Leak and Learn: An Attacker's Cookbook to Train Using Leaked Data from Federated Learning

Joshua C. Zhao, Ahaan Dabholkar, Atul Sharma, Saurabh Bagchi
Purdue University
{zhao1207,adabholk,sharm438,sbagchi}@purdue.edu

Abstract

Federated learning is a decentralized learning paradigm introduced to preserve privacy of client data. Despite this, prior work has shown that an attacker at the server can still reconstruct the private training data using only the client updates. These attacks are known as data reconstruction attacks and fall into two major categories: gradient inversion (GI) and linear layer leakage attacks (LLL). However, despite demonstrating the effectiveness of these attacks in breaching privacy, prior work has not investigated the usefulness of the reconstructed data for downstream tasks. In this work, we explore data reconstruction attacks through the lens of training and improving models with leaked data. We demonstrate the effectiveness of both GI and LLL attacks in maliciously training models using the leaked data more accurately than a benign federated learning strategy. Counter-intuitively, this bump in training quality can occur despite limited reconstruction quality or a small total number of leaked images. Finally, we show the limitations of these attacks for downstream training, individually for GI attacks and for LLL attacks.

1. Introduction

With growing concerns of data privacy, federated learning (FL) [17] has gained traction as a potential privacy-preserving method for training machine learning models. Compared to centralized learning where training is done on data localized at a central server, FL takes a decentralized approach where participating clients train a model on their local data and send their updates to the server. A typical training round involves a server sending a model to the clients, the clients training the model using their local data, and finally having the clients send their updates to the server for aggregation. However, the privacy-preserving property of FL only holds if the updates cannot be used to extract sensitive information about the local training data.

Despite only sending updates, prior works have shown the ability of attackers to gain information about the private training data through membership inference attacks [3, 19, 22], property-inference attacks [15, 18], or GAN-based methods [9, 23]. Within these privacy attacks, data reconstruction has stood out as the most powerful, allowing attackers to directly recover the training data of clients [20, 24]. Within data reconstruction attacks, gradient inversion (optimization-based attacks) [6, 26, 32] and linear layer leakage [1, 5, 30] attacks stand out as the most common.

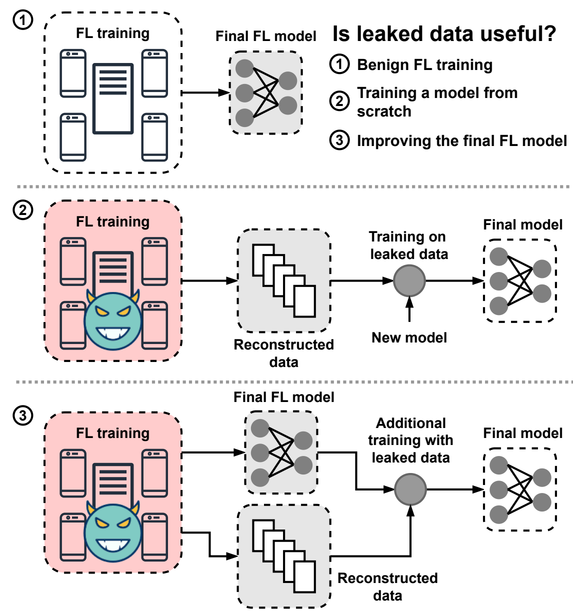


Figure 1. Training using leaked data.

While prior works have discussed the power and the limitations of these attacks in the context of an attacker simply breaching privacy [10, 29], to the best of our knowledge, none has discussed the effectiveness of using the leaked data for downstream model training. Figure 1 shows this process of training using leaked data. With the onset of deep learning, data has become a valuable commodity. This growth in AI also means that the value of data no longer lies just in breaching privacy of the raw data, but in the ability of the data to be used downstream for creating powerful models. Therefore, one measure of the success of a data reconstruction attack should be whether the leaked data is useful for machine learning training, an aspect omitted in prior work.

In order to discuss training on leaked data, we first introduce the attacks. Gradient inversion (GI) attacks typically reconstruct data through an iterative process where the distance between a gradient computed from a dummy image and the ground truth gradient is minimized by an optimizer. Label reconstruction, the process of recovering the batch labels from the client gradient, also plays a critical role in affecting the final reconstruction quality. These labels are typically leaked prior to optimization from the gradients of the final layer of the network [7, 16, 26, 28] and are matched with reconstructed images during the optimization. These attacks have shown success with individual updates of smaller batch sizes, up to 48 on ImageNet [26] or 100 on CIFAR [6]. However, this only considers breaching privacy, as a batch size of 48 on ImageNet or 100 on CIFAR result in low quality reconstructions and/or a very small number of identifiable images. Even for a modest batch size of 8 or 16 on CIFAR-10, some batch reconstructions already fail. Given the importance of data quality for training models, a sufficient condition for success is no longer if any images are identifiable, but rather the overall usefulness of the leaked data for the downstream training task. Here, even reconstructions with low similarity scores to the ground truth may still positively contribute to training.

Compared to GI, linear layer leakage (LLL) does not suffer from problems with reconstruction quality. Using modification of a fully-connected (linear, FC) layer, an attacker can directly recover a proportion of the images of a client batch [1, 5, 30]. Reconstructions are done by solving a simple linear equation and have little computation overhead. However, these attacks typically require modification of the model architecture and can add a large overhead in terms of model size [29]. While data quality is not an issue, when using the leaked data for training models, labeling the data becomes a problem. Unlike the optimization attacks, even if labels are known prior to reconstruction, the LLL process does not match them to the leaked images. As a result, after reconstruction there would be a set of leaked images and labels, where the images are not matched with their corresponding label. Furthermore, because only a proportion of images in the given batch are leaked for LLL, even with a full set of labels, some labels will not have a leaked image to go with. This can lead to a tedious process of manually labeling leaked images.

