

SoK: On Gradient Leakage in Federated Learning

Jiacheng Du^{†,‡,‡} Jiahui Hu^{†,‡,‡} Zhibo Wang^{†,‡,*}
Peng Sun[‡] Neil Zhenqiang Gong[‡] Kui Ren^{†,‡} Chun Chen^{†,‡}

[†]The State Key Laboratory of Blockchain and Data Security, Zhejiang University, P. R. China

[‡]Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security, P. R. China

[‡]College of Computer Science and Electronic Engineering, Hunan University, P. R. China

[‡]Department of Electrical and Computer Engineering, Duke University, USA

{jcd, jiahuihu, zhibowang, kuiren, chenc}@zju.edu.cn, psun@hnu.edu.cn, neil.gong@duke.edu

Abstract

Federated learning (FL) facilitates collaborative model training among multiple clients without raw data exposure. However, recent studies have shown that clients’ private training data can be reconstructed from shared gradients in FL, a vulnerability known as gradient inversion attacks (GIAs). While GIAs have demonstrated effectiveness under *ideal settings and auxiliary assumptions*, their actual efficacy against *practical FL systems* remains under-explored. To address this gap, we conduct a comprehensive study on GIAs in this work. We start with a survey of GIAs that establishes a timeline to trace their evolution and develops a systematization to uncover their inherent threats. By rethinking GIA in practical FL systems, three fundamental aspects influencing GIA’s effectiveness are identified: *training setup, model, and post-processing*. Guided by these aspects, we perform extensive theoretical and empirical evaluations of SOTA GIAs across diverse settings. Our findings highlight that GIA is notably *constrained, fragile, and easily defensible*. Specifically, GIAs exhibit inherent limitations against practical local training settings. Additionally, their effectiveness is highly sensitive to the trained model, and even simple post-processing techniques applied to gradients can serve as effective defenses. Our work provides crucial insights into the limited threats of GIAs in practical FL systems. By rectifying prior misconceptions, we hope to inspire more accurate and realistic investigations on this topic.

1 Introduction

Federated learning (FL) [60] has recently become a widely adopted privacy-preserving distributed machine learning paradigm. In FL, multiple clients collaborate to train a global model orchestrated by a central server for multiple rounds. In each round, clients update the global model locally using

private training data and then transmit gradients to the server for aggregation and global update, thereby alleviating privacy concerns from raw data exposure. Therefore, FL has attracted considerable academic interest and empowered various real-world applications, including mobile services such as Google Keyboard [31], healthcare [49], and finance [57].

However, recent works claim that clients’ data privacy can be compromised by their gradients sharing in FL. [80, 100] Notably, a curious central server can reconstruct their private data by employing gradient inversion attacks (GIAs) [80]. Fig. 1 depicts the development and milestones of the GIA, presenting two forms:

Optimization-based GIA: GIAs typically assume an *honest-but-curious* server and employ optimization-based methods to **passively** reconstruct the victim client’s training data [80, 100]. In this approach, the adversary (server) randomly initializes data and labels, computes gradients based on the same model as the victim client, and iteratively updates these initializations to mirror the ground truth by minimizing the distance between the computed gradient and the client’s shared gradient [100]. Further, it can be categorized into two primary subforms based on the optimization space: GIA with observable space optimization (GIA-O) [26, 30, 33, 55, 58, 88] and GIA with latent space optimization (GIA-L) [2, 42, 52], as defined in Sec. 2. Recent advancements have enabled GIAs to reconstruct larger-batch or higher-dimension data [42, 88], invert more complex model architectures [2, 11, 33], and adapt to various tasks [52, 74] and FL protocols [64, 79, 85].

Analytic-based GIA: Analytic-based methods aim to directly reconstruct training data and labels by formulating and solving equations between gradients and inputs [20, 99]. Initially, these methods were primarily used to infer labels directly from gradients [88, 95], but were limited to inverting low-dimension data on shallow models [20, 99]. To reconstruct higher-dimension inputs, recent efforts have assumed a *malicious* server capable of **actively** crafting [23, 24, 96] or modifying [4, 82] model parameters. Consequently, when a client trains the *malicious* model, the training data leave an “imprint” in the shared gradient, enabling the server to retrieve

*Zhibo Wang is the corresponding author.

[‡]Jiacheng Du and Jiahui Hu contribute equally to this work.

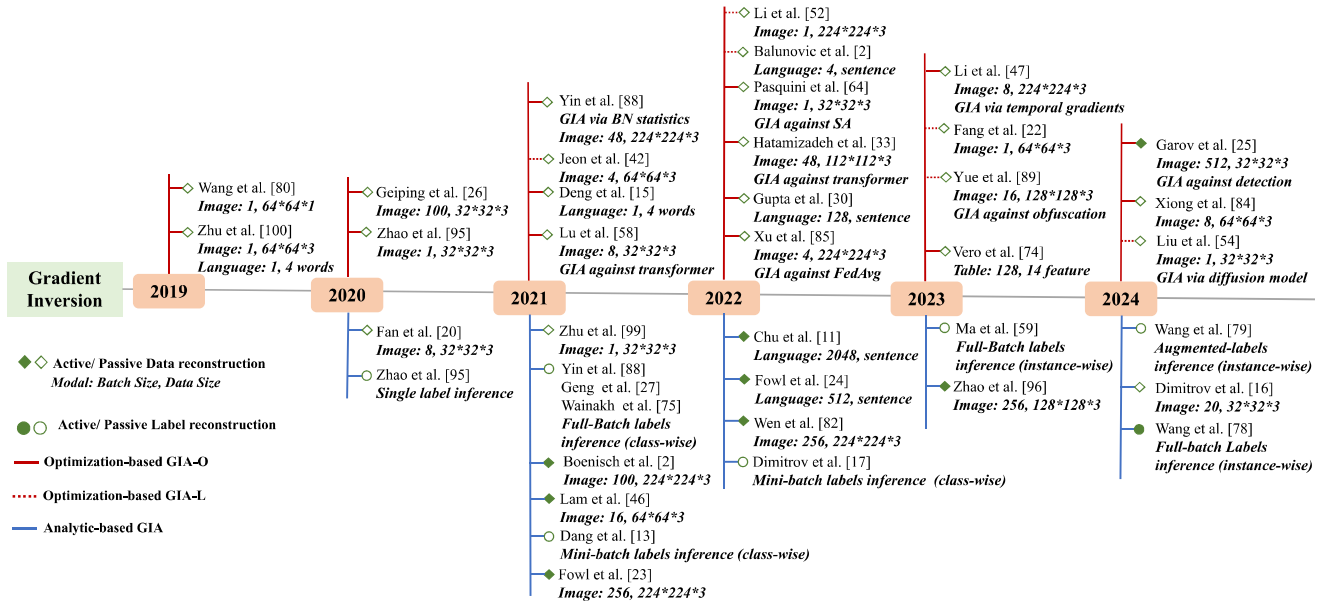


Figure 1: Evolution of Gradient Inversion Attack.

them by solving equations.

Despite the rapid growth and impressive performance, **there remains skepticism about GIA’s real capability and threat to practical FL systems, as often claimed.** On one hand, existing works tend to obsess over employing auxiliary assumptions to achieve heightened performance. Reviewing the milestones, Yin et al. [88] assumed the adversary possesses additional access to BN statistics as side information, while Lam et al. [46] relaxed server assumptions to a malicious extent, allowing the adversary to impractically tamper with the model. On the other hand, these works often evaluate GIAs in settings far from practicality. For instance, literature often assumes a specific client who aggregates all private data into a single batch and updates the model for one step, enabling the adversary to obtain the raw gradients. However, in practical scenarios, the shared gradients are model updates after mini-batch Stochastic Gradient Descent (SGD) and multiple-epoch training [60]. Moreover, the target models are often specially initialized [100] or designed [26, 88] solely to achieve superior reconstruction quality from gradients.

In this work, we conduct a comprehensive study on GIA to better understand its development, properties, and real threats to practical FL systems. Specifically, we make the following key contributions:

I. A survey on GIA. We start by establishing a summary on the development of the GIA. We thoroughly review the related works on the GIA, highlighting the milestones and breakthroughs in performance, as shown in Fig. 1. Moreover, we conduct a *systematization of the GIA* along three dimensions (Sec. 2): *threat model, attack, and defense*, as detailed in Tab. 7 in Appx C. Notably, we characterize the threat model of the GIA, categorizing the assumptions based on their practical

accessibility to potential adversaries.

II. Extensive investigations on GIA in practical FL. We identify *three fundamental aspects* that influence GIA’s effectiveness in FL: *training setup, model, and post-processing* (Sec. 3). Subsequently, we conduct comprehensive theoretical analyses and empirical evaluations on GIAs from the aspects (Sec. 4, Sec. 5, and Sec. 6) across diverse settings. Our investigations bridge the gap between literature and practice, revealing the real threats posed by GIAs in FL systems.

III. Analyses and insights for GIA in practical FL. We provide an in-depth analysis of GIA’s properties and the associated real threats, offering a summary of key insights. Our findings indicate that, despite its purported effectiveness, GIA is notably *constrained, fragile, and easily defensible*, as supported by the following observations:

(a) **GIA presents inevitable bottlenecks against practical training setups.** We investigate how the client’s local training impacts GIAs from training configuration and training data. Specifically, we theoretically prove that *as the number of local updates increases, reconstruction becomes much more difficult* (Sec. 4.1). Moreover, we evaluate the capabilities of SOTA GIAs across a wide range of data dimensions. With theoretical proofs, we indicate the *GIA’s bottleneck in data reconstruction as data dimension growth* (Sec. 4.2.1). Besides, we identify that *data content significantly affects GIAs with two canary cases* (Sec. 4.2.2). Specifically, GIA fails to reconstruct semantic details containing crucial private information. For generative adversaries, out-of-distribution (OOD) data constrains GIA’s performance.

(b) **GIA is extremely sensitive to the model, including training stage and architecture.** We propose a novel Input-Gradient Smoothness Analysis (IGSA) method to quantify

and explain the model’s vulnerability to the GIA during the FL training process (Sec. 5.1). Surprisingly, our findings indicate that *the GIA exclusively works in the early training stages*. Furthermore, we undertake a *deep investigation into the strong correlation between model architecture and the GIA*. Our analyses highlight that commonly employed structures (e.g., skip connections [34]), and even seemingly insignificant micro designs (e.g., ReLU), significantly impact the model’s resilience against GIAs (Sec. 5.2).

(c) *Even trivial post-processing measures applied to gradients in practical FL systems can effectively defend against GIAs while maintaining model accuracy*. We evaluate four post-processing techniques, considering the privacy-utility trade-off within a practical FL setting. We show that *even when confronted with SOTA GIAs, clients can readily defend against them by employing post-processing strategies (e.g., quantization [1]) to obscure the shared gradients* (Sec. 6).

2 Systematization of Gradient Inversion

2.1 System Model

We consider an FL system consisting of a server and M clients, each with a private training dataset containing N pairs of data (\mathbf{x}) and labels (\mathbf{y}). The M clients collaboratively train a global model W^g over T rounds under the coordination of the server. In each round t , the server selects K clients and sends the current global model W_t^g to them for local training. Each client k performs E epochs of local training utilizing mini-batch SGD with a batch size of B . Consequently, each client performs $U = EN/B$ local updates. The different configurations of E and B give rise to two FL protocols:

(1) **FedSGD** (Federated Stochastic Gradient Descent) [45]: Each client k aggregates all N local training data samples into a batch ($B = N$) and executes a single epoch of local training ($E = 1$), updating W_t^g for one time. The computed gradient $\nabla_k W_t^g$ is uploaded to the *server*. The *server* aggregates the collected gradients and updates the global model as follows: $W_{t+1}^g \leftarrow W_t^g + \eta \sum_{k=1}^K \nabla_k W_t^g$.

(2) **FedAvg** (Federated Averaging) [60]: Client k conducts $E > 1$ epochs of mini-batch SGD with $B \leq N$ for local training and shares an updated local model W_{t+1}^k . The server receives the K local models and computes their average to obtain the updated global model: $W_{t+1}^g \leftarrow \frac{1}{K} \sum_{k=1}^K W_{t+1}^k$.

For the server, the primary distinction between the two protocols lies in the type of parameters shared. In FedSGD, the server directly receives the **gradient**, which captures the precise parameter changes of W_t^g after a one-step gradient descent. In contrast, clients share the updated models in FedAvg. Consequently, the server could only obtain the parameter **updates** by $W_{t+1}^k - W_t^g$, which are the cumulative parameter changes after multiple steps of mini-batch gradient descent.

2.2 Threat Model

Existing studies primarily focus on scenarios where the server, receiving gradients from clients, acts as the adversary conducting GIAs. Therefore, we conduct a summary on the threat model of existing GIAs based on the following three aspects:

(1) *Goal*. The goal of the adversary is to reconstruct the client’s **data** and **labels** from the shared gradients. Initial investigations (e.g., [100]) attempt to reconstruct the data and labels simultaneously. However, follow-up works [88, 95] reveal that labels could be inferred directly from the gradients without explicitly solving them. As a result, subsequent GIAs focus on reconstructing either labels or data separately. Most GIAs concentrate on data reconstruction, aiming to retrieve more accurate private data, assuming that labels are either already known or can be reliably inferred beforehand. Meanwhile, investigations such as [13, 17, 59] explore label inference, which has two implications: First, foreknowledge of labels aids subsequent data reconstruction. Second, labels themselves contain sensitive information such as a user’s purchase history [50]. So far, GIAs have been capable of inferring the presence of certain classes (class-wise) [88] and further determining the number of instances within each class (instance-wise) [59] from a full-batch [27, 59, 75, 78, 88] or multiple mini-batches of data [13, 17].

