GIDM
Gradient Inversion of Federated Diffusion Models

Huang, Jiyue; Hong, Chi; Roos, Stefanie; Chen, Lydia Y.

DOI
10.1007/978-3-032-00624-0_19

Publication date
2025

Document Version
Final published version

Published in
Availability, Reliability and Security - 20th International Conference, ARES 2025, Proceedings

Citation (APA)
Huang, J., Hong, C., Roos, S., & Chen, L. Y. (2025). GIDM: Gradient Inversion of Federated Diffusion Models. In M. Dalla Preda, S. Schrittwieser, V. Naessens, & B. De Sutter (Eds.), *Availability, Reliability and Security - 20th International Conference, ARES 2025, Proceedings* (pp. 380-401). (Lecture Notes in Computer Science; Vol. 15992 LNCS). Springer. https://doi.org/10.1007/978-3-032-00624-0_19

GIDM: Gradient Inversion of Federated Diffusion Models

Jiyue Huang[1](✉), Chi Hong[1], Stefanie Roos[2], and Lydia Y. Chen[1,3]

[1] TU Delft, Delft, Netherlands
{j.huang-4,c.hong}@tudelft.nl
[2] RPTU Kaiserslautern, Kaiserslautern, Germany
stefanie.roos@cs.rptu.de
[3] University of Neuchâtel, Neuchâtel, Switzerland
yiyu.chen@unine.ch

Abstract. Diffusion models are becoming the most prevalent generative models, producing exceptional high-quality image data through a stochastic process of diffusion steps based on Gaussian noises. Recent studies explore the federated training of diffusion models, enabling the collaborative training of a model without clients sharing raw data. We demonstrate that even without direct sharing of the data, the shared gradients of federated diffusion models already leak sensitive information about the raw data. We design the first gradient inversion attack GIDM for diffusion, which can reconstruct the training data from the shared model updates. GIDM is a two-phase fusion attack that is both efficient and effective. In its first phase, GIDM leverages the trained diffusion model itself as prior knowledge to constrain the inversion search (latent) space, followed by a second phase of pixel-wise fine-tuning. Different from existing inversion attacks on the classification models, inverting diffusion models present new challenges, most notably that the noise term and randomly sampled diffusion step are not known to the attacker but are required for the reconstruction. To tackle this challenge, we propose a joint triple-optimization algorithm to approximate the raw data, sampling step, and noise term simultaneously. GIDM is shown to be able to reconstruct images almost identical to the original ones and clearly outperforms baselines, i.e., GIDM without the second phase and state-of-the-art attacks on classifiers adapted to diffusion. The code of our method is available at https://github.com/GillHuang-Xtler/Diffusion_inversion.

Keywords: Gradient inversion · Diffusion models · Reconstruction attack

1 Introduction

The emergence of likelihood-based diffusion models empowers probabilistic models to generate high-quality data, such as image and video data [11,20,24]. Diffusion models are trained by finding the reverse Markov transitions that maximize

M. Dalla Preda et al. (Eds.): ARES 2025, LNCS 15992, pp. 380–401, 2025.
https://doi.org/10.1007/978-3-032-00624-0_19

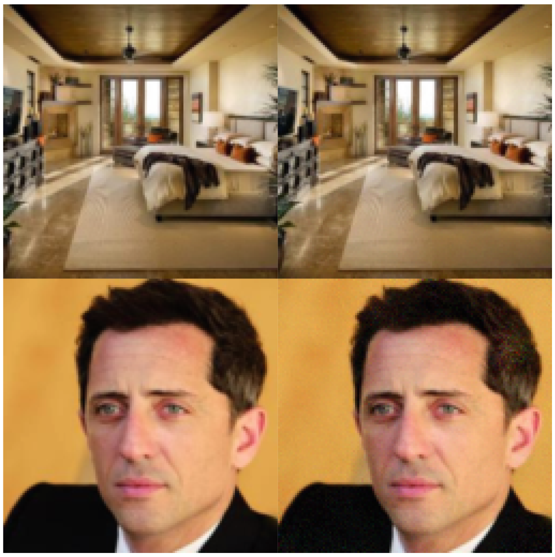


Fig. 1. Original training image (left) v.s. Recovered images (right) by our proposed GIDM on diffusion models: reconstructing almost identical to the original image of 128×128 size.

the likelihood of the training data. In practice, the training is done by gradually adding noise to and denoising the images over multiple steps.

However, training high-quality diffusion models usually requires a large amount of data. Practically, such data may be owned by different parties. Following data privacy regularization such as HIPAA [2] and GDPR [28], data owners are not allowed to share such data if it contains personally identifiable information, which includes in particular sensitive data such as medical records. As a consequence, prior works argue that diffusion models need to be trained in a distributed manner without sharing raw data, e.g., in a federated learning manner [17,22]. Indeed, the design of the Denoising Diffusion Probabilistic Model (DDPM) [11], the most common architecture for training diffusion models, enables distribution trivially: in each round, the algorithm independently samples a random Gaussian noise $\boldsymbol{\epsilon}$ for data sample(s) $\boldsymbol{x}_0$ at each sampling step t. Thus, there exist multiple designs for federated diffusion, which—while differing in the details of their implementation—all share the following approach: at each global round of training, multiple distributed data owners (clients) train the diffusion sub-models based on their local data. Then, the central server, which connects to every client, aggregates their gradients before returning the aggregated gradient to the clients for further training until convergence is reached.

Such a federated diffusion method successfully avoids direct data sharing and enables the server to obtain the intermediate training gradients and the diffusion model in the end. In this paper, we consider the privacy leakage caused by the shared gradients. While there have been privacy attacks on diffusion models, they do not consider gradients. For example, relying on inferring information from the trained model, e.g., error comparison of the forward process posterior estimation [5, 22], adversaries are able to launch membership inference attacks [5, 13, 32] or attribute inference attacks.