Viewing data reconstruction attacks through the lens of training models brings up many questions. How do models trained with leaked images compare to centralized or federated learning? Does the reconstruction quality impact how well models perform when trained on leaked data? How does the lack of label matching during reconstruction affect linear layer leakage? In this work we address these largely unanswered questions and highlight the problems that arise in this new setting. Our contributions go as follows:

- We demonstrate the effectiveness of training models using leaked data compared to federated learning and the centralized baselines. Using the data from linear layer leakage and optimization, models trained on leaked data can achieve 20.40% and 17.58% higher accuracy on CIFAR-10 compared to federated learning.
- We show that similarity metrics such as PSNR do not show if reconstructed images are useful for training, as even some of the worst images improve models. Training on CIFAR-10 with bad reconstructions generated by Inverting Gradients (batch size 16) with PSNR < 14 results in a 58.29% test accuracy. Removing these images and training only on images > 14 PSNR results in only a minor accuracy decrease from 76.83% to 75.48%.
- We quantify the effects and highlight the crucial limitations of current data reconstruction attacks when viewed through the lens of downstream model training. While gradient inversion can breach privacy for a batch size of 100 on CIFAR-10, this is impractical for training as even increasing the batch size from 4 to 16 results in a performance drop from 90.34% to 76.83%.

2. Related work

Gradient inversion (GI) attacks operate under an honest-but-curious threat model where a server only knows the update and model. The method involves feeding a dummy image into a model and computing the subsequent gradient. The dummy image starts out initialized as random noise, and an optimizer iteratively minimizes the distance between the computed gradient and the ground truth client gradient.

$$x^* = \arg\min_x ||\nabla L(x, y, \theta) - \nabla W||_2 \tag{1}$$

The intuition behind these attacks is that a similar computed dummy gradient $\nabla L(x, y, \theta)$ and ground truth gradient ∇W will result from similar images. Thus, as the gradients become closer, the final reconstructed image x^* should become close to the ground truth training image. The label y is assumed to be known prior to optimization. For a batch reconstruction with multiple labels, the labels are matched with the reconstructed images during optimization (as the loss and gradient are computed with these image-label pairs). Regularizers have also been introduced to assist the optimization process [6, 26]. However, these can result in reconstructed image artifacts.

Linear layer leakage (LLL) attacks are built on the observation that a fully-connected (FC) layer leaks the input to the layer through the gradients [4, 21].

$$x^i = \frac{\delta L}{\delta W^i} / \frac{\delta L}{\delta B^i} \tag{2}$$

Here, $\frac{\delta L}{\delta W^i}$ is the weight gradient and $\frac{\delta L}{\delta B^i}$ is the bias gradient of a neuron. Neuron i is activated by an image and x^i is the respective reconstructed image. However, this assumes that only a single image activated the neuron. If multiple images activate the same neuron, reconstructed image x^i becomes a combination of the images.

To mitigate this, the methods of trap weights [1] and Robbing the Fed [5] were proposed. Trap weights initializes the weights of the FC layer randomly as half positive and half negative, where the negative parameters are sampled from a slightly larger magnitude. This method has difficulty with scaling [30] compared to Robbing the Fed, which proposed a method where the FC layer weights are used to measure a property of the images (e.g., average pixel brightness). The biases of the layer were then set to fit the distribution of the dataset and a ReLU activation was used to threshold the activation. Images were reconstructed as

$$x^i = (\frac{\delta L}{\delta W^i} - \frac{\delta L}{\delta W^{i+1}})/(\frac{\delta L}{\delta B^i} - \frac{\delta L}{\delta B^{i+1}}) \tag{3}$$

where $i+1$ indicates the neuron with a bias being the next highest cutoff. This method requires basic knowledge about the input data distribution, but achieves a higher leakage rate compared to trap weights and is scalable to aggregation.

In the case of FedAvg, the sparse variant of Robbing the Fed utilizes a two-sided activation function (such as Hard tanh) and weight/bias scaling to maintain the same leakage as in FedSGD. However, the sparse variant is not scalable and leads to precision problems when attacking secure aggregation [2] or larger batches. The LOKI attack addresses the scalability problems of both methods through the introduction of a convolutional layer that splits the scaling between number of clients and batch size in secure aggregation [30]. They further introduce a convolutional scaling factor (CSF) which achieves a higher leakage rate in FedAvg without increasing the FC layer size.

While there have been many works exploring data reconstruction, prior work has not evaluated these attacks in the context of downstream tasks. The works on GI [6, 26, 32] and LLL [1, 5, 30] attacks only discuss the reconstruction quality of the methods in terms of standard image metrics such as PSNR (peak signal-to-noise ratio), SSIM (structural similarity index measure), or LPIPS (learned perceptual image patch similarity). However, these metrics only measure how similar the reconstruction are to the ground truth and only give a vague sense of how useful the data is for training. Another work discussed the limitations of GI attacks [10] measured by image similarity. In [29], LLL attacks were measured in terms of the resource overhead added by the attacks. However, these works also did not discuss the usefulness of reconstructed data post-leakage.

This work aims to bridge the gap and explore the important facet of training machine learning models on leaked data from reconstruction attacks. Viewing leaked data beyond privacy breach brings to light key weaknesses of current attacks in generating useful data. While LLL reconstructs high quality images, the other important half of data, namely having matching labels, is missing. GI attacks can breach privacy for larger batch sizes, but the quality of the leaked data would be nearly useless for training models. In the age of machine learning and artificial intelligence, the value of data has increased tremendously. Therefore, when looking at leaked data, it is important to measure success in terms of usefulness to model training.