(2) *Capacity & Server’s Trustworthiness*. Most existing GIAs presume an **honest-but-curious** server [2, 13, 15, 17, 20, 26, 27, 30, 33, 42, 47, 52, 58, 59, 75, 80, 85, 88, 95, 99, 100], which implies that the server merely analyzes shared gradients **passively** without interrupting the training process. Consequently, the victim client remains unaware, ensuring the stealthiness of GIAs. Furthermore, some studies consider that a **malicious** server not only analyzes the gradient but could also **actively** interfere with the learning process through malicious behaviors, thereby extracting more information about the input by gradients. Specifically, they consider that the server can craft [11, 23, 64] or modify [4, 24, 82, 96] the model parameters, enabling the adversary to achieve better reconstruction results. Besides, unlike the **honest-but-curious** server, such **malicious** behaviors could be easily detectable by clients.

(3) *Assumption*. Assumptions specify the adversary’s knowledge and allowed behaviors. Reflecting on the evolution of the GIA, diverse assumptions offer additional advantages to the adversary and even serve as the key to the asserted impressive performances [41]. Herein, we categorize and rank existing assumptions based on their accessibility to the adversary within practical FL systems and access GIAs in Tab. 7.

[Level 0]: Basic information refers to the necessities for gradient inversion, including gradient, model, data dimension, and number of data samples N at victim clients, which are readily accessible to the server in practical FL systems.

[Level 1]: Priors refer to side information, including established patterns or observations that are readily accessible. For instance, Geiping et al. in [26] utilized total variation [67] as

a prior, an established pattern of smoothness among neighboring pixels, effectively regulating the reconstruction process. Additionally, certain priors stem from observations on gradients, for example, Lu et al. in [58] discovered that the cosine similarity of gradients in the positional embedding layer is substantial for two similar images. Consequently, they designed a regularization term to invert vision transformers [18].

[Level 2]: Data distribution refers to the statistical characteristics of the client’s private dataset. Knowing the distribution enables the adversary to pre-train a generator, thereby improving the data reconstruction performance [2, 22, 42, 52, 80, 84, 89] (further elaborated in Eq. (2)). In FL, clients are not mandated to disclose their data distribution to the server. Nevertheless, given the server’s approximate knowledge of the task, it may occasionally estimate the distribution using open-source datasets. For instance, if the server knows that the client possesses facial data, it may utilize datasets like FFHQ [44] for generator pre-training.

[Level 3]: Client-side training details refer to settings such as local learning rates, epochs, mini-batch sizes, optimizer, etc. Xu et al. in [85] introduced a GIA capable of quickly approximating client-side multi-step updates in FedAvg, necessitating access to these training details. However, such information is typically unavailable to the server.

[Level 4]: BN statistics are the mean and variance of batched data acquired at BN layers that may be uploaded along with gradients in FL. They were initially utilized in the GIA by Yin et al. [88] and subsequently in several works [33, 41, 47, 85]. However, in practice, the BN layer can be easily substituted [83], and clients typically are not required to upload their BN statistics [51].

[Level 5]: Malicious behavior involves an adversary’s active manipulation of protocols [46], models and other components in FL to enhance the reconstruction performance. Existing studies concentrate on crafting [11, 23, 64] or modifying [4, 24, 82, 96, 97] model parameters. Notably, recent works [4, 23] presume the existence of a **large fully connected layer** at the **front** of the model, yet such anomalous designs can be easily detected [25]. Consequently, these behaviors lack stealthiness and practical applicability.

In summary, **Level 0** and **1** are assumptions easily accessible to the adversary in practical FL systems. On the other hand, **Level 2, 3,** and **4** are deemed strong assumptions as practical clients are not required to furnish this information to the server, although the adversary might approximate or acquire it under certain circumstances. Additionally, we contend that **malicious behavior (Level 5)** is over-assumed in practical FL systems due to its inherent lack of stealthiness.

2.3 Attack

GIAs employ two main attack strategies and involve several modalities based on the different types of FL tasks.

(1) *Strategy*. The attack strategies employed by GIAs can be

categorized into two forms: **optimization-based** and **analytic-based**, as illustrated in Fig. 1.

The process of **optimization-based** GIA involves iteration from a random initialization towards an approximation of the ground truth data, guided by a loss function of gradient similarity. It can be further categorized into two primary sub-forms based on the optimization space:

1) GIA-O: At training round t , the victim client holding N pairs of data \mathbf{x} and labels \mathbf{y} shares its gradient ∇W to the server after local training ($B \leq N, U \geq 1$). The adversary obtains ∇W and generates N pairs of randomly initialized data x' and labels y' with the same dimension of ground truth. Following the loss of gradient similarity, N pairs of initialization are updated until they approximate the ground truth (\mathbf{x}, \mathbf{y}) pairs:

Definition 1 (GIA with Observable Space Optimization).

$$\{\mathbf{x}'_n, \mathbf{y}'_n\}_{n=1}^N = \underset{\{\mathbf{x}'_n, \mathbf{y}'_n\}_{n=1}^N}{\operatorname{argmin}} \operatorname{Dist} \left(\frac{1}{N} \sum_{n=1}^N \frac{\partial \ell(\mathbf{x}'_n, \mathbf{y}'_n)}{\partial W} - \nabla W \right) + \alpha \mathcal{R}, \quad (1)$$

where $\operatorname{Dist}(\cdot)$ is a distance metric between two vectors (such as Euclidean distance [80, 100] and cosine similarity [26]), \mathcal{R} represents regularization terms, e.g., total variation [67] and BN statistics [88], and α is the weighting factor.

2) GIA-L: The fundamental concept behind GIA-L mirrors that of GIA-O, albeit with a distinction: GIA-L involves the initialization and optimization of N pairs of latent vectors \mathbf{z}' and labels \mathbf{y}' , reconstructing the private data by a generative adversarial network (GAN) [29] denoted as G .

Definition 2 (GIA with Latent Space Optimization).

$$\{G(\mathbf{z}'_n), \mathbf{y}'_n\}_{n=1}^N = \underset{\{\mathbf{z}'_n, \mathbf{y}'_n\}_{n=1}^N}{\operatorname{argmin}} \operatorname{Dist} \left(\frac{1}{N} \sum_{n=1}^N \frac{\partial \ell(G(\mathbf{z}'_n), \mathbf{y}'_n)}{\partial W} - \nabla W \right) + \alpha \mathcal{R}. \quad (2)$$

Analytic-based GIA (GIA-A) aims to reconstruct training data precisely by formulating and solving equation systems that relate gradients to inputs. Early studies solve this problem by recursively inferring feature maps layer by layer from the gradients until the input is reconstructed [20, 99]. However, these methods are limited to shallow, fully connected networks or the reconstruction of a single image. To improve the effectiveness, subsequent studies assume a more powerful malicious server capable of either **crafting** [23, 24, 96] or **modifying** [4, 82] model parameters:

1) GIA-A with malicious parameter crafting: This type of method reconstructs the inputs by crafting and inserting malicious structures into the benign model. Fowl et al. [23] introduce the “*imprint module*”, a huge linear layer inserted at the model’s front maliciously. Through specialized weight initialization, this module enables the reconstruction of inputs

exhibiting target properties (e.g., specific image brightness) from its gradients. This concept is later extended to language transformers. Fowl et al. [24] first disable all the attention layers and most of the feed-forward layers, and then *insert the imprint modules into the feed-forward layers to separate and reconstruct tokens*. Zhao et al. [96] further proposed a *sparsified imprint module* to mitigate the computational overhead for such methods.

2) GIA-A with malicious parameter modifying: This type of method does not insert any new structure but manually modifies a wide range of model parameters. Boenisch et al. [4] reconstruct the inputs by *modifying the parameters of the first fully-connected layer*. Specifically, they fine-tuned the layer’s weights W using an auxiliary dataset with the same distribution as the client’s private data. This modification ensures that a sample i in a batch activates only the neuron corresponding to row W_i , enabling the separation and reconstruction of batched samples. However, this method is limited to simple fully connected networks with ReLU activation functions. Wen et al. [82] further refined this technique by *modifying the parameters in the classification layer* associated with the target class, setting all other parameters to zero. This modification ensures that the batched gradients reflect only the gradient of the target sample.

Limitations of GIA-A in practice: Currently, *GIA-As face significant challenges in balancing utility and stealth in practical FL systems*. Without engaging in malicious behavior, an honest-but-curious server can only reconstruct a single image using shallow models through analytical methods [20,99], posing limited threats in practical settings. While with malicious behaviors, the attacks could be easily detectable by clients. Such behaviors typically involve inserting highly unusual structures into the model or making large-scale parameter modifications, rendering the malicious model highly conspicuous to clients [4,26,82]. Moreover, recent studies have shown that the anomalous functionality of these malicious models can also be further detected in the gradient space [25].

(2) *Modality*. As demonstrated in Fig. 1, most of the GIAs focus on Computer Vision (CV) tasks. Recent efforts have started investigating GIAs on Natural Language Processing (NLP) [2, 15, 30] tasks. However, GIAs currently poses a limited threat on language models, and we make a further discussion in Sec.7. Moreover, a recent study by Vero et al. in [74] has explored GIA’s applicability on tabular data.

2.4 Defense

Cryptographic methods, such as secure multi-party computation [5, 62] and homomorphic encryption [10], have been applied in various privacy-preserving tasks. However, in FL systems, these techniques often result in significant computation and/or communication overheads [77] (discussed in Appx.C). Consequently, recent research efforts mainly opt for either **perturbing representations** [68, 71] or **employ-**

ing post-processing of gradients [52, 100] to defend against GIAs. **Representation perturbation** relies on the premise that if the representations during the forward propagation process are perturbed, the gradient would struggle to accurately convey features about the input. Several approaches, such as pruning [71] or the integration of a variational module [68], have been employed to implement such perturbations. However, these methods have demonstrated less effectiveness against current GIAs [89]. Another defense approach is **post-processing on gradients**. Techniques like compression [52] and sparsification [89] mislead adversaries by perturbing the gradients. These post-processing methods are commonly adopted in practical FL systems and exhibit potential in countering GIAs.

Our focus: To explore the real threat of GIAs in practical FL systems, our study assumes that the adversary is an **honest-but-curious** server. We focus on **optimization-based GIAs** for **image reconstruction** tasks because of its applicability and widespread interests. In addition, we consider **gradient post-processing** techniques as defensive methods due to their common applications in practice.

3 Rethinking GIA in Practical FL

In this section, we identify three fundamental aspects affecting GIA. By examining the gap between the literature and practice, we highlight three key research questions (RQs) to reveal the potential threats posed by GIA in practical FL.

A practical FL system is a distributed machine learning framework where clients locally train models using *several private samples over multiple iterations* (e.g., mini-batch SGD). After local training, the clients share *parameter updates* with a central server, which aggregates these updates to produce a new global model. This process repeats over multiple rounds until the convergence.

Therefore, let us go back to the drawing board to identify the fundamental aspects that affect GIAs in practical FL systems. Since GIAs reconstruct local data from shared updates, the critical step lies in understanding how these updates are generated and shared. First, **the local training setup serves as the foundation**, which determines how private data participate in training. Second, **the model maps the inputs to parameter update**, through forward and backpropagation. Finally, **the raw update often undergoes post-processing before being shared**, to meet the security and efficiency requirements in deployments. For example, updates may be compressed to enhance communication efficiency.

As illustrated in Fig. 2, the transition from training data to the shared update follows these three critical steps. Consequently, evaluating the risk of GIA requires careful consideration of these aspects. **Training setup**. The literature often assumes that the victim combines all its local data into a single batch ($B = N$) and updates W_t^s for only one step ($U = 1$), then share the gradient ∇W_t^s , which benefits the adversary a lot.

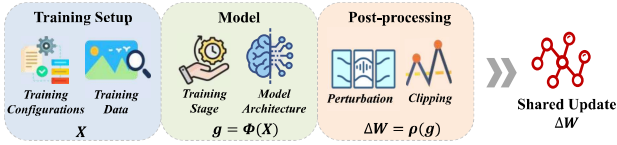


Figure 2: Three Fundamental Aspects of GIA.