Although studies on federated training classifiers demonstrate that the server is able to invert the client's raw data from their gradients [8, 34, 40, 41], we argue that existing attacks cannot be easily transferred to diffusion models. The key idea of previous attacks is to reconstruct images from dummy data by using the gradients to constrain the search space. In the context of classification, label information needs to be present or reconstructed as part of the attack. In contrast, diffusion does not have labels; rather it requires the sampling step t during the training process and the noise prediction ϵ in each step, so that a novel approach is needed.

In this paper, we systematically study the data reconstruction risk when training federated diffusion models. We first define the common factors of federated training methods for diffusion models, according to related studies. Afterwards, we design our attack, which relies on diffusion-specific information that other scenarios lack. Specifically, at the end of the training process, the server owns a trained diffusion model, which we can use to constrain the search space for the reconstructed images. Using generative models has been explored in prior studies on inversion attacks on classification [7, 15] but these studies rely on external models.

Concretely, to reconstruct images that resemble the training data, we thus design a fusion optimization, GIDM, that includes two phases. The generating phase maps the dummy data into a narrow latent space to optimize in-distribution images by utilizing the diffusion model as prior knowledge. Then the fine-tuning phase further optimizes the similarity between the dummy and real gradients to update the dummy data generated during the first phase of the attack.

Approximating the gradient of diffusion models also requires the knowledge of ϵ and t, which may only be known to the clients, a key challenge of inversion attacks on diffusion models. To solve this, the two phases of GIDM include a novel triple-optimization for dummy data, ϵ, and t. Specifically, the triple-optimization includes three independent optimizers for the dummy data, the noise, and the sampling step to refine the joint training. By coordinating the three optimizers with updating intervals, we are able to recover images without knowing the exact values of ϵ and t.

Our proposed GIDM is able to recover images almost identical to the original data up to size 128×128, as shown in Fig. 1. Our evaluation further shows that the fine-tuning phase is indeed required and GIDM cannot reliable reconstruct images without using the triple-optimization. For comparison, we adapt two attacks for image classifiers, DLG [41] and InvG [8], to the scenario of diffusion. Note that as these attacks do not consider ϵ and t, we weaken the adversarial model for them by providing the concrete parameter values. Despite these attacks having more information, which would not be available in a real-world setting, GIDM outperforms them in terms of four key image similarity metrics.

We summarize our main contributions as follows. We present the first gradient inversion attack GIDM on diffusion models, leveraging three key novel ideas: the use of the trained diffusion model to constrain the search space; a two-phase attack consisting of a phase based on the existing diffusion model, and a fine-tuning phase; a triple-optimization method that jointly reconstructs the data, the sampling step, and the noise parameter. Our evaluation shows that GIDM can successfully reconstruct images of a large size and highlights the impact of the different attack components through ablation studies.

2 Related Work

Diffusion Models and Privacy. Diffusion models employ a two-step process: First, they deconstruct the training data structure step by step in a forward manner. Second, they master the reconstruction of the structure from noise in a reverse process. The Denoising Diffusion Probabilistic Model (DDPM) [11] introduces a stable and efficient implementation of diffusion for high-quality image synthesis. DDPM relies on a forward process without learnable parameters while employing simplified Gaussian noise in the reverse phase. Further variants of diffusion models such as DDIM [24], Stable Diffusion [20], and Imagen [21] improve the sampling efficiency or involve deep language understanding for text-to-image generation. However, well-trained diffusion models have been shown to be vulnerable to privacy attacks, i.e., information leakage on the training data. Recent studies on privacy concerns of diffusion models mainly focus on membership inference attacks [5,13,32] or training data memorizing attacks [4,23]; both of which are executed on the trained model. None of the studies has addressed the data reconstruction from the gradients of diffusion models.

Privacy Defenses of Federated Learning. In order to enhance system privacy against privacy leakage attacks and not significantly reduce model accuracy [3,10,26,35], current defenses are primarily conducted individually by clients, falling into two categories: gradient perturbation and input perturbation. Gradient perturbation [1,3,6,10,14,25,27,35], preferred for its efficiency and maintaining global model accuracy, involves transmitting perturbed gradients. In contrast, input perturbation, such as mixing images before local training, is less common [38]. Prominent gradient perturbation defenses include differential private stochastic gradient descent [6], which adds noise and clips gradients to limit sensitivity [1]. Gradient sparsification accelerates training by setting small gradient entries to zero [3], differing from dropout by removing small entries rather than randomly selecting them [10,14,25]. Soteria proposes a fully-connected defense layer to perturb data representation, crucial for preventing inversion attacks while preserving Federated Learning performance [26]. Overall, defense efficacy depends on parameters like noise level, clipping bound, sparsity, and pruning rate, which need to be carefully chosen to balance model quality and privacy.

Gradient Inversion. As the first practical gradient inversion attack for classifiers, Deep Leakage from Gradients (DLG) reconstructs data and label simultaneously by directly approximating gradients from the dummy data input [41].

DLG tends to reconstruct images of low quality and cannot deal with large training batches. To strengthen DLG, one line of work improves DLG by developing different optimizers [8,34], distance metrics [8], or integrating direct features [40], e.g., they first infer labels before reconstructing. The other line of work focuses on leveraging external knowledge for inversion. Such knowledge includes prior data distributions for more accurate embeddings [9], adding batch normalization regularizers to manage larger batches [36], applying pre-trained generator models to ensure high-quality reconstructed data [7,15,18] and utilizing auxiliary datasets [31]. GGDM iteratively refines the noise via a pre-trained unconditional DDPM as a guidance, but also targeting invert classifiers. These advanced attacks [7,15,16,36] successfully integrate additional information to improve the attack effectiveness. However, such knowledge can also be further integrated to our proposed GIDM. Thus, our direct comparison baselines are DLG and InvG.

Apart from classification models, inversion can be applied to Generative Adversarial Networks (GANs), where the attacks aim to invert a generated image back into the latent space of a pre-trained GAN model [33], i.e., reconstructing the latent code instead of the training images. Currently, there exist no inversion attack studied for inverting diffusion models.