3. Training on leaked data

Data reconstruction attacks have largely focused on the quality of the recovered image in gradient inversion (GI) or the efficiency of linear layer leakage (LLL). However, label reconstruction is important for training on leaked data. Within the pipeline of GI attacks, labels are typically recovered prior to reconstruction, as they assist in the optimization problem [10]. Recent works have shown that label restoration with large batches and duplicate labels is possible [7, 16]. More accurate labels and the added flexibility of duplicate class images directly improves prior GI attacks such as Inverting Gradients [6] and GradInversion [26].

Interestingly, one strength of LLL has been in the ability to reconstruct images without knowledge of the data labels. While the class labels can affect the distribution of the client dataset (e.g., average image brightness can be different between classes) [30], the labels themselves do not help in the reconstruction. Knowing labels helps guide GI attacks (as gradients are computed from data and label pairs), but for LLL, images activating the neurons are reconstructed using only the weight and bias gradients of the FC layer as with Equation 3. Labels are necessary to train models using leaked data, but even with a set of recovered labels, the leaked images would still remain unmatched. Unlike GI which matches during optimization, images leaked by LLL would need to be manually matched. For this work, we experiment with cases when all labels are known and matched and alternatively, when only a small portion of labels are known and matched. Nonetheless, the label leakage and matching issue stands as an open problem for LLL attacks.

Identifying which images are reconstructed correctly is also a challenging problem for LLL. For FedSGD, bins without activation can be identified, as after the subtraction process the reconstruction will be zero. However, if multiple images activate the same bin, the reconstruction is non-zero. By checking for non-zero values, we cannot directly identify whether the reconstruction is a single image or a combination of multiple. Manually filtering this can become a tedious process in practice. This is somewhat mitigated in the FedAvg case by LOKI. Across separate FedAvg local mini-batches, multiple images will *not* activate the same neuron, resulting in much fewew overlapping reconstructions. Therefore the only images that can cause multiple activations are within the same mini-batch [30]. In this work, we manually filter out multiple image reconstructions in order to test the quality of the data. However, especially for LLL in FedSGD, the problem of how to automate

MNIST			
Batch size	Model	Time per batch (s)	Time for entire dataset (days)
8	DNN	54.79	4.76
16		62.34	2.71
32		64.32	1.40
CIFAR-10			
Batch size	Model	Time per batch (s)	Time for entire dataset (days)
4	ResNet-18	422.81	61.17
8		437.54	31.65
16		435.33	15.75

Table 1. Computation time for Inverting Gradients running on a NVIDIA A100 80GB GPU. Small batches have more total batches and take more time for the whole dataset. Computation time is significantly higher for CIFAR-10 given the more complex model.

removal or how to use these multiple image reconstructions for training still remains an important item of future work.

4. Experiments

The goal of our work is to discuss the effectiveness of leaked data when used for the downstream task of image classification. We use centralized training on the entire benign dataset and federated learning (with only updates and no data shared by the clients) as the baselines for comparison. Typically, these serve as the upper bound and the lower bound respectively for the model accuracy. For federated learning, we use two settings of 10 and 50 clients. We use a non-IID bias of 0.5 and FedAvg with 3 local iterations. Additional training results for the IID setting and FedSGD are included in the supplementary.

We use Inverting Gradients [6] as the gradient inversion (GI) attack. Another candidate is GradInversion, which is a SOTA GI attack (especially for ImageNet-sized images). However, we discarded this as the author code for this attack is not available and the third-party online implementations did not achieve comparable results to the original paper. For linear layer leakage (LLL), we use the LOKI attack [30]. We use LOKI as it achieves SOTA FedAvg leakage rate and equivalent SOTA leakage rate in FedSGD compared to Robbing the Fed. Model performance for other LLL methods [1, 5] can be found in the supplementary.

We use MNIST [13], CIFAR-10 [11], and the Tiny ImageNet [12] datasets. For the MNIST dataset, we train using a simple DNN with 2 convolutional layers and 2 FC layers. For CIFAR-10 and Tiny Imagenet, we use a ResNet-18 [8]. We evaluate on the validation set for Tiny ImageNet because the test set is unlabeled. The plots only show a single run for each setting, but the (numerical) testing performance is reported as an average over 5 runs. We run all experiments on NVIDIA A100 80GB GPUs.

4.1. Gradient inversion computation time

Compared to LLL attacks, the computational overhead of GI attacks when reconstructing data, while done offline, is still non-negligible. In order to fully evaluate the leaked data, we ran the Inverting Gradients atack on the entire dataset of MNIST with batch sizes of 8, 16, and 32. We

CIFAR-10			MNIST		
Batch size	PSNR	SSIM	Batch size	PSNR	SSIM
4	27.88	0.9067	8	23.16	0.6996
8	20.77	0.7215	16	19.15	0.6506
16	15.94	0.3978	32	15.78	0.4537

(a) Gradient Inversion PSNR/SSIM

	CIFAR-10		MNIST		Tiny ImageNet	
FC factor	Leaked images	Percent	Leaked images	Percent	Leaked images	Percent
8	43788	87.58	52548	87.58	85804	85.80
4	39464	78.93	45966	76.61	75149	75.15
2	29882	59.76	35795	59.66	58012	58.01
1	18242	36.48	21815	36.36	35393	35.39

(b) Linear layer attack leakage rate

Table 2. (a) Average PSNR↑ / SSIM↑ scores of gradient inversion (Inverting Gradients) reconstructions for various batch sizes and (b) number of leaked images/leakage rate of linear layer leakage attack (LOKI) for several FC layer sizes.

also ran over CIFAR-10 with batch sizes of 4, 8, and 16. We run 10,000 iterations of optimization for each batch. We evaluate the Inverting Gradients attack in the best case scenario with an untrained model where *all* labels are correctly recovered prior to optimization.

