (s)	Iterations (Max)	Iterations (Min)	Iterations (Avg.)	Iterations (SD)
5	0.001	0.60	0.001	0.9596	126.64	18	5	7.62	2.346
	0.0001	0.67	0.0002	0.9686	185.75	36	6	9.70	3.879
	0.00001	0.77	1.755e-05	0.9738	287.27	33	7	12.43	3.859
	0.000001	0.80	3.692e-05	0.9776	369.20	35	9	15.29	5.146
10	0.001	0.62	0.0009	0.96	115.75	13	5	7.21	1.815
	0.0001	0.70	0.0001	0.9668	191.88	37	6	9.94	4.157
	0.00001	0.76	3.438e-05	0.9745	285.23	26	8	12.62	3.479
	0.000001	0.76	6.235e-05	0.9799	402.05	59	9	17.17	7.116
15	0.001	0.62	0.0008	0.9596	114.35	16	5	7.24	1.775
	0.0001	0.66	0.001	0.966	228.93	77	6	12.24	10.07
	0.00001	**0.80**	3.736e-05	0.9715	**274.73**	20	8	12.04	2.799
	0.000001	0.83	2.309e-05	0.9786	389.58	38	9	15.58	5.109

Table 3. Privacy Attack Performance under Different Thresholds and Patience Values (Hybrid Early Stopping, MNIST)

Patience	Threshold	ASR	MSE (Avg.)	SSIM (Avg.)	Reconstruction Time (s)	Iterations (Max)	Iterations (Min)	Iterations (Avg.)	Iterations (SD)
5	0.001	0.16	0.006	0.934	289.34	88	38	50.31	12.128
	0.0001	0.57	0.001	0.9626	1624.35	300	51	84.96	36.559
	0.00001	**0.70**	0.0002	0.9898	**2895.69**	230	71	120.66	40.117
	0.000001	0.71	4.706e-05	0.9955	3766.06	300	81	163.46	60.82
10	0.001	0.13	0.005	0.9386	193.34	62	37	47.54	8.044
	0.0001	0.55	0.001	0.9642	1422.47	123	53	79.22	15.161
	0.00001	0.75	0.0002	0.9873	2926.45	248	60	117.71	36.764
	0.000001	0.72	6.909e-05	0.9968	3754.77	300	90	162.28	51.827
15	0.001	0.16	0.005	0.9318	241.86	66	37	47.06	7.521
	0.0001	0.58	0.001	0.9672	1581.15	166	55	80.66	20.529
	0.00001	0.66	0.0002	0.9906	2718.70	300	67	121.56	48.399
	0.000001	0.80	5.009e-05	0.9943	4778.28	300	76	166.35	54.68

Table 4. Privacy Attack Performance under Different Thresholds and Patience Values (Hybrid Early Stopping, CIFAR-10)

for different patience values. For example, the best average SSIM of the successfully reconstructed CIFAR-10 samples is 0.997 which implies that high-quality reconstructed data can be obtained even after employing this plateau-based early stopping method. *Second,* a lower patience P can reduce the computational costs. In Table 2, the lowest data reconstruction cost is achieved by P=5 in terms of the reconstruction time and number of iterations. For MNIST, it takes only 313.68 seconds in total to reveal 82% of samples with an average of 26 iterations per sample. For CIFAR-10, it requires 175 iterations on average to successfully reconstruct 69% of the samples with a total execution time of 2598.12 seconds. The results show that the computational costs can be significantly reduced by stopping the privacy attack when the loss reaches a plateau.

5.4. Hybrid Early Stopping

We then evaluate the hybrid early stopping mechanism, which integrates both threshold-based and plateau-based early stopping techniques. We vary both the patience and threshold values to study their combined effects on privacy attack performance. We present the experimental results in Table 3 for MNIST and Table 4 for CIFAR-10 respectively. We highlight two interesting observations. *First,* the hybrid early stopping method can effectively outperform the individual threshold-based or plateau-based early stopping techniques by achieving comparable ASR at a reduced execution cost. For MNIST, the hybrid early stopping can achieve over 80% ASR with only 274.73 seconds, which is much faster than the plateau-based (313.68 seconds) early stopping technique. Similarly, for CIFAR-10, the hybrid early stopping technique can achieve over 70% ASR with only 2895.69 seconds, significantly outperforming the threshold-based (3763.76 seconds) and plateau-based (4031.03 seconds) early stopping techniques. *Second,* both the patience and threshold values have high impact on the data reconstruction quality and costs. For CIFAR-10, the lowest reconstruction time is 193.34 seconds with