3 Methodology

In this section, we first introduce preliminaries on federated diffusion models, highlighting the common components of existing frameworks. Then, we propose our inversion attack GIDM, which leverages the trained diffusion model as prior knowledge for constrained optimization, consisting of two phases. The generative phase improves the quality of a dummy image to achieve fast convergence, followed by a fine-tuning phase to increase the pixel-wise similarity. For both phases, we argue that the prior art of gradient inversion attacks does not apply to diffusion models due to the inability to handle more unknown factors, namely the noise ϵ and sampling step t, according to the DDPM training strategy. Thus, we design a triple-optimization algorithm to infer the original image, ϵ, and t simultaneously.

3.1 Federated Diffusion Model Preliminaries

We consider a federated image generation task following the standard DDPM diffusion models [11], which aims to optimize the weighted variational bound:

$$L(\theta) = \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}} \left[\left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta \left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t \right) \right\|^2 \right], \tag{1}$$

where $\mathbf{x}_0 \in \mathbb{R}^m$ are training samples of dimension $m = \text{width} \times \text{height} \times \text{color}$, $L(\cdot)$ is the point-wise loss function, θ denotes the diffusion model network parameters and $\bar{\alpha}_t$ is a hyper-parameter controlling the forward noising process. Note that the sampling step t is chosen uniformly between 1 and T and we follow the definition [11] that $\boldsymbol{\epsilon}_\theta$ is a function approximator intended to predict the Gaussian noise $\boldsymbol{\epsilon}$ added to the image $\mathbf{x}_t$ of step t.

Algorithm 1. Federated Diffusion Model Training

```
Input: The number of clients $K$, number of global training round $R$, local datasets $X_k, k \in [1, K]$, diffusion steps $T$, global learning rate $\eta$.
Initialize model $\theta$
for $r = 1, 2, ..., R$ do
    for $k = 1, 2, ..., K$ do
        $t \sim \text{Uniform}(\{1, ..., T\})$
        $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
        $\mathbf{x}_0 \sim X_k$
        $g_k = \nabla_{\theta_k} \left\| \boldsymbol{\epsilon}_k - \boldsymbol{\epsilon}_{\theta_k} \left( \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_k, t \right) \right\|^2$
    $g = 1/K \sum_{k=1}^{K} g_k$
    Update $\theta = \theta - \eta g$
Return the trained $\theta^*$ ($\theta$ from the last round)
Result: the trained $\theta^*$
```

There are K clients serving as data owners who are responsible for diffusion model training. The k^{th} federated learning client owns the local real dataset $X_k, k \in \{1, 2, ..., K\}$, which is not shared with others. Each client reports the gradient:

$$\nabla_{\theta} \left\| \boldsymbol{\epsilon}_k - \boldsymbol{\epsilon}_{\theta_k} \left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_k, t \right) \right\|^2,$$

for the locally sampled data $\mathbf{x}_0$. We use R global training rounds. The single server $\mathcal{S}$, which does not own any data itself, aggregates, usually by computing their average, and distributes the aggregated model in each global training round.

In contrast to other federated learning models, e.g., classification tasks, diffusion models require sampling the parameters $\boldsymbol{\epsilon}$ and t. We assume that they are sampled by the clients who own the data during the training process. Consequentially, the server does not know which parameters were chosen by each client. In the evaluation, we compare this setting to a less privacy-preserving variant where the server chooses $\boldsymbol{\epsilon}$ and t and distributes them to the clients in Sect. 4.4 and Sect. 4.5. Both settings can be implemented as equivalent optimization problems to centralized diffusion models. The general training process is given in Algorithm 1.

3.2 Threat Model

Our threat model considers the federated server $\mathcal{S}$ as the adversary to reconstruct the input training data X_k of the target client C_k. The threat model is as follows.
Objective. The adversarial server aims to recover the input data X_k trained by client C_k based on the gradient:

$$g_k = \nabla_{\theta} \left\| \boldsymbol{\epsilon}_k - \boldsymbol{\epsilon}_{\theta_k} \left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_k, t \right) \right\|^2,$$

during a specific global training round r. As the inversion can be executed at one given global round, we drop the index r for simplicity. The attack is successful if the recovered image $\hat{X}$ is almost identical to X_k.

Capability. We assume that the honest-but-curious server does not have access to the real data of data owners. Moreover, the servers' computational resources are limited, so it cannot, e.g., break cryptographic primitives.
Knowledge. To recover input data at a specific round, we assume $\mathcal{S}$ naturally owns the global model and can access the model parameters and the submitted gradient of all clients. However, the initialization of ϵ and the sampling step of t are unknown to the adversary unless stated otherwise.

3.3 Two-Phase Inversion with Diffusion Prior

To assess the privacy leakage of federated diffusion, we propose the first gradient inversion attack GIDM: an honest-but-curious server reconstructs a victim client's data using the gradients submitted by the client. We model the inversion process as an optimization problem that iteratively modifies the dummy data by minimizing the distance between known and approximated gradients, as proposed by the inversion attacks for federated classifier training [41]. Following Algorithm 1, when client k computes the gradient for the training data $\mathbf{x}_0$, the gradient is:

$$g_k = \nabla_{\theta_k} \left\| \boldsymbol{\epsilon}_k - \boldsymbol{\epsilon}_{\theta_k} \left(\sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t} \boldsymbol{\epsilon}_k, t \right) \right\|^2 .$$

Note that since a gradient inversion attack can be launched at any specific global training round for any client, we drop the round and client indexes r and k in the following. Assuming that θ is second-order differentiable, we suppose that the dummy data $\hat{\mathbf{x}}_0$ is an approximation of $\mathbf{x}_0$ if $\hat{g} \sim g$, where $\hat{g}$ is the dummy gradient calculated based on $\hat{\mathbf{x}}_0$. Thus, our gradient inversion objective turns to:

$$\min_{\hat{x}_0} Dist(\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta \left(\sqrt{\bar{\alpha}_t} \hat{x}_0 + \sqrt{1-\bar{\alpha}_t} \boldsymbol{\epsilon}, t \right) \right\|^2 , g), \tag{2}$$

where $Dist(\cdot)$ is a distance metric for two gradients.