Table 1 shows the amount of time taken by Inverting Gradients run on a NVIDIA A100 80GB GPU for MNIST and CIFAR-10 on a DNN and ResNet-18 respectively. There is some variance in the average amount of time based on batch size, but when attacking the entire dataset, smaller batch sizes always take longer time as the number of total batches is much larger. CIFAR-10 takes significantly longer to reconstruct each batch given the more complex model of ResNet-18 compared to a DNN. For a batch size of 4, CIFAR-10 takes 422.81 seconds per batch. Iterating across the entire dataset on a single GPU takes 61.17 days.

4.2. Leakage statistics

For GI we use Inverting Gradients [6], and for LLL we use LOKI [30]. For Inverting Grads, we use a batch size of 4, 8, and 16 on CIFAR-10 and 8, 16 and 32 on MNIST. For LOKI, we choose the FC layer size based on the batch size. We vary this with factors of 1, 2, 4 and 8 on CIFAR-10, MNIST, and Tiny ImageNet. We use a client batch size of 64 for all datasets, so an FC size factor of 2 would mean $64 \cdot 2 = 128$ units in the FC layer.

Table 2a gives the average PSNR and SSIM metrics (where a higher score indicates greater similarity with the ground truth) for the reconstructions across the entire dataset of CIFAR-10 and MNIST from Inverting Gradients. Increasing the batch size results in a lower PSNR and SSIM score for both CIFAR-10 and MNIST.

Table 2b reports the number of leaked images for LOKI. For an FC factor of 4, LOKI leaks 78.93%, 76.61%, and 75.15% of images on the CIFAR-10, MNIST, and Tiny ImageNet datasets. Increasing the FC layer size increases the model size overhead, but results in a higher leakage rate as images are more likely to activate different neurons. Decreasing the size creates a smaller model size overhead but results in lower leakage rate.

4.3. Training on leaked data from scratch

We start with a discussion on the models that can be trained from the leaked data of both GI and LLL attacks. The attacks are applied under the FedSGD setting. We also start with assuming that *all* labels are known prior to training — this is a best case assumption for the downstream training. For CIFAR-10, we train the centralized models using SGD with momentum 0.9 and weight decay 0.0001. We use a batch size of 128 and train for 200 epochs starting with a initial learning rate of 0.1 and a scheduler that decreases on plateau. For MNIST, we use SGD without momentum or weight decay and train for 40 epochs with a LR of 0.01. For Tiny ImageNet we use Adam with learning rate 0.001 and train for 50 epochs with a batch size of 1024. For federated learning, we use 400 training rounds for CIFAR-10 and MNIST and 500 rounds for Tiny ImageNet. Client batch size of 32, 128, and 256 are used for MNIST, CIFAR-10, and Tiny ImageNet. The test accuracy is downsampled to fit on the same axis with centralized training.

Table 3a and Figure 2a show the results for CIFAR-10 for centralized training, federated learning (FedAvg), and training with leaked data. Centralized training achieves a 94.38% accuracy and federated learning achieves a 72.76% and 68.71% peak accuracy with 10 and 50 clients respectively. Centralized and FL indeed provide the upper bound and the lower bound of the accuracy here. Across all attack settings, GI and LLL achieve higher model accuracy compared to federated learning. The reconstruction quality of GI is adversely affected by larger batch sizes and in turn also impacts the final model performance. With a batch size of 4, 8, and 16, models trained on the leaked data from GI achieves 90.34%, 86.16%, and 76.83% accuracy. With even larger batch sizes, it is likely that the model performance will drop below federated learning. The performance of models using LLL data is more stable. Between an FC layer size of 512 (factor 8) and 64 (factor 1), the accuracy only drops from 93.16% to 88.86%. This is spite of a large drop in the number of leaked images from 43788 (87.58%) to 18242 (36.48%) as shown in Table 2b. This is a subtle and consequential result — the downstream training task can still be accomplished despite a big drop in the amount of data leakage. All prior work had stopped at the stage of evaluating the proportion of leakage and therefore had missed out on this insight.

Table 3b and Figure 2b show the results for MNIST. LLL performs better than the FedAvg baseline in all cases. With an FC size factor of 8, LOKI achieves a 98.82% test accuracy (nearly the same regardless of the FC size factor), only 0.07% lower than centralized at 98.89%. GI performs slightly *worse* than federated learning, achieving 95.96%, 95.50%, and 94.29% at batch sizes 8, 16, and 32 compared to 96.17% and 96.18% with 10 and 50 clients in FedAvg. We believe this is because the noise in the reconstructions hampers model performance with GI. This affects MNIST more severely than for CIFAR because the MNIST images are more sparse and sensitive to noise.

The top-1 validation accuracy for Tiny ImageNet are given in Table 3c and Figure 2c. The total number of leaked images hurts LLL more here than in the other two datasets. Between an FC layer size of 512 (factor 8) and 64 (factor 1), the validation accuracy drops from 46.70% to 35.20%, an 11.5% drop in performance. Federated learning achieves 37.00% and 35.06% accuracy with 10 and 50 clients respectively. FL with 10 clients has slightly better accuracy than LLL with an FC size factor of 1. However, any LLL setting with a larger FC layer achieves higher accuracy than both settings in federated learning.