However, in practice, the client updates W_t^s with mini-batch SGD, where W_{t+1}^k are shared after training W_t^s for multiple mini-batch steps. Therefore, the adversary could only obtain the update $W_{t+1}^k - W_t^s$. Besides, the training data are quite different in practice. First, the practical data dimensions (e.g., batches of 64 images with the resolution of 128×128) are much larger than those in the literature (e.g., reconstruct 8 images with the resolution of 32×32). Also, even if the data are of the same dimension, do they have the same risk of privacy leakage given the diversity of content? Based on these discussions, we aim to answer:

- **RQ1:** How do the training setups affect GIAs in practical FL systems? (Sec. 4)

Model. The model plays a crucial role in mapping training data to gradient, but the literature treats it as a black box and underestimates its significance. In pursuit of performance, GIAs are often evaluated on specific architectures (e.g., ResNet-50 [33, 88] pre-trained with MOCO-V2 [9]) or models with explicit initialization (a wide range of values from a uniform distribution [26, 42, 95, 100]). In practice, an adversary may launch a GIA against any model, in any training stage, rather than a tailored one. Therefore, uncovering the model’s black box and investigating its vulnerability is important to expose GIA’s privacy threats:

- **RQ2:** What are the factors that influence the model’s vulnerability to GIA? (Sec. 5)

Post-Processing. Literature often assumes that the adversary could obtain the raw gradients directly. However, in practical FL systems, clients often perform post-processing on gradients before sharing them. For instance, gradients are commonly quantized to alleviate communication overhead [53]. Essentially, post-processing induces gradient drifting and potentially disrupts the adversary, which reminds us of an attractive trade-off problem:

- **RQ3:** Can FL systems naturally defend against GIAs with post-processing techniques, while ensuring the utility? (Sec. 6)

4 Evaluation on Training Setup

The client’s training setup, specifying what and how training data is utilized to compute the gradient, fundamentally affects the difficulty of reconstruction in practical FL systems. In this section, we investigate how the client’s training setups affect GIA from two critical aspects: *training configurations* and *training data*.

4.1 Training Configurations

The client’s training configurations depict how it organizes the data for training. The local dataset, consisting of N samples, is divided into $\frac{N}{B}$ mini-batches. After completing E epochs, the client shares the model $W_{E,0}$ following a total of $U = E \times \frac{N}{B}$ updates. Previous studies have often assumed ideal conditions ($B = N, E = 1$) to maximize the reconstruction quality of N samples, where the adversary has access to the exact gradient of full batch data with a single update. However, in practical FL systems, an adversary can only perform GIA by approximating $(\mathbf{W}_{E,0} - \mathbf{W}_{0,0})/\eta$ where multiple gradients are squeezed into one update. In this subsection, we first analyze the impact of multiple updates on GIA in full-batch settings and then investigate the effectiveness of GIA against more general mini-batch SGD.

4.1.1 Evaluate GIA against Full-Batch Updates

We begin by examining the impact of multiple updates on GIA in the full-batch setting, where the victim client configures $B = N$ and uploads the updated model after U local updates. First, we formalize the relationship between gradient and inputs and derive the reconstruction error introduced by multiple local updates. We then empirically evaluate the performance of GIA under varying numbers of local updates.

Theoretical Analysis. Consider a binary classification task ($y \in \{-1, 1\}$), and an L -layer fully connected network with activation function σ :

$$\mu = y\mathbf{W}_L\mathcal{F}_{L-1}, \quad (3)$$

$$\mathcal{F}_{L-1} = \sigma(\mathbf{W}_{L-1}\mathcal{F}_{L-2}) \quad \mathcal{F}_{L-2} = \sigma(\mathbf{W}_{L-2}\mathcal{P}(\mathbf{x})), \quad (4)$$

where μ represents the logit, \mathbf{W}_L denotes the augmented parameter matrix (including weight and bias) of the L -th layer, \mathcal{P} represents all layers previous to $L - 2$, and \mathbf{x} denotes the flattened vector of input data. Given the network’s logit μ , the loss function can be expressed as:

$$\ell = \log(1 + e^{-\mu}). \quad (5)$$

Lemma 4.1. For a fully connect network, the input \mathbf{x} can be iteratively derived from gradient (Eq. (7)) by first solving the logit μ (Eq. (6)):

$$\frac{\partial \ell}{\partial \mathbf{W}_L} \cdot \mathbf{W}_L = \frac{-\mu}{1 + e^\mu}, \quad (6)$$

$$\mathbf{x} = y\mu\mathbf{W}_L^\top \prod_{l=1}^{L-1} \mathbf{W}_l^\top \odot \sigma^{-1}. \quad (7)$$

Lemma 4.1 establishes a connection between the input \mathbf{x} and the gradient by logit. If only one local update is performed, the adversary has the chance to reconstruct x^* precisely, as shown in Fig. 3. However, if the client performs multiple updates, we can further deduce the reconstruction error that it imposes on the adversary:

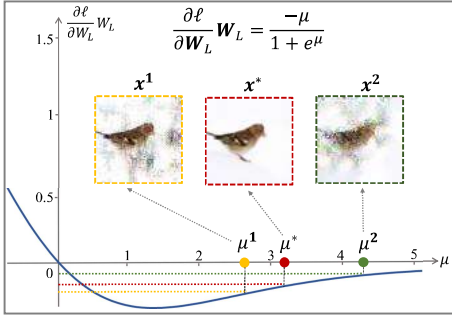


Figure 3: Dependence between Gradient $\frac{\partial \ell}{\partial \mathbf{W}}$, μ and Input \mathbf{x} . The ground-truth gradient corresponds to μ^* , \mathbf{x}^* . When the gradients are obfuscated, they correspond to the inaccurate μ^1, μ^2 , and $\mathbf{x}^1, \mathbf{x}^2$.

Theorem 4.2. Suppose the client updates the initial model \mathbf{W}^0 with the ground truth \mathbf{x}^* for U times to obtains \mathbf{W}^U . The adversary performs GIA by approximating $\frac{\mathbf{W}^U - \mathbf{W}^0}{\eta}$ and obtains the reconstructed \mathbf{x}^{rec} . The reconstruction error can be denoted as:

$$\mathbf{x}^{rec} - \mathbf{x}^* = y(\mu^{rec} - \mu^*) \mathbf{W}_L^\top \prod_{l=1}^{L-1} \mathbf{W}_l^\top \odot \sigma^{-1}, \quad (8)$$

where $\mu^{rec} - \mu^*$ can be approximated by:

$$\mu^{rec} - \mu^* \approx \frac{(e^{\mu^*} + 1)^2 \sum_{u=1}^{U-1} \frac{\partial \ell}{\partial \mathbf{W}_u} \mathbf{W}_L^0}{\mu^* e^{\mu^*} - e^{\mu^*} - 1}. \quad (9)$$

A detailed proof is provided in Appx. A.

Theorem. 4.2 reveals the obfuscation caused by multiple updates for an adversary to reconstruct input \mathbf{x} . Specifically, if $U > 1$, the adversary can only obtain the update $\frac{\mathbf{W}^U - \mathbf{W}^0}{\eta}$ which consists of the precise gradient $\frac{\partial \ell}{\partial \mathbf{W}^0}$ and $U - 1$ redundant terms $\sum_{u=1}^{U-1} \frac{\partial \ell}{\partial \mathbf{W}_u}$. Thus, the reconstructed \mathbf{x}^{rec} will drift from \mathbf{x}^* , as the examples (\mathbf{x}^1 and \mathbf{x}^2) in Fig. 3 show. Further, Eq. (8) and (9) show that the reconstruction error gradually accumulates and obfuscates the GIA as U increases.

Empirical Analysis. To empirically validate our theoretical conclusion, we evaluate the performance of the two SOTA GIAs (GIA-O with p_{TV} and GIA-L with pretrained DcGAN [66]) as the number of local updates (U) increases on the datasets CIFAR10 (C10, $N = B = 4$) and CIFAR100 (C100, $N = B = 8$). We use the metric *Learned Perceptual Image Patch Similarity (LPIPS)* [93] to measure GIA’s performance, where a smaller value represents higher reconstruction quality. The detailed setup is described in Appx. B. Tab. 1 demonstrates that when U is 1 or 2, the adversary can still reconstruct the data. However, as U increases, the *LPIPS* value exceeds 0.1, indicating that the reconstructed images are nearly unrecognizable. Furthermore, the reconstruction quality deteriorates as U increases, which indicates that the redundant terms in Eq. (9) gradually accumulate, and inaccuracy in the adversary’s update approximation grows.

Table 1: GIA against Local Updates (**LPIPS**↓). **Bold text** represents LPIPS value exceeds 0.1, indicating that the reconstructed image is completely unrecognizable.

Dataset	GIA	Number of Update (U)				
		1	2	4	6	8
CIFAR10	O	0.0189	0.1231	0.1499	0.1921	0.2522
	L	0.075	0.0952	0.1554	0.1607	0.1984
CIFAR100	O	0.0049	0.0404	0.1165	0.1653	0.2268
	L	0.0288	0.063	0.1205	0.1514	0.1852

Table 2: GIA against Mini-Batch SGD (**LPIPS**↓).

Dataset	GIA	Case	Number of Mini-batch						
			1	2	3	4	6	8	10
C10	O	WST.		0.061	0.1064	0.1117	0.1456	0.1314	0.1426
		BST.	0.0172	0.0641	0.1088	0.1075	0.1398	0.1403	0.1445
	L	WST.		0.0672	0.0834	0.1043	0.1436	0.1285	0.1376
		BST.	0.0669	0.0833	0.0859	0.1017	0.1444	0.1318	0.1451
C100	O	WST.		0.0732	0.0792	0.1003	0.1063	0.1193	0.1273
		BST.	0.0045	0.0581	0.0759	0.0928	0.1139	0.1183	0.1271
	L	WST.		0.0725	0.0788	0.0931	0.1017	0.1100	0.1174
		BST.	0.0436	0.0776	0.0817	0.093	0.1016	0.1076	0.1178

4.1.2 Evaluate GIA against Mini-Batch Updates

Here, we explore a more practical scenario in which the client shares the \mathbf{W}_U after several local mini-batch SGD updates ($B < N, U = N/B$). We examine two cases based on whether the adversary knows B . **Worst case (WST):** Previous studies [85, 100] often assume that the adversary has full knowledge of B and other local training configurations. This knowledge allows the adversary to simulate the client’s mini-batch updates more accurately, closely matching $\frac{\mathbf{W}_U - \mathbf{W}_0}{\eta}$, which maximizes the effectiveness of GIA. **Best case (BST):** However, in practice, the client is not obligated to provide local training details to the server, thus the adversary can only approximate the update by conducting one-step SGD ($B = N, U = 1$). This scenario significantly reduces the effectiveness of GIA because the adversary, lacking knowledge of the training configurations, cannot accurately approximate the update.

We evaluate GIAs in both cases, as shown in Tab. 2. We varied the number of mini-batches by adjusting the value of N/B on the CIFAR10 ($B = 4$) and CIFAR100 ($B = 8$) datasets. The results indicate that the reconstruction fails as the number of mini-batches increases for both cases (indicated by the bold *LPIPS* value exceeding 0.1), even if the adversary has access to local training configurations. Additionally, it is important to note that the maximum number of updates evaluated in Tab. 2 is 10, which is significantly lower than what is typical in the practical FL system. *Given that each client typically possesses several thousand to tens of thousands of data points, the number of local updates would be much higher, thereby posing an even greater challenge to GIA.*

(Insight 4.1) The training configurations significantly impact GIA’s effectiveness. Specifically, as the number of local updates increases, the reconstruction becomes increasingly challenging and even fails.

4.2 Training Data

The primary goal of GIAs is reconstructing training data. To investigate its impact, we evaluate the performance of GIAs on two basic data properties, **dimension** and **content**. To explore the impact of data dimension, we first give a theoretical upper bound on whether the adversary can accurately reconstruct the data as dimension grows, and then empirically evaluate the SOTA GIAs across various resolutions and batch sizes in practical FL. Subsequently, we investigate the influence of image content on GIAs—an interesting but largely overlooked aspect in existing literature.

4.2.1 Data Reconstruction across Wide Dimensions

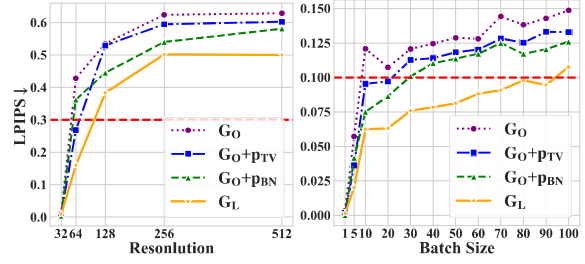
Intuitively, reconstruction becomes more difficult with higher data dimensions (larger batch sizes, higher resolution). This raises a critical question: **is there a theoretical upper bound that makes it impossible for an adversary to accurately reconstruct data beyond a certain dimension?**

Theorem 4.3. Suppose the “input-gradient” function $\frac{\partial \ell}{\partial \mathbf{w}} = \Phi(\mathbf{x})$, where the gradient $\frac{\partial \ell}{\partial \mathbf{w}} \in \mathbb{R}^{1 \times p}$, the input data $\mathbf{x} \in \mathbb{R}^{B \times D}$, B and D are batch size and data size. If $p < B \times D$, then there exists at least one mask Δ such that, $\Phi(\mathbf{x}) = \Phi(\mathbf{x} + \Delta)$.

A detailed proof is provided in Appx. A.

Theorem. 4.3 suggests that when the data dimension exceeds the number of model parameters, there will always exist at least one fake data point $\mathbf{x} + \Delta$ that differs from the ground truth \mathbf{x} but yields the same gradient. In other words, *even if the adversary has the strongest ability to approximate the received gradient, it will still be unable to accurately reconstruct the ground truth.* In practice, a bottleneck emerges that hinders data reconstruction when the dimensionality surpasses a certain threshold. Due to the model’s sparsity and the adversary’s limited computational resources, this threshold will be substantially lower than the theoretical value, p .