Metrics	MNIST			CIFAR-10		
	DLG	iDLG	EPAFL	DLG	iDLG	EPAFL
ASR	0.66	0.79	**0.80**	0.69	0.72	**0.75**
MSE (Avg.)	0.0004	0.0003	**3.736e-05**	**4.194e-05**	4.211e-05	0.0002
SSIM (Avg.)	**0.9839**	0.9824	0.9715	**0.9961**	0.996	0.9873
Reconstruction Time (s)	1029.59	1022.07	**274.73**	4797.01	4207.33	**2926.45**

Table 5. Comparison of Our Hybrid Early Stopping with Representative Privacy Attacks

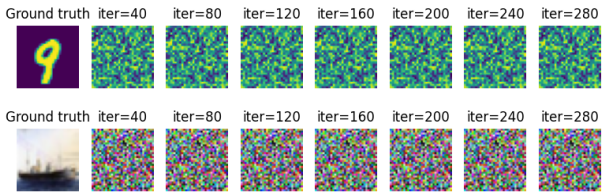

Figure 4. Failure Cases in Image Reconstruction

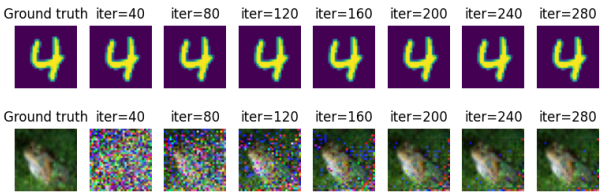

Figure 5. Successfully Reconstructed Images

threshold T=0.001 and patience P=10. However, the ASR is only 13% in this case. A higher ASR (75%) can be achieved by decreasing the threshold to 0.00001 and also outperforming the individual threshold-based or plateau-based methods in terms of reconstruction time. An interesting observation is that if we set T=0.000001 and P=15 for CIFAR-10, the ASR (80%) becomes quite higher but at the cost of a higher reconstruction time. This highlights the necessity of choosing a suitable combination of threshold and patience to maintain a higher attack success rate at lower computational costs.

5.5. Comparison to Other Privacy Attacks

We next compare our hybrid early stopping with two representative privacy attacks: DLG [43] and iDLG [41]. Table 5 presents the experimental results for comparing the three privacy attack methods. We primarily compare the data reconstruction quality metrics (ASR, MSE and SSIM) and the execution time. Three interesting observations should be highlighted. *First,* we observe that our hybrid early stopping approach can outperform other privacy attack methods in terms of ASR. Here, we configure P=15 and T=0.00001 for MNIST (see Table 3) and P=10 and T=0.00001 (see Table 4) for CIFAR-10. We attribute the enhanced ASR by EPAFL to the potential overfitting by DLG and iDLG for running the entire N iterations. *Second,* our early stopping method can achieve comparable MSE and SSIM to other privacy attack methods. In particular, our privacy attack method achieves the lowest average MSE for MNIST. *Third,* our early stopping method can significantly reduce the reconstruction time costs by over 73% for MNIST and over 30% for CIFAR-10.

5.6. Visualization

We present two different cases in privacy attacks: (1) attack failures in Figure 4 and (2) successful attacks in Figure 5. For failure cases, the dummy data consistently exhibits noise across all iterations of the privacy attack. For the successful attacks, the private data can often be effectively recovered, significantly prior to the pre-defined 300 iterations, such as within 40 iterations for the first example in Figure 5. These observations further demonstrate that stopping the privacy attack early can significantly enhance the efficiency of privacy attacks without impairing the attack effectiveness.

6. Conclusion

In this paper, we have identified a novel research problem of how to optimize the efficiency of privacy attacks in Federated Learning. We made three novel contributions. *First,* we analyzed the performance of privacy attacks represented by gradient leakage attacks and identified the huge potential to optimize their efficiency. *Second,* we proposed a holistic framework (EPAFL) to enhance the efficiency of privacy attacks by leveraging three early-stopping techniques. *Third,* we conducted experiments on two benchmark vision datasets, which shows that our approach can effectively recover private training data at a much lower execution costs and maintain comparable attack success rates. The proposed framework will provide fast evaluation of various privacy attacks and defense methods and facilitate efficient development of various defense strategies. Our ongoing studies include (1) evaluating the EPAFL framework on other benchmark datasets, (2) supporting other privacy attack algorithms in EPAFL, and (3) investigating other techniques for improving the efficiency of both privacy attacks and defense methods.