4.4. FedAvg with leaked images from LOKI

We explore the impact of leaking images using LLL in FedAvg when training models. We use 8 local iterations of mini-batch size 8, a local dataset size of 64, and a learning rate of 1e-4. Following the LOKI attack [30], CSF= 500 is used to achieve a higher leakage rate and reconstruction quality. Similar to before, we apply the attack across batches covering the entire CIFAR-10 training dataset.

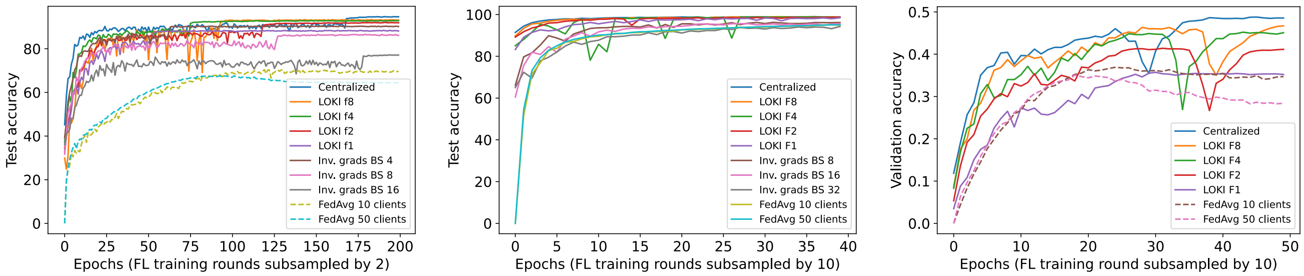
Table 4 shows the leakage rate of LOKI in FedAvg compared to FedSGD. Using $CSF = 500$, the leakage rate of LOKI in FedAvg is substantially higher than the leakage rates in the FedSGD setting. Using an FC layer of half the size, the leakage rate in FedAvg is comparable to that of FedSGD. For example, LOKI FedAvg factor 1 achieves a leakage rate of 58.39% compared to LOKI FedSGD factor 2 which achieves a leakage rate of 59.76%.

Table 5 shows the testing accuracy of LOKI FedAvg for the different FC layer sizes. The test accuracies are very comparable to the FedSGD settings. However, with a smaller FC layer size factor, the effect of having a larger total number of leaked images becomes visible. With FC factor 1, LOKI FedAvg achieves 91.11% accuracy while LOKI FedSGD achieves 88.86%.

4.5. Training with semi-supervised learning

In order to train machine learning models in a supervised fashion, images leaked through LLL need to be labeled. Previously, we assumed that all labels were known and matched to the images prior to training, but this is impractical. Within a LLL batch reconstruction, the order of images is not preserved. Even if labels were recovered like for GI attacks, this would result in a set of unmatched ground truth labels and images. Manual hand labeling is tedious and this inability to match labels is a weakness of LLL attacks *only when it comes to downstream tasks.*

In this experiment, we assume that only a small portion of images are labeled. This setting is similar to semi-supervised learning (SSL) [25, 31] and we use CoMatch [14], a semi-supervised algorithm, to train the mod-



(a) CIFAR-10 (b) MNIST (c) Tiny ImageNet

Figure 2. Training models on (a) CIFAR-10, (b) MNIST, and (c) Tiny ImageNet with leaked data compared to centralized and federated learning training. Both linear layer leakage and gradient inversion achieve higher accuracy than the federated learning (FedAvg) baseline for CIFAR-10 in all cases. For MNIST, LLL (LOKI) nearly reaches centralized accuracy while GI performs slightly worse than FL. Top-1 validation accuracy used when training models on Tiny ImageNet. Here, LLL performs better with a FC layer size factor of 2 or higher.

	Num. clients / Batch size / FC factor	Test accuracy
Centralized	-	94.38
FedAvg	10 50	72.76 68.71
Gradient inversion (Inverting grads [6])	4 8 16	90.34 86.16 76.83
Linear layer leakage (LOKI [30])	**8** 4 2 1	**93.16** 92.94 91.90 88.86

(a) CIFAR-10

	Num. clients / Batch size / FC factor	Test accuracy
Centralized	-	98.89
FedAvg	10 50	96.17 96.18
Gradient inversion (Inverting grads [6])	8 16 32	95.96 95.50 94.29
Linear layer leakage (LOKI [30])	**8** 4 2 1	**98.82** 98.70 98.72 98.46

(b) MNIST

	Num. clients / FC factor	Validation accuracy
Centralized	-	48.55
FedAvg	10 50	37.00 35.06
Linear layer leakage (LOKI [30])	**8** 4 2 1	**46.70** 45.09 41.15 35.20

(c) Tiny ImageNet

Table 3. (a) CIFAR-10 and (b) MNIST test accuracy, (c) Tiny ImageNet top-1 validation accuracy trained from scratch with leaked data. CIFAR-10 and Tiny ImageNet are trained with a ResNet-18 and MNIST is trained with a DNN. For the two attacks (gradient inversion and linear layer leakage), training happens with leaked data. Second column indicates the number of clients in federated learning (FedAvg), the batch size for the gradient inversion attack, and the fully-connected layer size factor for linear layer leakage. Best accuracy is used for FedAvg and final accuracy for all other settings.