To demonstrate how the increasing data dimension restricts the effectiveness of GIAs, we evaluate four GIAs across various settings, including batch sizes ranging from 1 to 100 and image resolutions from 32×32 to 512×512 . We evaluate the baseline G_O (GIA-O with gradient-matching loss only) and three SOTA GIAs: G_{O+pTV} (GIA-O with prior total variation [26]), G_{O+pBN} (GIA-O with prior BN statistics [88]), and G_L (GIA-L [42, 52]). We evaluate GIAs under FedSGD [45] using ResNet-18 on CIFAR10, CIFAR100, and ImageNet-1K [14] datasets. The performance of GIA is measured by *LPIPS*. Detailed setup is provided in Appx. B.



(a) GIA on Resolutions (b) GIA on Batch Sizes

Figure 4: GIA on Series of Data Dimensions.

Fig. 4 illustrates the statistical results, and *LPIPS* values above the red lines mean that the reconstructed images cannot disclose the privacy of the original images. Our findings indicate that (1) *GIA performs well in reconstructing low-dimensional data* (resolution $< 64 \times 64$ and batch size < 30), but has difficulty in reconstructing high-dimension data. (2) *GIA-L presents a better performance in reconstructing high-dimensional data compared to GIA-O.* GIA-L benefits from two aspects: First, GIA-L has a smaller search space, which allows the optimizer to better find the optimal. Second, generative knowledge guarantees the fidelity.

(Insight 4.2.1) An increase in data dimension weakens the effectiveness of GIA, and a bottleneck exists that prevents adversaries from effectively reconstructing higher-dimension data in practice.

4.2.2 Data Reconstruction across Various Contents

In practical scenarios, clients store diverse training data that share the same dimension and label space but hold various contents. Therefore, **does GIA pose the same level of privacy risk for data with different contents?**

To explore the effectiveness of GIA against various data contents, we investigate two “canary-testing” categories. **Category 1: Data with semantic details.** In practice, the degree of data privacy leakage is often determined by the quality of reconstructed key semantic details, rather than overall similarity. **Category 2: Out-of-distribution (OOD) data.** This category is specific to GIA-L. Previous studies often assume that the adversary has access to a dataset with an identical-distribution (ID) to the client’s data for generator pre-training. However, in practice, the adversary faces the OOD challenge because the client is not required to disclose its data distribution. As a result, the adversary must rely on public datasets for pre-training, leading to distributional bias.

I. Semantic details diminish the effectiveness of GIAs. To reveal the impact of semantic details on GIAs, we choose two types of attacks: GIA-O with priors including both p_{TV} and p_{BN} , and GIA-L with BigGAN’s generator [6] pre-trained on ImageNet-1K [14]. The experiments are conducted on ImageNet-1K dataset, and setup details are provided in the Appx. B. As shown in Fig. 5, the reconstructed images re-

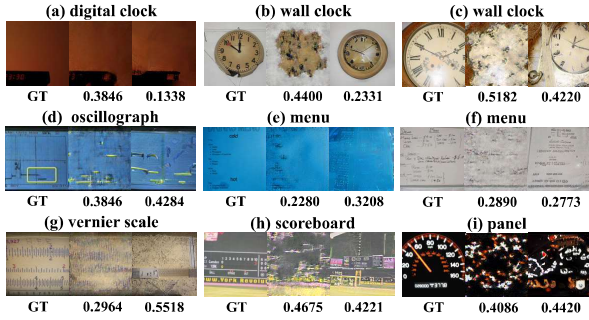


Figure 5: Failure to Reconstruct Semantic Details, Limited Privacy Leakage. (**Left:** Ground-Truth, **Middle:** Results of GIA-O, **Right:** Results of GIA-L, LPIPS \downarrow).

veal very little private information because all the crucial details are lost. Specifically, Fig. 5 (a)-(c) show the results of reconstructing *clock* images. Although the contours can be reconstructed, the crucial semantic details, such as the correct time, are completely missed. The neglect of semantic details is further exemplified in Fig. 5 (d)-(f), where only the background can be reconstructed without recognizing any critical contents (e.g., numbers, text). In addition, we find that images with semantic details are much more challenging to fit and tend to mislead GIAs, as shown in Fig. 5 (g)-(i). The reasons for GIA’s negligence of semantic details stem from two aspects: (1) the gradient reflects more the capture of category features than semantic details; (2) the pre-trained generator only learns class-wise features rather than identical details.

II. OOD data impedes the generalization capability of GIA-L. To explore the effectiveness of GIA-L against OOD data, we assume that the adversary has access to ImageNet-1K [35] dataset, and evaluate the GIAs on four OOD datasets. **Places** tests the performance of GIA-L in images involving complex background information [40], **Textures** [12] and **Objects** [35] contain various pattern or object distributions, and **Styles** is designed for cross-style generalization [48].

Fig. 6 demonstrates the reconstruction performance of GIA-L on ID and OOD data. The results indicate that GIA-L performs badly on OOD data due to its limited generative capability. **For each pair of samples labeled similarly, the LPIPS values for reconstructing ID images are much lower than those of reconstructing OOD ones.** In the case of Textures, GIA-L are much familiar with “discrete and transparent” bubbles than “dense and blue” ones, as demonstrated in Fig. 6, respectively. Another example involves different styles of guitars, where GIA-L lacks knowledge about the characteristics in *art* or *cartoon* style.

(Insight 4.2.2) The effectiveness of GIA varies against various data contents. Thus, GIA’s privacy risks are often overestimated, particularly when reconstructing semantic details and OOD data.

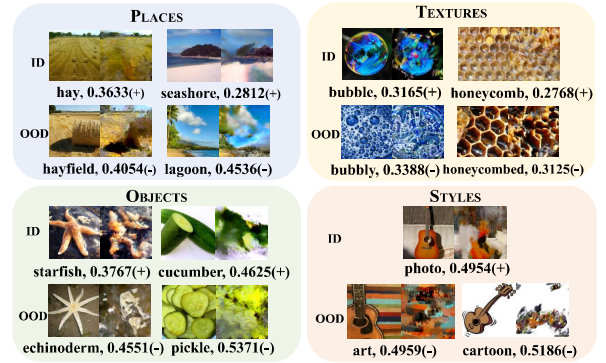


Figure 6: OOD Challenge for GIA-L. (**Left:** Ground-Truth, **Right:** Reconstructed Image, LPIPS \downarrow). Each column is a pair of samples with **similar labels**, **Up:** Sample from ImageNet-1K [14](ID) dataset, **Bottom:** Sample from the OOD datasets).

5 Evaluation on Model

The model determines the mapping from training data to the gradient. During the FL training phase, the adversary acquires the trained model and its shared gradient at a particular stage (round) for GIA. In this section, we investigate the impact of the model on gradient inversion from the perspectives of **stage** and **architecture**. To illustrate and quantify the model’s vulnerability to GIAs at different stages, we introduce a novel **Input-Gradient Smoothness Analysis (IGSA)** method. Through IGSA, we empirically assess the resistance of nine models against GIAs at different stages during the whole FedAvg training phase. Subsequently, we explore the sensitivity of GIAs to model architecture.

5.1 Training Stage

5.1.1 Input-Gradient Smoothness Analysis (IGSA)

The “input-gradient” function, denoted as $\Phi(\cdot)$, which includes both forward and backward propagation, maps input data to gradients, forming the foundation of GIA’s objective function. To understand how different training stages impact GIA, we begin by analyzing the properties of $\Phi(\cdot)$.

The smoothness of the objective function influences the difficulty in locating the global optimum, a concept that applies to GIA as well [65]. Specifically, smooth functions are easily optimized and vice versa, they tend to fall into local optima.

Therefore, we propose a novel method termed **IGSA** to characterize the model’s resilience to GIAs:

$$\begin{aligned} IGSA^{\mathbf{X}} &= (\mathbb{E}_{\Delta\mathbf{x}} [\|\Phi(\mathbf{X}) - \Phi(\mathbf{X} + \Delta\mathbf{X}) \cdot \omega\|_2])^{-1} \\ &= K \left(\sum_{k=1}^K \|\mathbf{J}(\mathbf{X}_k) \cdot \omega\|_2 \right)^{-1}, \end{aligned} \quad (10)$$

where $\mathbf{J}(\mathbf{X}) = \frac{\partial\Phi(\mathbf{X})}{\partial\mathbf{X}}$ denotes the Jacobi matrix to compute

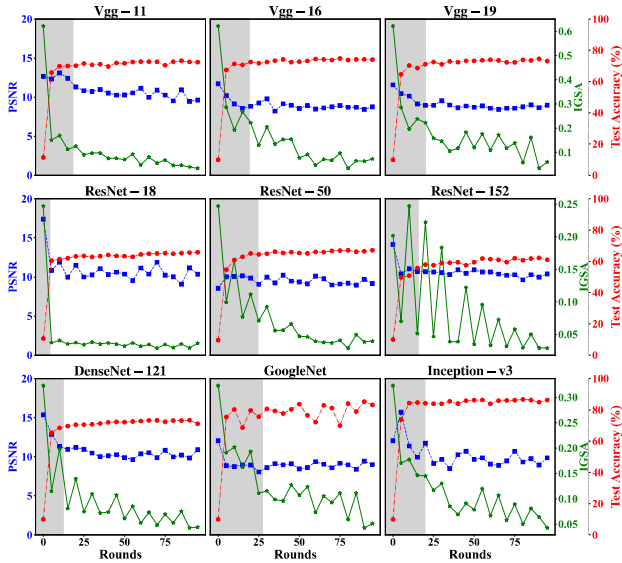


Figure 7: Model’s Resistance to GIA during FL Training. Model’s test accuracy: **Test Accuracy**↑, reconstruction quality: **PSNR**↑, resistance to GIA: **IGSA**↓. Gray areas represent the rounds with high privacy risk.

the first-order derivative of $\Phi(\cdot)$, which reflects **how drastically the gradient changes in response to input perturbations**. To estimate the IGSA values, we utilize K samples located within a radius r around \mathbf{X} . Considering that the gradients span across L layers, we construct a vector $\omega = [1 + 1/L, 1 + 1/(L-1), \dots, 2]$. This vector assigns higher weights to the shallow layers when averaging l_2 -norms. **A higher IGSA value indicates that $\Phi(\cdot)$ possesses a greater degree of smoothness, enabling it easier for the adversary to locate and reconstruct training data.**

5.1.2 Evaluating Model’s Vulnerability to GIA in Different Stages

In this part, we analyze and explain the model’s vulnerability to GIAs during the training process. We choose 9 models from four families that are widely used in vision tasks, i.e., VggNet-(11, 16, 19) [70], ResNet-(18, 50, 152), DenseNet121 [39], and InceptionNet (GoogleNet [72] and Inception-v3 [73]).

Experimental setup: We evaluate nine models using the CIFAR10 dataset. Each model undergoes 100 training rounds. During each round, we launch GIA on the current model while assessing the accuracy and IGSA value. The GIA performance is measured by PSNR (Peak Signal-to-Noise Ratio). To enhance clarity, we average the results every five rounds in Fig. 7. Further details are provided in Appx. B.

Fig. 7 illustrates the resilience of the nine models against the GIA during training. We can intuitively observe that **GIA poses higher risks in the early training stages**. For instance, in the case of the Vgg-11 model, initially, the PSNR value

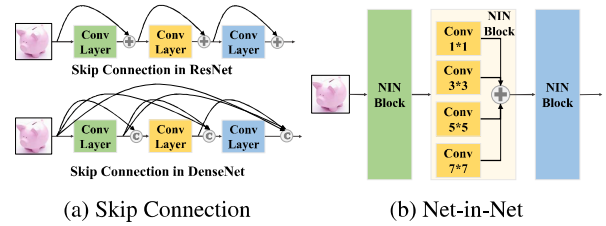


Figure 8: Two Widely Used Model Structures.

is high, but as the training process continues, it consistently stays below 10 after approximately 15 rounds, indicating the reconstructed images cause no privacy leakage. Moreover, as indicated by the test accuracy curves, we note that the model’s resistance correlates well with the fitting degree. The model will be more fragile in the underfitting stages. Additionally, the IGSA curves exhibit distinct phases, where the reconstruction quality is acceptable when IGSA values remain high. Conversely, reconstruction tends to fail when IGSA curves drop and fluctuate at lower levels.

(Insight 5.1) During FL training, early-stage models are more vulnerable to GIA, while late-stage ones have good resistance.

5.2 Model Architecture

The model architecture determines how features are extracted and the information flow, further influencing the path of backpropagation and gradient computation, thereby potentially affecting GIAs. In this subsection, we study the impact of model architecture on the GIA from two perspectives: **structures** and **micro designs**.

5.2.1 Model Structure: A Double-edged Sword

Model structure refers to the way layers are connected to each other. In this subsection, we explore the effect of model structures on the GIA with two widely used cases in practice: **skip connection** [34] (used in models like ResNet and DenseNet families) and **net-in-net** [72] (used in models like GoogleNet and other InceptionNet families).