Without constraining the search space, the difficulty of inversion increases exponentially with the image size [41]. We leverage the final trained diffusion model, which the server naturally possesses as it is the result of the training process, to constrain the search space. In contrast to prior work, no external pre-trained model is required. Concretely, we propose a constrained fusion model consisting of two phases: the generative phase and the fine-tuning phase. First, the generative phase leverages prior knowledge of the trained final diffusion model, $\mathcal{D}_{\theta^*} : \mathbb{R}^m \rightarrow \mathbb{R}^m$, where θ^* is the diffusion model parameter. Then, the fine-tuning phase improves the dummy image from the generative phase by efficiently minimizing the pixel-wise similarity. The overall workflow is summarized in Fig. 2, where the first generative phase outputs the optimized latent code as the input to the fine-tuning phase. The fine-tuning phase then output the final recovered images.

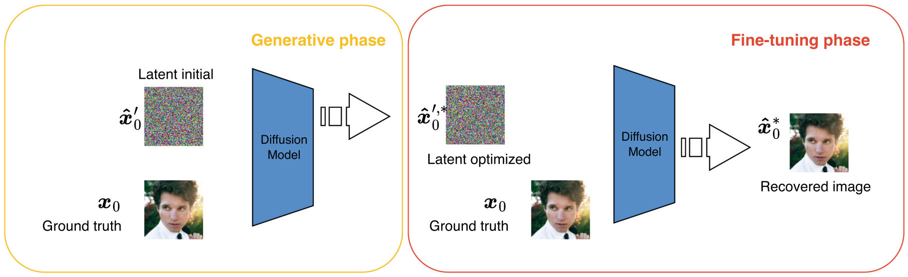


Fig. 2. The workflow of two phases of GIDM: Generative phase to optimize the latent code, while the fine-tuning phase continues to improve the pixel-wise similarity between the recovered images and the original ones to achieve both efficiency and effectiveness.

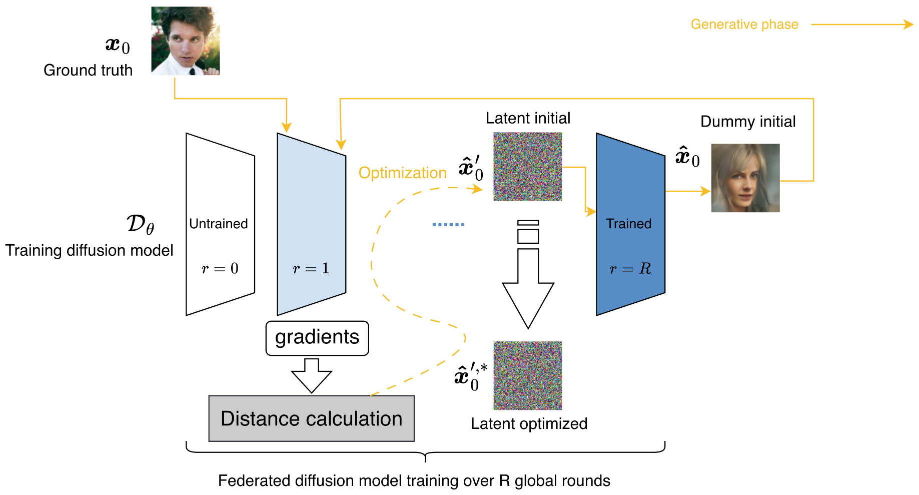


Fig. 3. We utilize one 128 × 128 image from *CelebA* as an example to show every intermediate result above. The generative phase executes 5000 iterations in this example, finally outputting the high-quality latent code which is able to sample resembled images from the trained diffusion models.

Generative Phase. We aim to generate images that resemble the training dataset as the starting point for dummy data optimization by gradient approximation. Our approach is to map the original search space into a narrow latent space with constraints (based on prior knowledge). The details of the generative phase is presented in Fig. 3. Let the latent code $\hat{\mathbf{x}}'_0$ [29] be of the same dimension as the dummy data $\hat{\mathbf{x}}_0$ from Eq. 2 and $\hat{\mathbf{x}}_0 = \mathcal{D}_\theta(\hat{\mathbf{x}}'_0, t)$. We execute multiple iterations to optimize $\hat{\mathbf{x}}'_0$, as described below, so that the gradient computed on $\hat{\mathbf{x}}_0$ has a small distance to the real gradient. Thus, instead of directly updating the dummy data, we perform a latent space search over $\hat{\boldsymbol{x}}'_0$, which is the input of $\mathcal{D}_\theta$, outputting $\hat{\mathbf{x}}_0$. That is:

$$\hat{\mathbf{x}}_0^{\prime,*} = \mathcal{D}_\theta \left(\arg\min_{\hat{\mathbf{x}}_0^\prime} Dist(\delta, g)\, , t \right),$$

with:

$$\delta = \nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta \left(\sqrt{\bar{\alpha}_t} \mathcal{D}_\theta(\hat{\mathbf{x}}_0^\prime, t) + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t \right) \right\|^2 .$$

Fine-Tuning Phase. The generative phase is able to generate high-quality data, yet, indirectly optimizing $\hat{\mathbf{x}}_0$ does not guarantee pixel-wise similarity. Moreover, each iteration of optimizing the latent code $\hat{\mathbf{x}}_0^\prime$ requires T sampling steps of the trained diffusion model, e.g., $T = 1000$ in DDPM, which is computationally expensive. Therefore, our fine-tuning phase executes direct optimizing of $\hat{\mathbf{x}}_0$ following Eq. 2 based on the output of the generative phase $\hat{\mathbf{x}}_0^{\prime,*}$ to increase the pixel-wise similarity. The details of the finet-tuning phase is presented in Fig. 4.