FC size factor	LOKI FedSGD	LOKI FedAvg
8	87.58 (43788)	**92.19 (46097)**
4	78.93 (39464)	**85.98 (42989)**
2	59.76 (29882)	**75.29 (37645)**
1	36.48 (18242)	**58.39 (29196)**

Table 4. Leakage rate % (leaked images) of LOKI in FedSGD and FedAvg on CIFAR-10 for several FC layer sizes. FedAvg with half the FC layer size has comparable leakage rate to FedSGD.

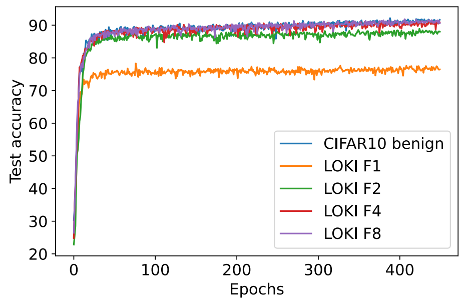
	FC factor	FedSGD accuracy	FedAvg accuracy
Linear layer leakage (LOKI [30])	8 4 2 1	93.16 **92.94** 91.90 88.86	**93.31** 92.88 **92.35** **91.11**

Table 5. CIFAR-10 test accuracy with LOKI FedAvg.

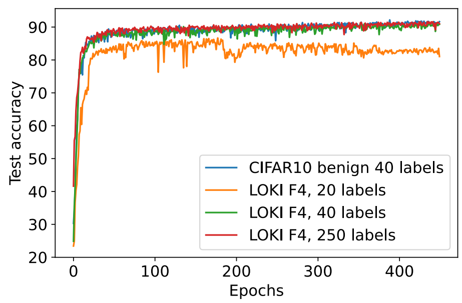
els. We use a WideResNet-28-2 [27] and train for 450 epochs on CIFAR-10. Figure 3a fixes the number of known labels to be 40 and shows the test accuracy curve with different fully connected layer sizes. We note that when there is only a small proportion of leaked data, this leads to a pronounced decrease in performance in SSL. LOKI with an FC factor of 1 (with 36.48% leakage rate) illustrates this, as it only achieves 78.31% max accuracy compared to the CIFAR-10 40 label baseline of 92.18%. With a factor of 2, the accuracy improves and jumps to 88.91% as the leakage rate also substantially increases to 59.76%. With factors 4 and 8 (78.93% and 87.58% leakage rate), the accuracy is very close to the baseline at 91.18% and 91.84% respectively. Figure 3b shows the test accuracy plot of LOKI size factor 4 with the number of known labels being 20, 40, and 250. With 20 and 250 labels, LOKI factor 4 achieves 86.69% and 91.65% peak accuracy. We note that there is a greater variability in accuracy with a smaller number of labels, a trend consistent with other SSL methods [14, 25, 31].

4.6. Starting training from federated models

While the previous experiments worked with leaked data from all batches in the dataset, such large amounts of leaked data may not always be available. For example, LLL attacks may affect model performance due to manipulation of the model parameters and/or architecture. In later rounds when model performance is high, this may be easier to detect. Similarly, GI attacks such as Inverting Gradients [6] have better reconstruction quality with an untrained network. In both cases, it is possible that the attack is only applied during a few initial rounds of training and only a small portion of data is leaked. Training from scratch with a small portion of leaked data does not achieve good model performance alone, so instead, we can initialize from the fully trained FL



(a) Varying leakage rate



(b) Varying number of labels

Figure 3. Semi-supervised learning using CoMatch on CIFAR-10 with a WideResNet with (a) a varying FC size and leakage rate (LR) for LOKI and the known labels fixed at 40 and (b) a fixed number of leaked images and 20, 40, and 250 known labels.

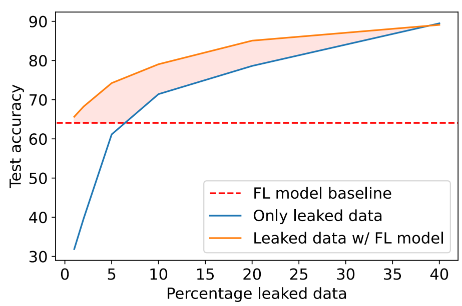


Figure 4. FL model trained with 50 clients as start point for training with leaked LLL data. Highlighted area indicates improvement above the FL model and training on the leaked data alone.

model and then train it centrally using the leaked data.

We use smaller portions of the leaked data from LOKI FC size factor of 8 for this experiment. With only 5% (2500) of leaked images in CIFAR-10, a model trained from scratch can only achieve 61.15% test accuracy. With 2% (1000 images) of data, the model performance drops to 39.63%. Both cases are below an FL model with 50 clients that achieves a 64.08% final test accuracy. However, by using the FL model as the initial starting point, even a small amount of leaked data allows for an increase in performance. We take the model that has been trained using FL and then additionally train that model centrally. Using only 2% and 5% of the leaked data, the models are able to achieve 68.25% and 74.25% final testing accuracy, a 28.62% and 13.10% improvement from starting from scratch. Figure 4 shows the improvement in test accuracy achieved by using the trained FL model as initialization instead of starting from scratch. The improvement is especially noticeable with small percentages of leaked data.

PSNR	% imgs kept	Test accuracy
> 20	15.88	70.04
> 18	27.59	71.52
> 16	40.78	73.73
> 14	60.22	75.48
> 12	84.42	76.16

(a) PSNR above threshold

PSNR	% imgs kept	Test accuracy
< 20	84.12	73.03
< 18	72.41	67.09
< 16	59.22	63.71
< 14	39.78	58.29
< 12	15.58	45.05

(b) PSNR below threshold

Table 6. Training models using CIFAR-10 leaked data from inverting gradients batch size 16. Only the reconstructions with a PSNR (a) above and (b) below the threshold are used in training.