1. Skip connection is a widely used structure in deep neural networks that helps address gradient vanishing during training [34]. It enables the flow of features from one layer to another by creating direct connections between non-adjacent layers. Generally, there are two common types of skip connections, derived from ResNets and DenseNets, as shown in Fig. 8. In ResNet, skip connections take the form of identity mappings, where the input to a layer is **added directly** to the output of the subsequent layer. In contrast, DenseNet takes a more aggressive approach by densely **concatenating all previous layers within a block**, enhancing feature reuse [39].

To illustrate how skip connections affect the GIA, we start with a derivation on how they affect backpropagation. Assume

a particular layer with \mathbf{I}_l and \mathbf{O}_l represent the input and output, respectively. We consider three types of connections: normal (sequential), ResNet-like, and DenseNet-like. And the process of backpropagation can be presented:

For **normal case**, $\mathbf{O}_l^* = \mathbf{O}_l$:

$$\frac{\partial \ell}{\partial \mathbf{I}_l} = \frac{\partial \ell}{\partial \mathbf{I}_{l+1}} \frac{\partial \mathbf{I}_{l+1}}{\partial \mathbf{I}_l} = \frac{\partial \ell}{\partial \mathbf{I}_{l+1}} \frac{\partial \mathbf{I}_{l+1}}{\partial \mathbf{O}_l} \mathbf{W}_l. \quad (11)$$

For **ResNet** and **DenseNet**, the gradient in the deeper layers is passed to the front layers by adding or concatenation (\oplus) represents both operations in Eq. (12)):

$$\begin{aligned} \frac{\partial \ell}{\partial \mathbf{I}_l} &= \frac{\partial \ell}{\partial \mathbf{I}_{l+1}} \frac{\partial \mathbf{I}_{l+1}}{\partial \mathbf{I}_l} = \frac{\partial \ell}{\partial \mathbf{I}_{l+1}} \frac{\partial \mathbf{I}_{l+1}}{\partial \mathbf{O}_l} \frac{\partial \mathbf{O}_l}{\partial \mathbf{I}_l} \\ &= \frac{\partial \ell}{\partial \mathbf{I}_{l+1}} \frac{\partial \mathbf{I}_{l+1}}{\partial \mathbf{O}_l} \frac{\partial (\mathbf{O}_l^* \oplus \mathbf{I}_l)}{\partial \mathbf{I}_l} = \frac{\partial \ell}{\partial \mathbf{I}_{l+1}} \frac{\partial \mathbf{I}_{l+1}}{\partial \mathbf{O}_l} (\mathbf{W}_l \oplus \mathbf{1}). \end{aligned} \quad (12)$$

Compared with Eq. (11), there is one more **residual term** in the gradient of Eq. (12). For models with skip connections, the residual terms multiply cumulatively during backpropagation, leading to a gradient that incorporates a greater number of combinations. This mechanism helps prevent gradient vanishing and **enhances the performance of GIAs by enabling them to utilize more information**.

We then verify the effect of skip connections on GIA based on ResNets and DenseNets, respectively. First, we evaluate how cutting skip connections at different positions affects the GIA on ResNet-18 and ResNet-34, as shown in Fig. 9. We find that **(1) the presence of skip connections enhances the performance of the GIA**. Fig. 9 demonstrates that the original models are much more vulnerable to the GIA than others with connection cuts. Moreover, cutting the skip connection at any position significantly worsens the effectiveness of the GIA, even resulting in failed reconstruction ($LPIPS > 0.1$). In addition, we find that **(2) skip connections close to shallow layers have a greater impact on the GIA**. Fig. 9b shows that cutting shallow connections (e.g., #0, 1, 2, 4, 5) drastically worsens the performance of the GIA. Recent works [69] regarding information flow in deep neural networks state that shallow layers are more sensitive to the input, and so are their gradients, which benefits GIAs.

Besides, we explore the impact of the number of skip connections on the GIA with DenseNets. We select DenseNet-43 and DenseNet-53 as baselines and obtain two variants for each by employing different cutting strategies [43]. As shown in Fig. 10, **reducing the number of skip connections greatly affects the performance of GIAs**. Images that can be easily inverted in *baselines* are largely unrecognizable in *variants-1*. Moreover, GIAs are completely unable to invert any information from *variants-2*. Reducing the number of skip connections in a model decreases both the backpropagation paths and the residual terms. This leads to the gradients becoming less informative, which in turn limits the effectiveness of GIAs.

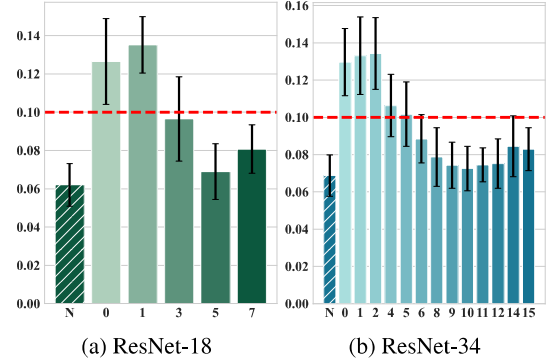


Figure 9: Position of Skip Connections Affects GIA. **N (No Cut)** represents the original model, and thereafter **Id** represents the model cutting the **Id** connection, with darker colors representing deeper positions. $LPIPS \downarrow$.

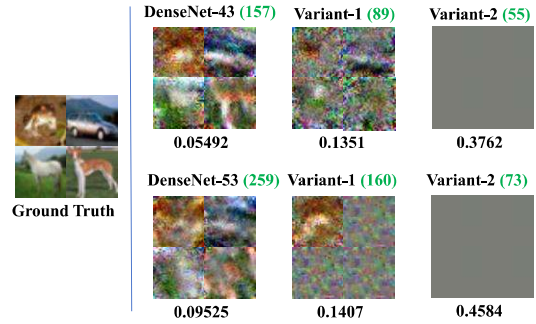


Figure 10: Number of Skip Connections Affects GIA. **Green number** represents the number of skip connections. $LPIPS \downarrow$.

(Insight 5.2.1) (1) Skip connections alleviate gradient vanishing (**Pros**), while increasing the backpropagation paths and introducing residual terms, providing the adversary more information from gradients (**Cons**).

II. Net-in-net (NIN), as proposed in [72], is a module that integrates multi-scale convolutional kernels within a single block, as illustrated in Fig. 8b. In other words, NIN works like a widening layer, where more kernels capture richer input features. However, wider layers and multi-scale kernels also yield more informative gradients through backpropagation, consequently enhancing the effectiveness of GIAs.

To further demonstrate the impact of NIN, we conduct ablation studies on GoogleNet and InceptionNet-V3, as detailed in Tab. 3. Specifically, we compare the reconstruction results using the full gradients against those using only the gradients from a single NIN block. These findings validate that **the gradients from NIN blocks are crucial to the model's vulnerability to the GIA**. For example, as shown in Tab. 3, although only the gradient of the first NIN block in GoogleNet, constituting merely 2.27% of the total parameters, was used, we unexpectedly achieved a result better than that obtained with the full gradient ($0.0879 < 0.2047$). This outcome occurs because approximating the full gradient is computationally in-

Table 3: The Effectiveness of GIA Using Full or a Single NIN Block’s gradient.

Model	Metric	The ID of NIN Blocks Providing Gradient				
		Full	#1	#2	#3	#4
GoogleNet	Params Ratio	100.00%	2.72%	11.68%	6.58%	9.25%
	LPIPS \downarrow	0.2047	0.0879	0.0776	0.0859	0.0756
Inception V3	Params Ratio	100.00%	1.14%	1.56%	1.76%	5.26%
	LPIPS \downarrow	0.2672	0.0858	0.0897	0.0875	0.0804

Table 4: The Effectiveness of GIA against Model Modifications. (+): Modifications improve the reconstruction quality, while others diminish it (-). (R)emove, (I)ncrease, (D)ecrease.

Mods	None	R ReLU	R Dropout	R MaxPool2d	I kernel to 4
LPIPS \downarrow	0.0044(+)	0.0011(+)	0.0043(+)	0.0000(+)	0.0012(+)
Mods	R bias	D kernel to 2D	I kernel to 1I	I padding to 2I	I padding to 3
LPIPS \downarrow	0.0327(-)	0.0299(-)	0.1507(-)	0.0071(-)	0.0506(-)

tensive for an adversary and does not necessarily yield better reconstruction. In contrast, gradients of small, critical components, such as NIN blocks, are more likely to be approximated and leak inputs. **More broadly, auditing the risk of gradient leakage should focus more on identifying vulnerable structures within the gradients.**

(Insight 5.2.1) (2) NIN utilizes multi-scale kernels for stronger feature extraction capability (**Pros**), while requiring different updates to multiple kernels, which provides informative gradients to benefit GIAs (**Cons**).

5.2.2 Micro Design: Tiny Clue Reveals General Trend

Micro designs are subtle techniques ubiquitous in nearly all modern models. To investigate their impact on the GIA, we evaluate six prevalent micro designs: *bias*, *activation function (ReLU)*, *dropout*, *max pooling*, *convolutional kernels (size)*, and *padding*. In particular, we make a series of modifications to a configurable model, ConvNet [26], which only includes components related to micro designs ensuring a clean setting. The standard ConvNet incorporates bias, employs ReLU functions, and includes two max pooling layers and one dropout layer. Additionally, all convolutional layers are equipped with 3×3 kernels and padding 1. Details are provided in Appx. C.

Our findings indicate that **micro designs significantly impact the model’s resistance to the GIA**. As shown in Tab. 4, removing *ReLU*, dropout, *max pooling* layers, or increasing the *kernel size* substantially exacerbates the model’s vulnerability to GIAs. In contrast, removing *bias*, decreasing the *kernel size*, or expanding *padding* enhances the resilience.

Essentially, micro design affects the amount of information available in the feature map. Specifically, **reducing the information related to the input in feature maps would render GIAs less effective**. (1) Feature map sparsification. The *activation function* and *padding* transform or zero out

elements in the feature map. (2) Feature map aggregation. The *max pooling* layer selects representative elements, and a smaller *kernel size* focuses on more localized features. These designs reduce the correlation between the input and the feature maps, and thus affect the accurate inversion of input data from gradients. However, **enhancing the available information contained in the feature map benefits GIAs**. Increasing *kernel size* would promote the model in extracting features on a broader scale, thereby containing more information, while *bias* provides extra parameters for the adversary.

(Insight 5.2.2) Micro designs influence the available information about the input in feature maps, consequently affecting the gradient and the effectiveness of GIAs.

6 Evaluation on Post-Processing

In practical FL systems, clients often apply post-processing techniques to gradients before sharing them. These methods obfuscate the shared gradients to offer potential defense for clients against GIAs. In this section, we study the effectiveness of four commonly utilized post-processing techniques in defending against GIAs under a practical FL setting. Moreover, we evaluate their capacity to address the critical trade-off between the model utility and defensive performance.

Experimental setup: We consider a practical FL system that involves 100 clients collaboratively training ResNet-18 and Swin [56] models on CIFAR10 and CIFAR100 datasets. The server launches three SOTA GIAs: **GIA-O**, **GIA-L**, and **ROG** [89], a recent GIA that breaks through gradient obfuscation. To defend against these attacks, the client employs four post-processing techniques on the gradient: quantization (**Q**) [1], sparsification (**S**) [19], clipping (**C**) [61], and perturbation (**P**) [63]. Details are provided in Appx. B.

We conduct experiments to assess the effectiveness of four post-processing techniques against GIAs and their impact on accuracy under different parameter settings. We choose the optimal performance for each post-processing technique, representing the best privacy-utility trade-off in Fig. 11. The results of ROG are presented in Tab. 6, Appx. C. We show that: **most post-processing techniques can effectively defend against the strongest GIAs without significantly compromising accuracy**. This is illustrated by their points distributed above the blue line (accuracy degradation of no more than 30%) and to the right of the red line. Among these techniques, quantization demonstrates the most favorable trade-off. In the experiment involving ResNet-18 and CIFAR100 (Fig. 11), utilizing 2-bit quantization on shared gradients enables clients to defend against GIAs with an accuracy drop of no more than 5%. However, clipping fails to guarantee the trade-off. As shown in Fig. 11, even with a large clip value that significantly impacts accuracy, it has almost no effect on GIAs. Overall, the post-processing methods demonstrate effective defense capabilities against GIAs while avoiding the signifi-

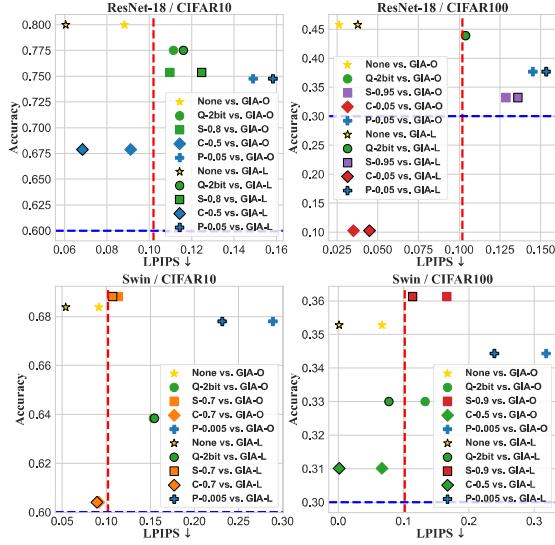


Figure 11: Best Results That Maintain Privacy-Utility Trade-off in FL Systems Utilizing Various Post-Processings. **Red Line** is a split line of privacy leakage. **Blue Line** is a split line of model’s acceptable utility.

cant performance degradation or computational and communication overheads that may be associated with traditional defense mechanisms such as secure multi-party computation (discussed in the Appx. C).