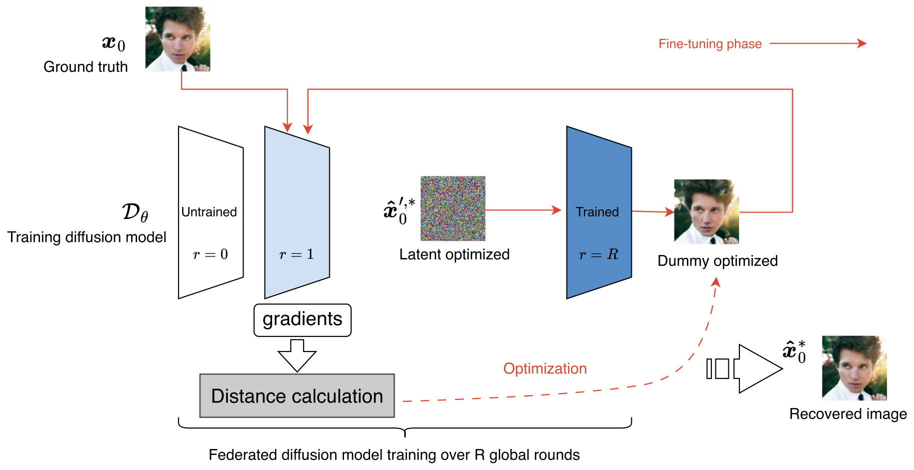


Fig. 4. We utilize one 128 × 128 image from *CelebA* as an example to show every intermediate result above. The fine-tuning phase consists of 1500 iterations. The latent optimized $\hat{\mathbf{x}}_0^{\prime,*}$ is the output of the generative phase.

The output of the generative phase, i.e., the intermediate dummy data, generated by the optimized latent code $\hat{\mathbf{x}}_0^{\prime,*}$, is then optimized by the fine-tuning phase. From Fig. 4, we see that the generative phase is already able to reconstruct a high-quality image that overall resembles the original picture while the pixel-wise similarity with the original data is low. As in the comparison example, The output of the generative phase differs from the original data in hair, face, and background. By integrating the generative phase and the fine-tuning phase, we recover high-quality data efficiently and effectively with high pixel-wise similarity.

Within both phases, calculating the gradients requires the knowledge of private $\boldsymbol{\epsilon}$ and t. We first conduct an exploratory experiment to determine whether inversion methods for classifiers adapted for diffusion models enable successful attacks. The result, presented in Sect. 4.4, demonstrates that adapted existing attacks are unable to recover training samples when applied to diffusion models due to the large search space of the diffusion optimization procedure. This motivates the novel design for gradient approximation on diffusion models.

3.4 Triple-Optimization

To enable inversion without knowing $\{\boldsymbol{\epsilon}, t\}$, we optimize three different parameters simultaneously while taking their design principles and differences into consideration. Concretely, we design a triple-optimization method to refine the coordination of the three independent optimizations, namely, of $\mathbf{x}_0$, $\boldsymbol{\epsilon}$, and t by determining the approximations $\hat{\mathbf{x}}_0$, $\hat{\boldsymbol{\epsilon}}$, and $\hat{t}$, respectively. Note that all three optimizations are based on backpropagating with the goal approximating the gradient.

Optimizing $\hat{\boldsymbol{\epsilon}}$. In DDPM, we compute the distribution of the noisy sample after t iterations of the forward process in closed-form by:

$$q\left(\mathbf{x}_t \mid \mathbf{x}_0\right) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, \left(1-\bar{\alpha}_t\right)\mathbf{I}\right).$$

The number of iterations in the forward process is set to a large T, and the variance levels $\beta_t \in (0,1)$ increase linearly (ranging from 10^{-4} to 0.02), which means that $\bar{\alpha}_t = \prod_{i=1}^{t}\left(1-\beta_t\right)$ approximates 0 for large t. Thus, the latent distribution is a Gaussian distribution ϵ sampled locally by the client, i.e., we should have $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. To follow such a Gaussian distribution, i.e., reduce the probability of sampling unlikely values, when implementing stochastic gradient descent requires a small learning rate η_ϵ to update $\hat{\boldsymbol{\epsilon}}$. Also, following Eq. 1, training diffusion models is to predict the noise added during the forward process and the training performs well without changing the noise term during every iteration of inversion. Thus, we utilize an interval updating strategy for $\hat{\boldsymbol{\epsilon}}$. The updating is achieved by:

$$\hat{\boldsymbol{\epsilon}}^{i+1} = \hat{\boldsymbol{\epsilon}}^{i} - \eta_\epsilon \frac{\partial Dist(\hat{g}, g)}{\partial \hat{\boldsymbol{\epsilon}}^{i}}.$$

Optimizing $\hat{t}$. In contrast to $\boldsymbol{\epsilon}$ and $\mathbf{x}_0$, which are floating tensors, t is an discrete integer in $\{1, \ldots, T\}$. To find the optimal t by stochastic gradient descent, we initialize a $1 \times T$ auxiliary vector following the uniform distribution: $\hat{\boldsymbol{t}} \sim \text{Uniform}(\mathbf{0}, \boldsymbol{J})$ where $\boldsymbol{J}$ is a vector of ones. In each iteration of optimization for approximating the dummy gradient to the real gradient, $\hat{\boldsymbol{t}}$ is updated by the learning rate η_t. The inferred $\hat{t}$ is chosen as the index of the maximum element in the vector after softmax transition: $\hat{t} = \arg\max(softmax(\hat{\boldsymbol{t}}))$. $\hat{t}$ is updated at each inversion iteration:

$$\hat{t}^{i+1} = \hat{t}^{i} - \eta_t \frac{\partial Dist(\hat{g}, g)}{\partial \hat{t}^{i}}.$$