References

[1] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *International conference on artificial intelligence and statistics*, pages 2938–2948. PMLR, 2020. 1

[2] Ka-Ho Chow, Ling Liu, Wenqi Wei, Fatih Ilhan, and Yanzhao Wu. Stdlens: Model hijacking-resilient federated learning for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16343–16351, 2023. 1

[3] Badhan Chandra Das, M. Hadi Amini, and Yanzhao Wu. Privacy risks analysis and mitigation in federated learning for medical images. In *2023 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 1870–1873, 2023. 2

[4] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc'aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, et al. Large scale distributed deep networks. *Advances in neural information processing systems*, 25, 2012. 2

[5] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. Local model poisoning attacks to {Byzantine-Robust} federated learning. In *29th USENIX security symposium (USENIX Security 20)*, pages 1605–1622, 2020. 1

[6] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradients-how easy is it to break privacy in federated learning? *Advances in Neural Information Processing Systems*, 33:16937–16947, 2020. 2

[7] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018. 2

[8] Xiao Jin, Pin-Yu Chen, Chia-Yi Hsu, Chia-Mu Yu, and Tianyi Chen. Cafe: Catastrophic data leakage in vertical federated learning. *Advances in Neural Information Processing Systems*, 34:994–1006, 2021. 2

[9] Arthur Jochems, Timo M Deist, Johan Van Soest, Michael Eble, Paul Bulens, Philippe Coucke, Wim Dries, Philippe Lambin, and Andre Dekker. Distributed learning: developing a predictive model based on data from multiple hospitals without data leaving the hospital–a real life proof of concept. *Radiotherapy and Oncology*, 121(3):459–467, 2016. 2

[10] Arthur Jochems, Timo M Deist, Issam El Naqa, Marc Kessler, Chuck Mayo, Jackson Reeves, Shruti Jolly, Martha Matuszak, Randall Ten Haken, Johan van Soest, et al. Developing and validating a survival prediction model for nsclc patients through distributed learning across 3 countries. *International Journal of Radiation Oncology* Biology* Physics*, 99(2):344–352, 2017. 1

[11] Keras Developers. "reducelronplateau". https://keras.io/api/callbacks/reduce_lr_on_plateau/, 2019. [Online; accessed 9-Mar-2024]. 5

[12] Hanjoo Kim, Jaehong Park, Jaehee Jang, and Sungroh Yoon. Deepspark: Spark-based deep learning supporting asynchronous updates and caffe compatibility. *CoRR, vol. abs/1602.08191*, 2016. 2

[13] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016. 1, 2

[14] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009. 2, 5

[15] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 2, 5

[16] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine*, 37(3):50–60, 2020. 2

[17] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020. 2

[18] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019. 2

[19] Priyanka Mary Mammen. Federated learning: Opportunities and challenges. *arXiv preprint arXiv:2101.05428*, 2021. 1

[20] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017. 1, 2

[21] Krishna Pillutla, Sham M Kakade, and Zaid Harchaoui. Robust aggregation for federated learning. *IEEE Transactions on Signal Processing*, 70:1142–1154, 2022. 2

[22] Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and H Brendan McMahan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020. 2

[23] Ahmed Salem, Michael Backes, and Yang Zhang. Get a model! model hijacking attack against machine learning models. In *29th Annual Network and Distributed System Security Symposium, NDSS 2022, San Diego, California, USA, April 24-28, 2022*. The Internet Society, 2022. 1

[24] Alexander Sergeev and Mike Del Balso. Horovod: fast and easy distributed deep learning in tensorflow. *arXiv preprint arXiv:1802.05799*, 2018. 2

[25] Micah J Sheller, Brandon Edwards, G Anthony Reina, Jason Martin, Sarthak Pati, Aikaterini Kotrotsou, Mikhail Milchenko, Weilin Xu, Daniel Marcus, Rivka R Colen, et al. Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data. *Scientific reports*, 10(1):12598, 2020. 1