(Insight 6) In practical FL systems, even trivial post-processings applied to gradients can easily defend against the most powerful GIAs while maintaining model accuracy.

7 Discussion

In this section, we summarize our systematization and evaluation, and provide recommendations for the future exploration and mitigation of gradient leakage risks in FL systems.

Explore the risks of GIAs in practical FL systems. Although previous research has demonstrated that GIAs pose significant threats, these notable effects rely on assumptions that the attacks are either overpowered or not fully integrated with the FL system. Our evaluations indicate that GIAs are limited and fragile in practice, suggesting several promising directions for future research: **(1) Investigating the update leakage beyond the gradient.** Future studies should focus on more practical local training settings (e.g., mini-batch updates), exploring the reconstruction of local data based on parameter updates instead of raw gradients. **(2) Assessing GIA risks across practical scenarios.** Most existing GIAs are focused on image classification tasks. However, the risks associated with other practical tasks—such as federated instruction fine-tuning [91]—as well as diverse modalities and advanced architectures (e.g., large language and vision models) remain

undereexplored. **(3) Developing robust GIAs.** In real-world applications, shared gradients are often post-processed or defended. Future research could explore reconstructing local data with inaccurate gradients.

Mitigate the risks of GIA through inherent and lightweight mechanisms. By evaluating three key aspects, we demonstrate that GIA’s effectiveness can be easily compromised. Consequently, we advocate for defenses against GIA that leverage the inherent properties of the FL system or utilize lightweight mechanisms, rather than utilizing subtle and complex designs. **(1) Exploring the natural defensive potential of practical FL systems.** Future applications could focus on designing FL systems that are inherently resistant to GIAs. For example, secure local training settings can be configured by increasing the number of local update rounds or augmenting the client training data, such as by injecting public samples. **(2) Developing fine-grained defense mechanisms.** To prevent over-defense from compromising utility, future research could identify vulnerable phases within the FL training process or pinpoint key structures (e.g. skip connections) within the model for targeted defense. **(3) Enhancing post-processing techniques.** Given that even basic post-processing techniques demonstrate substantial potential in defending against GIAs, future efforts could focus on further improving their effectiveness in balancing the privacy-utility tradeoffs, e.g., designing adaptive post-processing methods.

Gradient Leakage on language models. While existing GIAs primarily focus on image classification tasks, investigating the reconstruction of textual data on language models is crucial for advancing future research, particularly in the era of large language models (LLMs) and multimodal systems. In this context, we provide some concerns and outlooks on this topic: **(1) Metrics for privacy leakage in textual data:** Unlike image data, textual data lacks robust metrics to quantify privacy leakage for reconstruction attacks. For a reconstructed text, “look-alike” does not necessarily indicate privacy leakage if the semantic meaning or critical information is altered. For example, reconstructing “The cat is on the mat” as “The cat is on the hat” introduces minimal character changes but significantly alters the meaning, leading to a little privacy leakage. Therefore, developing fair metrics is essential for auditing GIAs on textual data. **(2) Emphasis on text generation tasks:** Existing GIAs tend to focus on text classification tasks [2, 15, 30], yet language models in practical FL systems are mainly trained for generation tasks (e.g., Q&A). These tasks differ fundamentally in gradient computation: while classification tasks calculate gradients after processing the entire input, generation tasks compute and accumulate gradients word-by-word in an auto-regressive manner. This distinction introduces unique challenges for applying GIAs to text generation tasks. **(3) Challenges with large language models (LLMs):** Recent studies have highlighted that LLMs, pre-trained or fine-tuned in FL settings, are potentially vulnerable to gradient leakage [87]. However,

implementing GIAs on LLMs presents significant obstacles. First, LLMs are trained with large-batch, high-dimensional datasets—for instance, GPT-3-175B training involves batch sizes of up to 3.2 million tokens [7]—which surpass current GIA capabilities. Second, the complexity of LLM architectures and their vast parameter counts complicate gradient inversion. Finally, communication constraints often result in clients uploading only partial gradients during training, frequently employing proxy models or parameter-efficient fine-tuning (PEFT) techniques [21]. This severely restricts the information available to adversaries.

8 Conclusion

In this work, we conducted a comprehensive study on GIAs in practical FL systems. We thoroughly reviewed the evolution of the GIA, highlighting the key milestones and breakthroughs. Additionally, we established a systematization of GIAs to reveal their inherent threats. We indicated that the notable effectiveness demonstrated by current GIAs relies on ideal settings with auxiliary assumptions. To evaluate the actual threat of GIAs against practical FL systems, we identified three fundamental aspects influencing GIAs’s effectiveness: *training setup*, *model*, and *post-processing*. Through theoretical and empirical evaluations of SOTA GIAs in diverse settings, our findings indicate that GIAs are *constrained*, *fragile*, and *easily defensible* in practice. The actual threats posed by GIAs to practical FL systems are limited, despite their perceived potency in previous literature. We hope our work corrects some misconceptions and promotes more precise and realistic investigations into GIAs within FL systems.

Acknowledgments

This work is supported by National Natural Science Foundation of China (Grant No. U24B20182, 62122066, 62472158, 62102337), National Key R&D Program of China (Grant No. 2021ZD0112803), Key R&D Program of Zhejiang (Grant No. 2024C01164, 2022C01018), Young Elite Scientists Sponsorship Program by CAST (Grant No. 2023QNRC001), and Natural Science Foundation of Hunan Province, China (Grant No. 2023JJ40174).

Ethical Considerations

The authors have reviewed the ethical considerations outlined in the conference documents, including the *Call for Papers*, *Submission Policies and Instructions*, and *Ethics Guidelines*. We confirm that all datasets and models utilized in our experiments are open-source, publicly available, and non-sensitive. Our evaluations did not involve the disclosure of harmful or sensitive content to the public or other researchers. Moreover,

our study did not introduce any new risks; instead, it highlighted that existing GIAs are constrained, fragile, and easily defensible in FL systems, thereby contributing to a better understanding of gradient leakage.

Compliance with the Open Science Policy

In full compliance with the Open Science Policy, we commit to sharing all research artifacts associated with this study, including datasets, scripts, and source code. These resources are publicly available via a Zenodo repository: <https://zenodo.org/records/14664682>.

References

- [1] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. Communication-efficient sgd via gradient quantization and encoding. *NeurIPS*, 2017.
- [2] Mislav Balunovic, Dimitar Dimitrov, Nikola Jovanović, and Martin Vechev. Lamp: Extracting text from gradients with language model priors. *NeurIPS*, 2022.
- [3] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. Compressed optimisation for non-convex problems. In *ICML*, 2018.
- [4] Franziska Boenisch, Adam Dziedzic, Roei Schuster, Ali Shahin Shamsabadi, Ilia Shumailov, and Nicolas Papernot. When the curious abandon honesty: Federated learning is not private. In *IEEE EuroS&P*, 2023.
- [5] 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 *ACM CCS*, 2017.
- [6] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv:1809.11096*, 2018.
- [7] Tom B Brown. Language models are few-shot learners. *arXiv:2005.14165*, 2020.
- [8] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. Membership inference attacks from first principles. In *IEEE S&P*, 2022.
- [9] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv:2003.04297*, 2020.
- [10] Kewei Cheng, Tao Fan, Yilun Jin, Yang Liu, Tianjian Chen, Dimitrios Papadopoulos, and Qiang Yang. Secureboost: A lossless federated learning framework. *IEEE Intelligent Systems*, 2021.

- [11] Hong-Min Chu, Jonas Geiping, Liam H Fowl, Micah Goldblum, and Tom Goldstein. Panning for gold in federated learning: Targeted text extraction under arbitrarily large-scale aggregation. In *ICLR*, 2022.
- [12] Mircea Cimpoi, Subhansu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *CVPR*, 2014.
- [13] Trung Dang, Om Thakkar, Swaroop Ramaswamy, Rajiv Mathews, Peter Chin, and Françoise Beaufays. Revealing and protecting labels in distributed training. *NeurIPS*, 2021.
- [14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.
- [15] Jieren Deng, Yijue Wang, Ji Li, Chao Shang, Hang Liu, Sanguthevar Rajasekaran, and Caiwen Ding. Tag: Gradient attack on transformer-based language models. *arXiv:2103.06819*, 2021.
- [16] Dimitar I Dimitrov, Maximilian Baader, Mark Niklas Müller, and Martin Vechev. Spear: Exact gradient inversion of batches in federated learning. *arXiv:2403.03945*, 2024.
- [17] Dimitar Iliev Dimitrov, Mislav Balunovic, Nikola Konstantinov, and Martin Vechev. Data leakage in federated averaging. *TMLR*, 2022.
- [18] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv:2010.11929*, 2020.
- [19] Negar Foroutan Eghlidi and Martin Jaggi. Sparse communication for training deep networks. *arXiv:2009.09271*, 2020.
- [20] 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. *Federated Learning: Privacy and Incentive*, pages 32–50, 2020.
- [21] Tao Fan, Yan Kang, Guoqiang Ma, Weijing Chen, Wenbin Wei, Lixin Fan, and Qiang Yang. Fate-llm: A industrial grade federated learning framework for large language models. *arXiv:2310.10049*, 2023.
- [22] Hao Fang, Bin Chen, Xuan Wang, Zhi Wang, and Shu-Tao Xia. Gifd: A generative gradient inversion method with feature domain optimization. In *ICCV*, 2023.
- [23] Liam Fowl, Jonas Geiping, Wojtek Czaja, Micah Goldblum, and Tom Goldstein. Robbing the fed: Directly obtaining private data in federated learning with modified models. *arXiv:2110.13057*, 2021.
- [24] Liam Fowl, Jonas Geiping, Steven Reich, Yuxin Wen, Wojtek Czaja, Micah Goldblum, and Tom Goldstein. Decepticons: Corrupted transformers breach privacy in federated learning for language models. *arXiv:2201.12675*, 2022.
- [25] Kostadin Garov, Dimitar I Dimitrov, Nikola Jovanović, and Martin Vechev. Hiding in plain sight: Disguising data stealing attacks in federated learning. *arXiv:2306.03013*, 2023.
- [26] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. How easy is it to break privacy in federated learning? *NeurIPS*, 2020.
- [27] Jiahui Geng, Yongli Mou, Feifei Li, Qing Li, Oya Beyan, Stefan Decker, and Chunming Rong. Towards general deep leakage in federated learning. *arXiv:2110.09074*, 2021.
- [28] Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv:1712.07557*, 2017.
- [29] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *NeurIPS*, 2014.
- [30] Samyak Gupta, Yangsibo Huang, Zexuan Zhong, Tianyu Gao, Kai Li, and Danqi Chen. Recovering private text in federated learning of language models. *NeurIPS*, 2022.
- [31] 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:1811.03604*, 2018.
- [32] Ali Hatamizadeh, Hongxu Yin, Pavlo Molchanov, Andriy Myronenko, Wenqi Li, Prerna Dogra, Andrew Feng, Mona G Flores, Jan Kautz, Daguang Xu, et al. Do gradient inversion attacks make federated learning unsafe? *IEEE Trans. on Medical Imaging*, 2023.
- [33] Ali Hatamizadeh, Hongxu Yin, Holger R Roth, Wenqi Li, Jan Kautz, Daguang Xu, and Pavlo Molchanov. Gradvit: Gradient inversion of vision transformers. In *CVPR*, 2022.
- [34] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