4.7. Quality of data reconstruction

Quality of reconstruction is an important concern for GI attacks. As the batch size continues to increase, the reconstruction quality decreases. Table 2a also shows this relationship. From the previous experiments, we also see a negative relationship between the increasing batch size and the usefulness of the leaked data in downstream model training.

Here, we explore whether poor quality reconstructions are still useful for training models. Using the CIFAR-10 leaked dataset created using Inverting Gradients on a batch size of 16, we first sort the reconstructions by their PSNR value. Our first set of experiments removes the worst images from training. Table 6a shows the percent of images above the PSNR threshold and the final test accuracy when trained on that data. Compared to the baseline accuracy of 76.83% achieved by using the entire leaked dataset, even removing the worst images with a PSNR < 12 from the training set results in a small but non-zero performance loss. Then we take a different look at this question by now training on the worst images. Table 6b shows the test accuracy of models trained on all images *below* a PSNR threshold. Even when training on the lowest quality images with a PSNR < 12, the model achieves a 45.05% final testing accuracy, significantly worse than the baseline, but much above zero.

From the previous results, we see that even leaked data with a low reconstruction quality is still useful to the model training process. This message is a new insight — all prior works when calculating the leak rate do not count poorly reconstructed images. We additionally include results using SSIM in the supplementary section. Visual examples of poor quality reconstructions used to train the models are also included in the supplement.

4.8. Observations on reconstruction quality trends

We additionally note a few trends manually observed when going through the GI batch reconstructions of CIFAR-10.

While some reconstructions have very low quality, only images of the same label will be swapped within the batch during label matching. This is a desirable property for train-

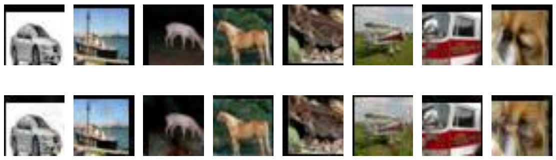


Figure 5. Reconstructions on CIFAR-10 from Inverting Gradients batch size 8. Ground truth images are on top and reconstructions are on the bottom. All labels in the batch are different and reconstructed images are high quality.

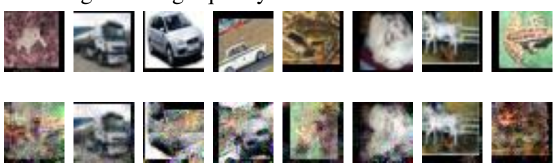


Figure 6. Reconstructions on CIFAR-10 from Inverting Gradients batch size 8. Duplicate labels exist between images 0, 4, 7 and images 2, 3. Reconstruction 3 contains parts of the content from both images 2 and 3. Reconstructions 0, 4, and 7 appear almost as noise. However, the other images still have reasonable quality.

ing on leaked data. As long as the set of labels is leaked correctly, the optimization process will match images to a correct label. So even though the image quality is poor (and can be entirely unrecognizable), since the label is correctly matched, such an image still does help in the training (this buttresses the message from Tables 6a and 6b). Naturally, as the PSNR goes down, the image helps less and less.

We also note that when all labels are different within a batch, the reconstruction quality of the entire batch of images is almost always good. An example on CIFAR-10 is shown in Figure 5. Usually when the reconstructions are poor, it occurs between groups of images in the same batch that belong to the same class. Poor reconstructions can appear as almost entirely noise, but occasionally images can have visible pieces of each image from the same class in the batch stitched together. The example batch in Figure 6 depicts these properties of poorly reconstructed images. As a side note, we observe that a batch may have a mix of well reconstructed and poorly reconstructed images. We also note that the reverse mapping of the same label images to poor reconstructions is not true — having duplicate labels does not always imply that the reconstructions will be bad.

5. Discussion

As a main goal of our work, we have explored the effect of leaked data through today's leading data reconstruction attacks on downstream model training. Our main message is that it is important to also consider the use of data beyond reconstruction quality and image similarity. It is important to consider how far these leaked samples help in a downstream training task. From the experiments, we have seen that current reconstruction attacks are powerful, but still lack in several critical areas. For gradient inversion attacks, the reconstruction quality does pose an issue for training models. Another impediment to practical use of this attack type is the time and computational cost.

Linear layer leakage attacks do not suffer from reconstruction quality issues and are much lighter weight computationally, but still suffer from other challenges. While LOKI has increased the efficiency of the FC layer leakage in FedAvg, there is still room for improvement (especially in FedSGD). As shown in [29], the model size overhead added by these attacks can be very large, especially if secure aggregation is used. Furthermore, the lack of labels poses a large challenge in training models on the data. With a small dataset like CIFAR-10, hand labeling 40 images is not too challenging. However, larger datasets will require more total labels. For example, SimMatch with 1% of labels on ImageNet can achieve 67.2% top-1 accuracy [31]. However, even 1% of the total images in ImageNet-1k is 12,812 images. Even if the images were leaked, hand labeling would be extremely hard. Furthermore, we see from Figure 3a that having less total data and fewer labels results in an even steeper decrease in performance.

While not discussed previously, there are many other aspects that affect data reconstruction attacks when considering training models. For example, how does the non-IID aspect of client data affect reconstruction? Other relevant settings of federated learning such as asynchronous federated learning, client selection, and differential privacy can also affect the final leaked dataset. Centralized training also allows for more complex and larger models without the communication or computational restrictions of clients in federated learning. These topics can be explored in future work.