Optimizing $\hat{\mathbf{x}}_0$. Replacing $\boldsymbol{\epsilon}$ and t in Eq. 2 with $\hat{\boldsymbol{\epsilon}}$ and $\hat{t}$, we perform the optimization of $\hat{\mathbf{x}}_0$ starting with a random initialized image $\hat{\mathbf{x}}_0 \sim \text{Uniform}(\mathbf{0}, 2^p\boldsymbol{I})$, given that the source data is a p-bit binary encoded value. With each iteration, we update $\hat{\mathbf{x}}_0$ by back-propagating the distance between $\hat{g}$ and g. We set the learning rate of the dummy data as η_x. $\hat{\mathbf{x}}_0$ is updated at each inversion iteration:

$$\hat{\mathbf{x}}_0^{i+1} = \hat{\mathbf{x}}_0^i - \eta_x \frac{\partial Dist(\hat{g}, g)}{\partial \hat{\mathbf{x}}_0^i}.$$

The optimizers are coordinated based on an optimization interval S. After every S iterations of updating $\hat{\mathbf{x}}_0$ and $\hat{t}$, we perform S iterations of $\hat{\boldsymbol{\epsilon}}$, $\hat{\mathbf{x}}_0$ and $\hat{t}$ simultaneously.

In summary, the triple-optimization gradient inversion coordinates three independent optimization processes to perform data reconstruction for the practical federated diffusion models without knowledge of the private noise and step sampled by each client.

4 Experimental Evaluation

We design a set of experiments to study the impact and effectiveness of gradient inversion attacks on diffusion models, as well as the ablation studies. The experimental setups, baselines, evaluation metrics, employment details, and results are presented in this section.

4.1 Setups

Datasets. Unless stated otherwise, our experiments use two datasets : *Celeb-A* [19] and *LSUN-Bedroom* [37], both with images resized to 128×128 since the original images from these datasets are of varying sizes. Both datasets are trained using the standard DDPM model [11] as by Algorithm 1 with 50 rounds. The optimization interval of GIDM for $\hat{\boldsymbol{\epsilon}}$ optimizer is set to be $S = 50$. For all three optimizations of our triple-optimization, we use Adam as the optimizer. We choose $Dist(\cdot)$ to be the L2-Norm [41] distance, as the distance calculated by cosine similarity is large in our model, which may cause exploding gradients during training.
Hardware. For hardware, we conducted our experiments using an Alienware Aurora R13 running Ubuntu 20.04. This system boasts 64 GB of memory, a GeForce RTX 3090 GPU, and a 16-core Intel i9 CPU. With each of its 8 P-cores supporting two threads, the machine houses a total of 24 logical CPU cores.
Hyperparameters. The project of this paper is based on Pytorch 2.3.0. In our gradient inversion attackers, we apply Adam as the optimizer for both $\boldsymbol{\epsilon}$, t and the dummy image. The learning rate used in this paper is 0.01 without any scheduler for learning rate decay. The diffusion model for our experiments is the commonly used DDPM, which uses UNet to implement the sampling of each step. For our topic, the server can reconstruct the data from each client. Thus, we experiment on one random client for this work, following the settings of the baseline works [34,41].

4.2 Baselines with Adaptation

Since we are the first to study the gradient inversion attack on diffusion models, there is no direct baseline to compare to. State-of-the-art inversion attacks are designed for classifiers. Thus, we compare GIDM with adapted versions of the attacks that are compatible with diffusion models: DLG-dm [41] and InvG-dm [8] with Adam optimizer. Please note that without knowing $\{\boldsymbol{\epsilon}, t\}$, DLG-dm and InvG-dm are **not** able to invert images similar to the original. As a consequence, we provide $\{\boldsymbol{\epsilon}, t\}$ for these methods, giving them additional information that GIDM does not have. Specifically, for the same training input of images, the initialization works the same for DLG-dm and InvG-dm as for DLG and InvG. The backpropagation for calculating the gradients of diffusion models requires the additional inputs of $\boldsymbol{\epsilon}$ and t, which does not apply to the baseline DLG and InvG. Thus, for DLG-dm and InvG-dm on diffusion models, we randomly sample $\boldsymbol{\epsilon}$ following a Gaussian distribution of the same size of the dummy image and t from $[1, T]$ following a uniform distribution. They are kept constant over all inversion iterations. When it comes to the distance metric, we keep L2-Norm for DLG-dm and cosine similarity for InvG-dm, as for DLG and InvG.

4.3 Metrics

In evaluating the gradient inversion attack, which aims to reconstruct data that closely matches the original client data, we employ four metrics to gauge the resemblance between the reconstructed and real data: Mean Squared Error (MSE), Structural Similarity Index (SSIM) [30], Peak Signal-to-Noise Ratio (PSNR) [12], and Learned Perceptual Image Patch Similarity (LPIPS) score [39]. Statistically, *MSE* calculates the average squared difference on a pixel level between the reconstructed and original images:

$$\mathrm{MSE}\left(q_1, q_2\right) = \frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left(q_1(i,j) - q_2(i,j)\right)^2,$$

where $q_1(i,j)$ and $q_2(i,j)$ indicate the original and recovered images, respectively. *PSNR* relates this *MSE* to the maximum pixel values, concretely, considering the ratio of the maximal value to *MSE* in a logarithmic manner:

$$\mathrm{PSNR} = 10 \log_{10} \frac{R^2}{\mathrm{MSE}},$$

where R is the maximum possible pixel value of the image. When the pixels are represented using 8 bits per sample, this is 255. For more modern visual assessments, *SSIM* mimics the human visual system to measure the structural variance of images based on luminance, contrast, and structure:

$$SSIM = \frac{2\mu_{x_r}\mu_{x_x} + c_1 2\sigma_{x_x x_x} + c_2}{\mu_{x_r}^2 + \mu_{x_z}^2 + c_1 \sigma_{x_r}^2 + \sigma_{x_y}^2 + c_2},$$

where μ_{x_r} and μ_{x_g} represent the means of the ground truth and the generated image, respectively. Accordingly, $\sigma_{\hat{x}_r}$ and σ_{x_g} are the standard deviations of $\hat{x}_r$ and x_g. Moreover, σ denotes the covariance between both images, while c_1 and c_2 are constants set to avoid instability.