- [35] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *CVPR*, 2021.
- [36] Jiahui Hu, Jiacheng Du, Zhibo Wang, Xiaoyi Pang, Yajie Zhou, Peng Sun, and Kui Ren. Does differential privacy really protect federated learning from gradient leakage attacks? *IEEE TMC*, 2024.
- [37] Jiahui Hu, Zhibo Wang, Yongsheng Shen, Bohan Lin, Peng Sun, Xiaoyi Pang, Jian Liu, and Kui Ren. Shield against gradient leakage attacks: Adaptive privacy-preserving federated learning. *IEEE/ACM TON*, 2023.
- [38] Rui Hu, Yuanxiong Guo, and Yanmin Gong. Federated learning with sparsified model perturbation: Improving accuracy under client-level differential privacy. *IEEE TMC*, 2023.
- [39] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017.
- [40] Rui Huang and Yixuan Li. Mos: Towards scaling out-of-distribution detection for large semantic space. In *CVPR*, 2021.
- [41] Yangsibo Huang, Samyak Gupta, Zhao Song, Kai Li, and Sanjeev Arora. Evaluating gradient inversion attacks and defenses in federated learning. *NeurIPS*, 2021.
- [42] Jinwoo Jeon, Kangwook Lee, Sewoong Oh, Jungseul Ok, et al. Gradient inversion with generative image prior. *NeurIPS*, 2021.
- [43] Rui-Yang Ju, Jen-Shiun Chiang, Chih-Chia Chen, and Yu-Shian Lin. Connection reduction of densenet for image recognition. In *IEEE ISPACS*, 2022.
- [44] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019.
- [45] Jakub Konečný, Brendan McMahan, and Daniel Ramage. Federated optimization: Distributed optimization beyond the datacenter. *arXiv:1511.03575*, 2015.
- [46] Maximilian Lam, Gu-Yeon Wei, David Brooks, Vijay Janapa Reddi, and Michael Mitzenmacher. Breaking privacy in federated learning by reconstructing the user participant matrix. In *ICML*, 2021.
- [47] Bowen Li, Hanlin Gu, Ruoxin Chen, Jie Li, Chentao Wu, Na Ruan, Xueming Si, and Lixin Fan. Temporal gradient inversion attacks with robust optimization. *arXiv:2306.07883*, 2023.
- [48] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *ICCV*, 2017.
- [49] Jiachun Li, Yan Meng, Lichuan Ma, Suguo Du, Haojin Zhu, Qingqi Pei, and Xuemin Shen. A federated learning based privacy-preserving smart healthcare system. *IEEE Trans. on Industrial Informatics*, 2021.
- [50] Oscar Li, Jiankai Sun, Xin Yang, Weihao Gao, Hongyi Zhang, Junyuan Xie, Virginia Smith, and Chong Wang. Label leakage and protection in two-party split learning. *arXiv:2102.08504*, 2021.
- [51] Xiaoxiao Li, Meirui Jiang, Xiaofei Zhang, Michael Kamp, and Qi Dou. Fedbn: Federated learning on non-iid features via local batch normalization. *arXiv:2102.07623*, 2021.
- [52] Zhuohang Li, Jiabin Zhang, Luyang Liu, and Jian Liu. Auditing privacy defenses in federated learning via generative gradient leakage. In *CVPR*, 2022.
- [53] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and William J Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. *arXiv:1712.01887*, 2017.
- [54] Xuan Liu, Siqi Cai, Qihua Zhou, Song Guo, Ruibin Li, and Kaiwei Lin. Gradient diffusion: A perturbation-resilient gradient leakage attack. *arXiv:2407.05285*, 2024.
- [55] Yahui Liu, Bin Ren, Yue Song, Wei Bi, Nicu Sebe, and Wei Wang. Breaking the chain of gradient leakage in vision transformers. *arXiv:2205.12551*, 2022.
- [56] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.
- [57] Guodong Long, Yue Tan, Jing Jiang, and Chengqi Zhang. Federated learning for open banking. In *Federated Learning: Privacy and Incentive*. Springer, 2020.
- [58] Jiahao Lu, Xi Sheryl Zhang, Tianli Zhao, Xiangyu He, and Jian Cheng. April: Finding the achilles' heel on privacy for vision transformers. In *CVPR*, 2022.
- [59] Kailang Ma, Yu Sun, Jian Cui, Dawei Li, Zhenyu Guan, and Jianwei Liu. Instance-wise batch label restoration via gradients in federated learning. In *ICLR*, 2022.
- [60] 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*, 2017.

- [61] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. *arXiv:1710.06963*, 2017.
- [62] Payman Mohassel and Yupeng Zhang. Secureml: A system for scalable privacy-preserving machine learning. In *IEEE S&P*, 2017.
- [63] Mohammad Naseri, Jamie Hayes, and Emiliano De Cristofaro. Local and central differential privacy for robustness and privacy in federated learning. *arXiv:2009.03561*, 2020.
- [64] Dario Pasquini, Danilo Francati, and Giuseppe Ate-niese. Eluding secure aggregation in federated learning via model inconsistency. In *ACM CCS*, 2022.
- [65] Guannan Qu and Na Li. Harnessing smoothness to accelerate distributed optimization. *IEEE Trans. on Control of Network Systems*, 2017.
- [66] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv:1511.06434*, 2015.
- [67] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Non-linear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 1992.
- [68] Daniel Scheliga, Patrick Mäder, and Marco Seeland. Precode-a generic model extension to prevent deep gradient leakage. In *WACV*, 2022.
- [69] Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv:1703.00810*, 2017.
- [70] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014.
- [71] Jingwei Sun, Ang Li, Binghui Wang, Huanrui Yang, Hai Li, and Yiran Chen. Soteria: Provable defense against privacy leakage in federated learning from representation perspective. In *CVPR*, 2021.
- [72] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.
- [73] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.
- [74] Mark Vero, Mislav Balunović, Dimitar I Dimitrov, and Martin Vechev. Data leakage in tabular federated learning. *arXiv:2210.01785*, 2022.
- [75] Aidmar Wainakh, Fabrizio Ventola, Till Müßig, Jens Keim, Carlos Garcia Cordero, Ephraim Zimmer, Tim Grube, Kristian Kersting, and Max Mühlhäuser. User label leakage from gradients in federated learning. *arXiv:2105.09369*, 2021.
- [76] Fei Wang, Ethan Hugh, and Baochun Li. More than enough is too much: Adaptive defenses against gradient leakage in production federated learning. *IEEE/ACM ToN*, 2024.
- [77] Junxiao Wang, Song Guo, Xin Xie, and Heng Qi. Protect privacy from gradient leakage attack in federated learning. In *IEEE INFOCOM*, 2022.
- [78] Yanbo Wang, Jian Liang, and Ran He. Towards eliminating hard label constraints in gradient inversion attacks. *arXiv:2402.03124*, 2024.
- [79] Zhibo Wang, Zhiwei Chang, Jiahui Hu, Xiaoyi Pang, Jiacheng Du, Yongle Chen, and Kui Ren. Breaking secure aggregation: Label leakage from aggregated gradients in federated learning. In *INFOCOM*, 2024.
- [80] 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.
- [81] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE TIFS*, 2020.
- [82] Yuxin Wen, Jonas Geiping, Liam Fowl, Micah Goldblum, and Tom Goldstein. Fishing for user data in large-batch federated learning via gradient magnification. *arXiv:2202.00580*, 2022.
- [83] Yuxin Wu and Kaiming He. Group normalization. In *ECCV*, 2018.
- [84] Gaojian Xiong, Xianxun Yao, Kailang Ma, Jian Cui, et al. Gi-pip: Do we require impractical auxiliary dataset for gradient inversion attacks? *arXiv:2401.11748*, 2024.
- [85] Jin Xu, Chi Hong, Jiyue Huang, Lydia Y Chen, and Jérémie Decouchant. Agic: Approximate gradient inversion attack on federated learning. In *SRDS*, 2022.
- [86] Haomiao Yang, Mengyu Ge, Dongyun Xue, Kunlan Xiang, Hongwei Li, and Rongxing Lu. Gradient leakage attacks in federated learning: Research frontiers, taxonomy and future directions. *IEEE Network*, 2023.

- [87] Rui Ye, Wenhao Wang, Jingyi Chai, Dihan Li, Zexi Li, Yinda Xu, Yaxin Du, Yanfeng Wang, and Siheng Chen. Openfedllm: Training large language models on decentralized private data via federated learning. *arXiv:2402.06954*, 2024.
- [88] Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M Alvarez, Jan Kautz, and Pavlo Molchanov. Image batch recovery via gradinversion. In *CVPR*, 2021.
- [89] Kai Yue, Richeng Jin, Chau-Wai Wong, Dror Baron, and Huaiyu Dai. Gradient obfuscation gives a false sense of security in federated learning. In *USENIX Security*, pages 6381–6398, 2023.
- [90] Chengliang Zhang, Suyi Li, Junzhe Xia, Wei Wang, Feng Yan, and Yang Liu. {BatchCrypt}: Efficient homomorphic encryption for {Cross-Silo} federated learning. In *USENIX ATC*, 2020.
- [91] Jianyi Zhang, Saeed Vahidian, Martin Kuo, Chunyuan Li, Ruiyi Zhang, Tong Yu, Guoyin Wang, and Yiran Chen. Towards building the federatedgpt: Federated instruction tuning. In *IEEE ICASSP*, 2024.
- [92] Junpeng Zhang, Hui Zhu, Fengwei Wang, Jiaqi Zhao, Qi Xu, and Hui Li. Security and privacy threats to federated learning: Issues, methods, and challenges. *Security and Communication Networks*, 2022.
- [93] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.
- [94] Rui Zhang, Song Guo, Junxiao Wang, Xin Xie, and Dacheng Tao. A survey on gradient inversion: Attacks, defenses and future directions. *arXiv:2206.07284*, 2022.
- [95] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. idlg: Improved deep leakage from gradients. *arXiv:2001.02610*, 2020.
- [96] 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 *CVPR*, 2023.
- [97] 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 *S&P*, pages 30–30. IEEE, 2023.
- [98] Zihao Zhao, Yuzhu Mao, Yang Liu, Linqi Song, Ye Ouyang, Xinlei Chen, and Wenbo Ding. Towards efficient communications in federated learning: A contemporary survey. *Journal of Franklin Institute*, 2023.

- [99] Junyi Zhu and Matthew Blaschko. R-gap: Recursive gradient attack on privacy. *arXiv:2010.07733*, 2020.
- [100] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. *NeurIPS*, 2019.

Appendices

A Proofs

I. Proof for Lemma. 4.1: Based on Eq. (3), (4) and (5), we can derive the gradients of every layer for the following iterative form:

$$\begin{aligned}\frac{\partial \ell}{\partial \mathbf{W}_L} &= y \frac{\partial \ell}{\partial \mu} \mathcal{F}_{L-1}^\top, \\ \frac{\partial \ell}{\partial \mathbf{W}_{L-1}} &= \left(\mathbf{W}_L^\top \left(y \frac{\partial \ell}{\partial \mu} \right) \right) \odot \sigma'_{L-1} \mathcal{F}_{L-2}^\top, \\ \frac{\partial \ell}{\partial \mathbf{W}_{L-2}} &= \left(\mathbf{W}_{L-1}^\top \left(\left(\mathbf{W}_L^\top \left(y \frac{\partial \ell}{\partial \mu} \right) \right) \odot \sigma'_{L-1} \right) \right) \\ &\quad \odot \sigma'_{L-2} \mathbf{P}^\top.\end{aligned}\quad (13)$$

Particularly, from Eq. (3) and (13), we can characterize the dependence between the gradient of the last layer $\frac{\partial \ell}{\partial \mathbf{W}_L}$ and μ as follows [99]:

$$\frac{\partial \ell}{\partial \mathbf{W}_L} \mathbf{W}_L = y \frac{\partial \ell}{\partial \mu} \mathcal{F}_{L-1}^\top \mathbf{W}_L = \frac{\partial \ell}{\partial \mu} \mu = \frac{-\mu}{1 + e^\mu}. \quad (14)$$

With μ , the input \mathbf{x} can be determined by iteratively solving Eq. (3), as we derived in Eq. (7).

II. Proof for Theorem. 4.2: Based on Eq. (8), the error between the ground truth \mathbf{x}^* and the reconstructed data \mathbf{x}^{rec} is determined by the outputs:

$$\mathcal{G}(\mu^*) = \frac{-\mu^*}{1 + e^{\mu^*}} = \frac{\partial \ell}{\partial \mathbf{W}_L^0} \mathbf{W}_L^0, \quad (15)$$

$$\mathcal{G}(\mu^{rec}) = \frac{-\mu^{rec}}{1 + e^{\mu^{rec}}} = \left(\frac{\partial \ell}{\partial \mathbf{W}_L^0} + \sum_{u=1}^{U-1} \frac{\partial \ell}{\partial \mathbf{W}_L^u} \right) \mathbf{W}_L^0. \quad (16)$$

The output difference can be approximated by:

$$\mu^{rec} - \mu^* \approx \frac{\mathcal{G}(\mu^{rec}) - \mathcal{G}(\mu^*)}{\mathcal{G}'(\mu^*)} = \frac{(e^{\mu^*} + 1)^2 \sum_{u=1}^{U-1} \frac{\partial \ell}{\partial \mathbf{W}_L^u} \mathbf{W}_L^0}{\mu^* e^{\mu^*} - e^{\mu^*} - 1}. \quad (17)$$

III. Proof for Theorem. 4.3: By the Lagrange Mean Value Theorem, for a given mask Δ , we can find a ξ such that:

$$\Phi(\mathbf{x} + \Delta) - \Phi(\mathbf{x}) = \Phi'(\xi) \Delta. \quad (18)$$

Assume that Δ and \mathbf{x} are linearly related and that $\mathbf{x} + \Delta$ is neighboring \mathbf{x} :

$$\Phi(\mathbf{x} + \Delta) - \Phi(\mathbf{x}) = \Phi'(\xi) \mathbf{e} \mathbf{x} \approx \Phi'(\mathbf{x}) \mathbf{e} \mathbf{x}, \quad (19)$$

where ε is the scale factor. $\Phi(\mathbf{x} + \Delta) - \Phi(\mathbf{x}) = \mathbf{0}$ if and only if $\Phi'(\mathbf{x})\mathbf{x} = \mathbf{0}$, which indicates $\Phi'(\mathbf{x})$ is not full rank. Considering $\Phi'(\cdot) \in \mathbb{R}^{p \times B \times D}$, it follows that there must exist at least one nonzero solution when $p < B \times D$.