6. Conclusion

We examine data reconstruction attacks through the lens of training models on the leaked data. While highlighting how the weaknesses of gradient inversion in terms of reconstruction quality affect downstream model training, we also show that even poorly reconstructed images are useful for training. We also discuss how the label matching problem for linear layer leakage can be mitigated through semi-supervised learning. Under ideal conditions, we demonstrate that leaked data from both gradient inversion and linear layer leakage attacks are able to train powerful models comparable to even a centralized baseline. On CIFAR-10, gradient inversion and linear layer leakage attacks achieve 90.34% and 93.16% testing accuracy respectively, 17.58% and 20.40% higher than federated learning and only 4.04% and 1.22% lower compared to centralized training.

Acknowledgements. This material is based in part upon work supported by Adobe Research, the Army Research Office under Contract number W911NF-2020-221, and the National Science Foundation under Grant Number CCF-1919197. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

References

[1] Franziska Boenisch, Adam Dziedzic, Roei Schuster, Ali Shahin Shamsabadi, Ilia Shumailov, and Nicolas Papernot. When the curious abandon honesty: Federated learning is not private. *8th IEEE European Symposium on Security and Privacy (IEEE Euro S&P)*, 2023. 1, 2, 3, 4

[2] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, 2017. 3

[3] Christopher A Choquette-Choo, Florian Tramer, Nicholas Carlini, and Nicolas Papernot. Label-only membership inference attacks. In *International conference on machine learning*, pages 1964–1974. PMLR, 2021. 1

[4] Lixin Fan, Kam Woh Ng, Ce Ju, Tianyu Zhang, Chang Liu, Chee Seng Chan, and Qiang Yang. Rethinking privacy preserving deep learning: How to evaluate and thwart privacy attacks. In *Federated Learning*, pages 32–50. Springer, 2020. 2

[5] Liam H Fowl, Jonas Geiping, Wojciech Czaja, Micah Goldblum, and Tom Goldstein. Robbing the fed: Directly obtaining private data in federated learning with modified models. In *International Conference on Learning Representations*, 2022. 1, 2, 3, 4

[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. 1, 2, 3, 4, 6

[7] Jiahui Geng, Yongli Mou, Feifei Li, Qing Li, Oya Beyan, Stefan Decker, and Chunming Rong. Towards general deep leakage in federated learning. *arXiv preprint arXiv:2110.09074*, 2021. 2, 3

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4

[9] Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. Deep models under the gan: information leakage from collaborative deep learning. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 603–618, 2017. 1

[10] Yangsibo Huang, Samyak Gupta, Zhao Song, Kai Li, and Sanjeev Arora. Evaluating gradient inversion attacks and defenses in federated learning. *Advances in Neural Information Processing Systems*, 34:7232–7241, 2021. 1, 3

[11] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Master's thesis, University of Toronto, 2009. 4

[12] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 4

[13] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998. 4

[14] Junnan Li, Caiming Xiong, and Steven CH Hoi. Comatch: Semi-supervised learning with contrastive graph regularization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9475–9484, 2021. 5, 6

[15] Xinjian Luo, Yuncheng Wu, Xiaokui Xiao, and Beng Chin Ooi. Feature inference attack on model predictions in vertical federated learning. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 181–192. IEEE, 2021. 1

[16] Kailang Ma, Yu Sun, Jian Cui, Dawei Li, Zhenyu Guan, and Jianwei Liu. Instance-wise batch label restoration via gradients in federated learning. In *The Eleventh International Conference on Learning Representations*, 2022. 2, 3

[17] 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

[18] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE symposium on security and privacy (SP)*, pages 691–706. IEEE, 2019. 1

[19] Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE symposium on security and privacy (SP)*, pages 739–753. IEEE, 2019. 1

[20] Dario Pasquini, Danilo Francati, and Giuseppe Ateniese. Eluding secure aggregation in federated learning via model inconsistency. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 2429–2443, 2022. 1

[21] Le Trieu Phong, Yoshinori Aono, Takuya Hayashi, Lihua Wang, and Shiho Moriai. Privacy-preserving deep learning: Revisited and enhanced. In *International Conference on Applications and Techniques in Information Security*, pages 100–110. Springer, 2017. 2

[22] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017. 1

[23] Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. Beyond inferring class representatives: User-level privacy leakage from federated learning. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pages 2512–2520. IEEE, 2019. 1

[24] Yuxin Wen, Jonas Geiping, Liam Fowl, Micah Goldblum, and Tom Goldstein. Fishing for user data in large-batch federated learning via gradient magnification. *International Conference on Machine Learning*, 2022. 1

[25] Fan Yang, Kai Wu, Shuyi Zhang, Guannan Jiang, Yong Liu, Feng Zheng, Wei Zhang, Chengjie Wang, and Long Zeng. Class-aware contrastive semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14421–14430, 2022. 5, 6

[26] 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. 1, 2, 3

[27] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 6

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

[29] Joshua C Zhao, Ahmed Roushdy Elkordy, Atul Sharma, Yahya H Ezzeldin, Salman Avestimehr, and Saurabh Bagchi. The resource problem of using linear layer leakage attack in federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3974–3983, 2023. 1, 2, 3, 8

[30] Joshua Christian Zhao, Atul Sharma, Ahmed Roushdy Elkordy, Yahya H Ezzeldin, Salman Avestimehr, and Saurabh Bagchi. Loki: Large-scale data reconstruction attack against federated learning through model manipulation. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 30–30. IEEE Computer Society, 2023. 1, 2, 3, 4, 5, 6

[31] Mingkai Zheng, Shan You, Lang Huang, Fei Wang, Chen Qian, and Chang Xu. Simmatch: Semi-supervised learning with similarity matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14471–14481, 2022. 5, 6, 8

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