LPIPS determines the perceptual similarity between the original and reconstructed images by learning the inverse mapping from the generated image back to the original. For MSE and LPIPS, lower values indicate greater similarity, whereas for SSIM and PSNR, higher values signify closer resemblance.

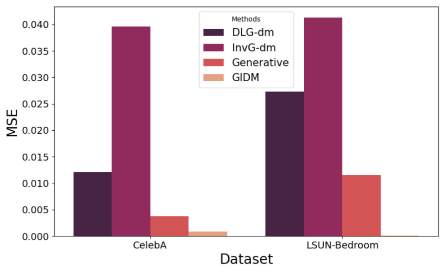
4.4 Final Reconstructed Images

The performance of our GIDM is assessed by the similarity between the final recovered image and the original image used during training. To demonstrate our inversion effectiveness, we report the final MSE, LPIPS, PSNR, SSIM results in Fig. 5. We also visualize examples of the final reconstructed images in Fig. 6 for *CelebA* and *LSUN-bedroom*. For GIDM, we assume $\{\boldsymbol{\epsilon}, t\}$ is sampled by the client and randomly initialized: $\hat{\boldsymbol{\epsilon}} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ and $\hat{t} \sim \text{Uniform}(\{1, ..., T\})$. Recall that for the baselines, $\{\boldsymbol{\epsilon}, t\}$ is assumed to be known. "Generative" refers to only the output of the generative phase before the fine-tuning phase, with $\{\boldsymbol{\epsilon}, t\}$ known.

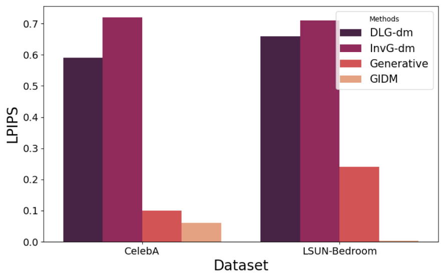
From Fig. 5, it is evident that GIDM outperforms baseline methods consistently for all four evaluation metrics, demonstrating superior reconstructing results. Yet, our generative phase does not always recover better images than DLG-dm and InvG-dm, which can be explained by the difference in goals in terms of similarity. The four evaluation metrics compute the quality of similarity based on pixels. In contrast, "Generative" optimizes the latent space to conduct indirect inversion. Thus, the high-quality output (semantically similar and clear) from the diffusion model may result in lower pixel-wise similarity than baselines, which appear comparably blurred in Fig. 6. Adding the fine-tuning phase resolves the issue and achieves better performance in terms of the metrics despite using less information.

Figure 6 visualizes and compares the recovered images on both datasets. GIDM successfully reconstructs high-quality images from the gradients that are nearly indistinguishable from the ground truth perceptually. As a comparison, DLG-dm and InvG-dm, which are designed for classifiers, fail to recover images resembling the ground truth. Specifically, they are only able to create images of similar color palettes as the original data without recreating the original object, let alone high-quality details. This meets our expectations since they conduct pixel-wise optimization. These methods suffer from exponentially increased difficulty when the image size is large.

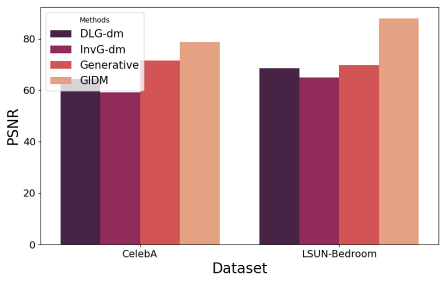
When it comes to the difference between "Generative" and GIDM, we observe that our generative phase can already recover good approximations of the original image. Generally, the main color, object outline, and positions after the generative phase are almost identical to the ground truth, though there are still minor differences in the details of the images. For example, the face and hair shape, the background texture, or sometimes the makeup color is different from the ground truth for the *CelebA* human-facial dataset. The fine-tuning phase



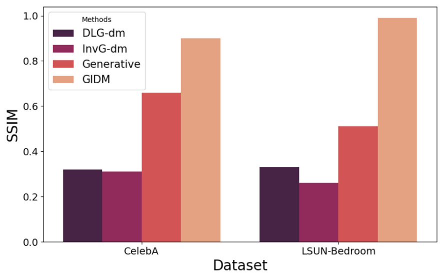
(a) **MSE** ↓



(b) **LPIPS** ↓



(c) **PSNR** ↑



(d) **SSIM** ↑

Fig. 5. Inversion results of diffusion models on *CelebA* and *LSUN-Bedroom*. We compare our GIDM (unknown $\{\boldsymbol{\epsilon}, t\}$) with DLG-dm, InvG-dm, and our generative phase result (all three known $\{\boldsymbol{\epsilon}, t\}$). ↓ stands for the lower the better, ↑ for the higher the better.

adjusts the generated images by direct gradient approximation, which results in successful reconstruction.

To summarize, gradient inversion of diffusion models is possible. Using the trained diffusion model to constrain the search space is highly effective. Still a fine-tuning phase is required to indeed recover high-quality images. With this phase, GIDM clearly outperforms the adapted baselines, despite using less knowledge.

4.5 Intermediate Inversion Outputs of GIDM

We now analyze the quality of the reconstructed images during the optimization process. We use example images from the *CelebA* dataset, both for known and unknown $\{\boldsymbol{\epsilon}, t\}$.

Figure 7 shows the intermediary results for our example image. It highlights that both phases are necessary for the inversion process. For both known and unknown $\{\boldsymbol{\epsilon}, t\}$, we observe gradual improvements, with the color palette and the profile details getting more similar to the original image in each step.