B Experimental Setups

I. Details for experiments in Sec. 4: We conduct evaluations on CIFAR10 (at resolutions of 32×32 and 64×64) and ImageNet-1K (at resolutions of 128×128 , 256×256 , and 512×512) datasets, employing $B = 1$ for Fig. 4a. Subsequently, for Fig. 4b, we assess GIAs using the CIFAR100 dataset with a fixed resolution of 32×32 . In GIA-O, the ratios for priors are specified as $1e^{-5}$. For GIA-L, we adhere to Jeon’s approach [42] involving two-stage optimization, comprising DcGAN pretraining for CIFAR10/100 and BigGAN for ImageNet-1K. We opt for the Adam optimizer (8000 iterations for GIA-O, 400 iterations for latent space optimization and 6000 iterations for optimizing the generator in GIA-L). The evaluation of GIAs is performed on the ResNet-18 model [34] initialized with PyTorch’s default initialization.

II. Details for experiments in Sec. 5: We adopt Geiping’s GIA [26] as the baseline, incorporating solely gradient matching and p_{TV} as loss function. Nine models are trained on the CIFAR10 dataset with a batch size of 128 and a learning rate of 0.01 for 10 local epochs. The whole FL process involves 5 clients for 100 rounds totally. For each model, we calculated its IGSA values by averaging the reconstruction results of 10 independent samples, with parameters $K = 10000$ and $r = 1e^{-3}$, and subsequently normalized for each model.

III. Details for experiments in Sec. 6: We adopt a practical FedAvg setup where, in each round, 10% of clients are randomly chosen for training, with $E = 1$ and $B = 10$. To optimize GIA’s performance, our client pool comprises 99 normal clients ($N = 500$) and 1 victim client ($N = 10$).

C Supplementary Materials

I. Related works: The most closely related works can be categorized into two main groups: (1) Survey on GIAs. Prior surveys [86, 94] primarily focused on a coarse-grained categorization of GIAs with optimization-based and analysis-based, providing their definitions. In this paper, we go a step further by summarizing the evolution of GIAs, marking a key milestone, and offering a detailed systematization. Specifically, we characterize the threat models, categorizing the adversary assumptions so far. (2) Evaluation on GIAs. Previous works [32, 41] have highlighted that the lack of a strong assumption (BN statistic) affects the efficacy of GIAs in practice. However, they have not systematically identified the key challenges that practical FL systems pose to GIAs and given a comprehensive analysis. Yue et al. [89] have pointed out that techniques such as clipping and compression are ineffective

Table 5: Comparison of Different Metrics.

GIA	Metric	Batch size						
		5	10	15	20	50	80	100
O	LPIPS	0.0338	0.0653	0.0789	0.0932	0.1167	0.1217	0.1225
	BRR	67%	34.75%	15.42%	4.06%	0%	0%	0%
	WSRR	95%	98.75%	83.75%	47.50%	0%	0%	0%
L	LPIPS	0.0446	0.0706	0.0875	0.0973	0.1238	0.1295	0.1321
	BRR	48.25%	25.63%	11.67%	5.13%	0.27%	0.11%	0.10%
	WSRR	82.50%	87.50%	63.75%	55%	7.50%	6.25%	8.75%

in defending GIAs, which appears to contradict the conclusions in Sec 6. This discrepancy arises because they did not conduct a thorough evaluation of the defensive performance of these methods in a practical FL setting. Besides, Wang et al. [76] indicate that the effectiveness of GIA in practice is limited. However, their evaluations are restricted to the impact of model initialization and the number of local updates on two earlier GIA-Os with comparatively weaker effectiveness. In contrast, we evaluate several state-of-the-art GIAs from the GIA-O and GIA-L categories, and present extensive evaluations from various aspects, uncovering findings into the limited threats posed by GIAs in practice comprehensively.

II. Discussion on metric selection: In this paper, we primarily use the averaged *LPIPS* metric to evaluate the privacy leakage of the batch of reconstructed images, which is widely adopted in previous works. While Carlini et al. [8] argue that averaged metrics are inadequate for membership inference attacks due to the potential presence of partially vulnerable samples, the averaged metric remains appropriate for GIAs. This is because reconstructed samples within a batch typically exhibit similar reconstruction quality.

To support this, we introduce two sample-level metrics for a more fine-grained assessment of GIA reconstruction outcomes. To ensure rigorous auditing, we combine two similarity metrics to define an indicator function for identifying whether a single reconstructed sample compromises privacy: $\mathbb{1}[\text{PSNR}(x, x') > \tau_1 \wedge \text{LPIPS}(x, x') < \tau_2]$. Based on this, we propose the “Batch Reconstruction Rate” (*BRR*) and the “Worst-Sample Reconstruction Rate” (*WSRR*). *BRR* measures the proportion of samples in a batch that leaks privacy, while *WSRR* quantifies the likelihood of reconstructing the most vulnerable sample. We evaluate the consistency of the averaged *LPIPS*, *BRR*, and *WSRR* in quantifying privacy leakage in GIAs across various batch sizes in Tab. 5. The experiments are conducted using the CIFAR-10 dataset, with τ_1 and τ_2 set to 18 and 0.1, respectively. *WSRR* is calculated using 10 independent attack initializations, and all the results are averaged over 10 random seeds. Our findings demonstrate that the averaged *LPIPS* effectively measures privacy leakage in GIAs. When *LPIPS* is below 0.1, both *BRR* and *WSRR* are high, indicating significant privacy risks. Conversely, when *LPIPS* approaches or exceeds 0.1, *BRR* approaches zero, and even the worst samples cannot be reliably reconstructed.

III. Introduction to post-processings: (1) Quantization refers to transforming gradients to lower precision (e.g., 4-bit, 1-bit) before sharing. We consider the quantization method QSGD ([1, 2, 3, 4]-bit) in [1] and SignSGD [3]. **(2) Sparsification** transforms a full gradient to a sparse one with a subset of significant elements and sets others to zero [98]. We consider $top-k$ sparsification [19], selecting proportion k ([0.6, 0.7, 0.8, 0.9, 0.95]) greatest absolute values as significant elements. **(3) Clipping** ensures the gradient values are all in a predefined bound by clipping extreme values. Here, we consider flat clipping [61] with bound ([0.5, 0.3, 0.1, 0.05, 0.01] for ResNet-18, [1, 0.7, 0.5, 0.3, 0.1] for Swin). **(4) Perturbation** refers to adding noise into the gradients. In FL, clients often locally perturb the gradients before sharing to achieve local-differential-privacy [63]. Here, we add Gaussian noise with multipliers ([0.05, 0.1, 0.15, 0.2, 0.25] for ResNet-18, [0.005, 0.007, 0.01, 0.03, 0.05] for Swin) to the gradient.

IV. Traditional defense mechanisms in FL systems: We provide a discussion on traditional defense mechanisms against GIAs in FL, focusing on their privacy-utility tradeoffs, as well as their computational and communication overhead.

- **Differential Privacy (DP).** DP ensures that individual data samples from local datasets are hard to identify or infer by applying perturbation mechanisms, such as adding Gaussian noise, to the shared gradients [28, 37, 81]. However, achieving an optimal privacy-utility tradeoff when applying DP in FL is challenging [77]. To effectively defend against GIAs, significant perturbations are often required, which can severely degrade FL training accuracy [36]. Additionally, introducing noise at the client slows the training process, necessitating more communication rounds for convergence [38].

- **Secure Multi-Party Computation (SMPC).** SMPC enables collaborative computations among multiple participants without disclosing private inputs to other participants, thereby ensuring privacy preservation [5, 62]. In FL, SMPC-based secure aggregation protocols protect client gradients by employing techniques such as Secret Sharing and pairwise masking. For instance, local gradients are masked through the weighted average of gradient vectors from a random subset, and the random factors cancel out during aggregation by a trusted server [5]. As a result, for honest-but-curious servers without additional side information, SMPC effectively defends against GIAs by revealing only aggregated gradients rather than individual ones. Nevertheless, SMPC introduces significant computational and communication overhead. For example, in Secret Sharing, all shares generated by one client are required to interact with other clients. and this overhead grows exponentially with the number of clients [92].

- **Homomorphic Encryption (HE).** HE allows specific computations, such as addition, to be performed directly on encrypted gradients without requiring decryption [10, 90]. By performing gradient aggregation on ciphertexts, HE ensures that gradients remain inaccessible to external parties, including the server. Unlike DP, HE does not reduce training

accuracy, as no obfuscation is added to the gradients during encryption. However, HE imposes substantial computational and communication overhead due to the complexity of encrypting gradients and transmitting the resulting ciphertexts.

V. Additional results in Sec. 6:

Table 6: The ROG [89] against Post-Processing Techniques.

Model	Dataset	Post-processing			
		Q	S	C	P
ResNet-18	C10	0.1548	0.1823	0.1771	0.229
	C100	0.1168	0.1144	0.0529	0.1645
Swin	C10	0.1680	0.1242	0.1117	0.1900
	C100	0.1820	0.0906	0.045	0.1856

VI. Systematization on gradient inversion attacks.

Table 7: Systematization on Gradient Inversion Attacks.

Publication	Threat mode			Attack			Defence ⁴
	Server's Trust. ¹	Capability	Goal	Assumption ²	Strategy ³	Modality	
Wang et al. (2019) [80]	HBC, M	Active, Passive	data	0, 1, 2	GIA-O	CV	None
Zhu et al. (2019) [100]	HBC	Passive	label, data 0		GIA-O	CV, NLP	Q, S, P
Zhao et al. (2020) [95]	HBC	Passive	label, data 0		GIA-O	CV	None
Geiping et al. (2020) [26]	HBC	Passive	data	0, 1, 3	GIA-O	CV	None
Fan et al. (2020) [20]	HBC	Passive	label, data 0, 1		GIA-A	CV	P
Zhu et al. (2021) [99]	HBC	Passive	data	0, 1	GIA-A	CV	P
Wainakh et al. (2021) [75]	HBC	Passive	label	0	GIA-A	CV	S, P
Geng et al. (2021) [27]	HBC	Passive	label, data 0, 1, 3		GIA-O	CV	None
Deng et al. (2021) [15]	HBC	Passive	data	0, 1	GIA-O	NLP	None
Jeon et al. (2021) [42]	HBC	Passive	data	0, 2	GIA-L	CV	S, P
Yin et al. (2021) [88]	HBC	Passive	label, data 0, 1, 4		GIA-O	CV	None
Dang et al. (2021) [13]	HBC	Passive	label	0	GIA-O, GIA-A	CV, NLP	Q, S
Li et al. (2022) [52]	HBC	Passive	label, data 0, 1, 2		GIA-L	CV	S, P, C
Hatamizadeh et al. (2022) [33]	HBC	Passive	data	0, 1, 2, 4	GIA-O	CV	None
Lu et al. (2022) [58]	HBC	Passive	data	0, 1	GIA-O, GIA-A	CV	P
Balunovic et al. (2022) [12]	HBC	Passive	data	0, 1, 2	GIA-L	NLP	P
Gupta et al. (2022) [30]	HBC	Passive	data	0, 1	GIA-O	NLP	S, P
Dimitrov et al. (2022) [17]	HBC	Passive	label, data 0, 1, 3		GIA-O	CV	None
Xu et al. (2022) [85]	HBC	Passive	label, data 0, 1, 3, 4		GIA-O	CV	None
Ma et al. (2023) [59]	HBC	Passive	label	0	GIA-A	CV	None
Li et al. (2023) [47]	HBC	Passive	label, data 0, 4		GIA-O	CV, NLP	None
Vero et al. (2023) [74]	HBC	Passive	label, data 0		GIA-O, GIA-A	Tabular	None
Lam et al. (2021) [46]	M	Active	data	0, 5	GIA-A	CV	None
Boenisch et al. (2021) [4]	M	Active	data	0, 5	GIA-A	CV	C, P
Fowl et al. (2021) [23]	M	Active	data	0, 5	GIA-A	CV	None
Chu et al. (2022) [11]	M	Active	data	0, 5	GIA-A	NLP	None
Fowl et al. (2022) [24]	M	Active	data	0, 5	GIA-A	NLP	C, P
Pasquini et al. (2022) [64]	M	Active	label, data 0, 5		GIA-O	CV	None
Wen et al. (2022) [82]	M	Active	data	0, 5	GIA-A	CV	Q
Zhao et al. (2023) [96]	M	Active	data	0, 5	GIA-A	CV	None
Fang et al. (2023) [22]	HBC	Passive	data	0, 1, 2	GIA-L	CV	S, C, P
Yue et al. (2023) [89]	HBC	Passive	data	0, 1, 2	GIA-L	CV	Q, S, P
Garov et al. (2024) [25]	M	Active	data, label 0, 2		GIA-L	CV	C, P
Xiong et al. (2024) [84]	HBC	Passive	data	0, 1, 2	GIA-L	CV	None
Liu et al. (2024) [54]	HBC	Passive	data, label 0		GIA-O	CV	P
Wang et al. (2024) [79]	M	Active	label	0, 5	GIA-A	CV	S, P
Dimitrov et al. (2024) [16]	HBC	Passive	data	0	GIA-A	CV	None
Wang et al. (2024) [78]	HBC	Passive	label	0	GIA-A	CV	P

¹ HBC for Honest-but-curious; M for Malicious

² Assumption: [0] Basic information, [1] Priors, [2] Data distribution, [3] Client-side training details, [4] BN statistics, [5] Malicious behavior

³ GIA-O for GIA with Observable Space Optimization; GIA-L for GIA with Latent Space Optimization; GIA-A for Analytic-based GIA

⁴ Quantization (Q), Sparsification (S), Clipping (C), and Perturbation (P)