[26] Vale Tolpegin, Stacey Truex, Mehmet Emre Gursoy, and Ling Liu. Data poisoning attacks against federated learning systems. In *Computer Security–ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part I 25*, pages 480–501. Springer, 2020. 1

[27] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. *arXiv preprint arXiv:2002.06440*, 2020. 2

[28] Yijue Wang, Jieren Deng, Dan Guo, Chenghong Wang, Xianrui Meng, Hang Liu, Caiwen Ding, and Sanguthevar Rajasekaran. Sapag: A self-adaptive privacy attack from gradients. *arXiv preprint arXiv:2009.06228*, 2020. 2

[29] Wenqi Wei, Ling Liu, Margaret Loper, Ka-Ho Chow, Mehmet Emre Gursoy, Stacey Truex, and Yanzhao Wu. A framework for evaluating client privacy leakages in federated learning. In *Computer Security–ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part I 25*, pages 545–566. Springer, 2020. 1, 2, 3

[30] Wenqi Wei, Ka-Ho Chow, Fatih Ilhan, Yanzhao Wu, and Ling Liu. Model cloaking against gradient leakage. In *2023 IEEE International Conference on Data Mining (ICDM)*, pages 1403–1408, 2023.

[31] Wenqi Wei, Ling Liu, Jingya Zhou, Ka-Ho Chow, and Yanzhao Wu. Securing distributed sgd against gradient leakage threats. *IEEE Transactions on Parallel and Distributed Systems*, 34(7):2040–2054, 2023. 1

[32] Wenqi Wei, Ka-Ho Chow, Yanzhao Wu, and Ling Liu. Demystifying data poisoning attacks in distributed learning as a service. *IEEE Transactions on Services Computing*, 17(1): 237–250, 2024. 1

[33] Jie Wen, Zhixia Zhang, Yang Lan, Zhihua Cui, Jianghui Cai, and Wensheng Zhang. A survey on federated learning: challenges and applications. *International Journal of Machine Learning and Cybernetics*, 14(2):513–535, 2023. 1

[34] Yanzhao Wu and Ling Liu. Selecting and composing learning rate policies for deep neural networks. *ACM Trans. Intell. Syst. Technol.*, 14(2), 2023. 5

[35] Yanzhao Wu, Ling Liu, Juhyun Bae, Ka-Ho Chow, Arun Iyengar, Calton Pu, Wenqi Wei, Lei Yu, and Qi Zhang. Demystifying learning rate policies for high accuracy training of deep neural networks. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 1971–1980, 2019. 5

[36] Yanzhao Wu, Ling Liu, and Ramana Kompella. Parallel detection for efficient video analytics at the edge. In *2021 IEEE Third International Conference on Cognitive Machine Intelligence (CogMI)*, pages 01–10, 2021. 2

[37] Yanzhao Wu, Ling Liu, Calton Pu, Wenqi Cao, Semih Sahin, Wenqi Wei, and Qi Zhang. A comparative measurement study of deep learning as a service framework. *IEEE Transactions on Services Computing*, 15(1):551–566, 2022. 2

[38] Wensi Yang, Yuhang Zhang, Kejiang Ye, Li Li, and Cheng-Zhong Xu. Ffd: A federated learning based method for credit card fraud detection. In *Big Data–BigData 2019: 8th International Congress, Held as Part of the Services Conference Federation, SCF 2019, San Diego, CA, USA, June 25–30, 2019, Proceedings 8*, pages 18–32. Springer, 2019. 2

[39] Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M Alvarez, Jan Kautz, and Pavlo Molchanov. See through gradients: Image batch recovery via gradinversion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16337–16346, 2021. 2

[40] Jiale Zhang, Bing Chen, Xiang Cheng, Huynh Thi Thanh Binh, and Shui Yu. Poisongan: Generative poisoning attacks against federated learning in edge computing systems. *IEEE Internet of Things Journal*, 8(5):3310–3322, 2020. 1, 2

[41] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. idlg: Improved deep leakage from gradients. *arXiv preprint arXiv:2001.02610*, 2020. 1, 2, 3, 8

[42] Junyi Zhu and Matthew Blaschko. R-gap: Recursive gradient attack on privacy. *arXiv preprint arXiv:2010.07733*, 2020. 2

[43] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. *Advances in neural information processing systems*, 32, 2019. 1, 2, 3, 8