For the generative phase, the initialized random Gaussian noise can directly output a distinguishable image of a human face. However, this image shows an

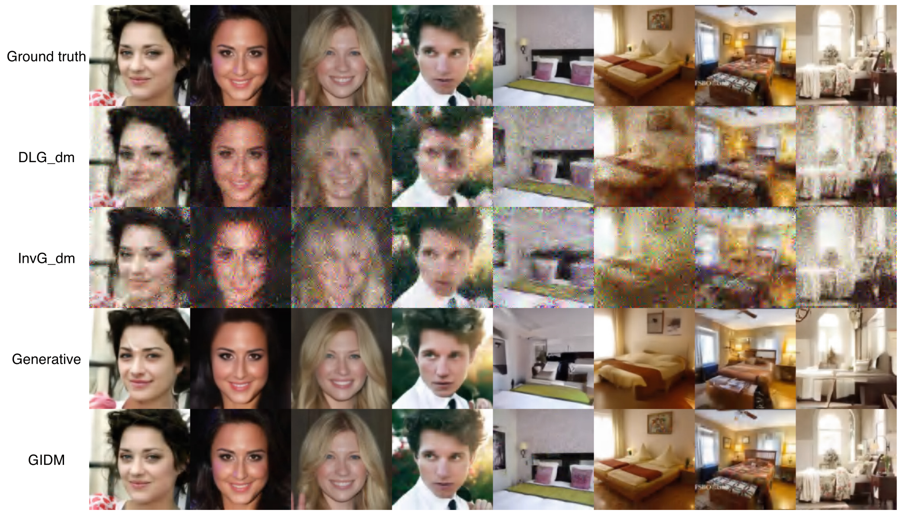


Fig. 6. Visualization of recovered images comparing with baselines on *CelebA* and *LSUN-Bedroom* datasets. "Generative" in the figure is the output of our generative phase before the fine-tuning phase. For GIDM, we assume that the server does not know $\{\boldsymbol{\epsilon}, t\}$, while the baselines assume known $\{\boldsymbol{\epsilon}, t\}$.

arbitrary human face, with no strong similarity to the original image. Thus, we have constrained the search space to images of human faces, which now enables finding one particular human face. We hence see that leveraging the trained diffusion model as a prior is highly effective in narrowing down the set of potential results.

After calculating the dummy gradient based on the previous image, our back-propagation adjusts the Gaussian noise so that it gradually generates an image of a similar color palette and object profiles. One interesting finding is that GIDM can reconstruct similar colors at an early stage of the training. In contrast, object outlines, such as the hair, converge slowly and gradually. Upon reaching the 3000-th iteration of the generative phase, the recovered image has been gradually started resembling the original image but does not mirror all the details of the original. Note that increasing the number of iterations to 10,000 does still not result in a fully recovered image, further motivating the need for the fine-tuning phase.

The fine-tuning phase starts with the output of the generative phase and we train 1500 iterations. Following the red arrows, the hair outline, and background start to approximate the original data. The fine-tuning phase does not utilize the diffusion model for optimizing alternative space. Thus, the reconstruction is efficient by not unnecessarily executing diffusion model sampling at each optimization iteration as in the generative phase.

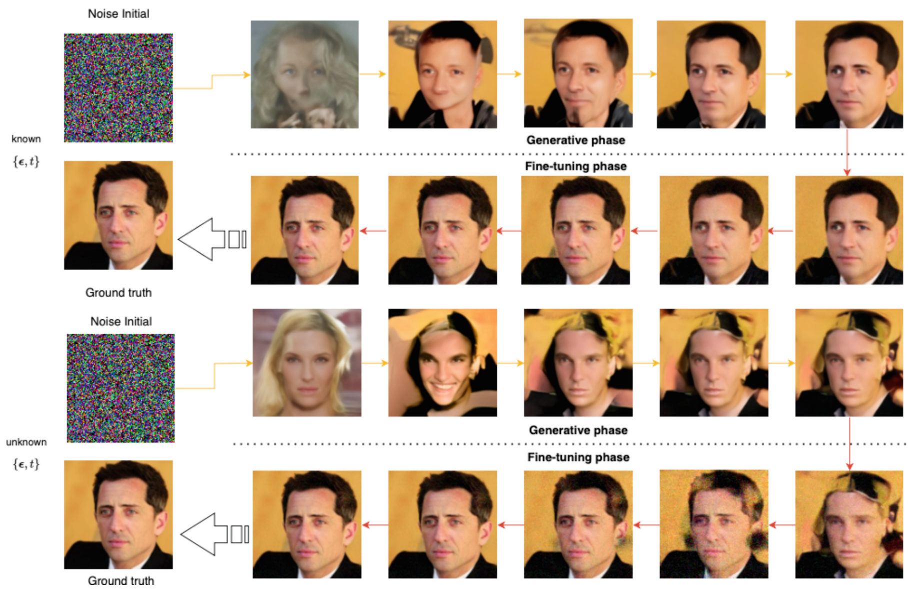


Fig. 7. Step-by-step gradient inversion intermediate results visualization for known and unknown $\{\boldsymbol{\epsilon}, t\}$. The generative phase is marked by yellow arrows while the fine-tuning phase is red arrows. We initialize the latent code using 128×128 Gaussian noise. The generative phase is trained for $r_g = 5000$ and the fine-tuning phase for $r_f = 1500$ iterations. The same number n_i of images is shown for both phases, with the iterations between images being r_g/n_i and r_f/n_i for the generative and fine-tuning phases, respectively. (Color figure online)

4.6 The Impact of Knowing t or $\boldsymbol{\epsilon}$ on GIDM

Our results in Sect. 4.4 consider $\boldsymbol{\epsilon}$ and t initialized together by the client, which is unknown to the server. Here we provide an ablation study on the impact of knowing one factor and keeps the other one unknown respectively. We start by discussing the impact of known and unknown t. Hence, we assume that $\boldsymbol{\epsilon}$ is known in this section. We use an image from *CelebA* as an example and show the process of optimization in Fig. 8. From the steps, it can be observed that when t is known to the server, the optimization process is smooth. This demonstrates that generating the changing t as the input of $\mathcal{D}(\hat{\boldsymbol{x}}_0, t)$ influences the approximation during the optimization process of t. However, even without knowing t, the 128×128 images can be recovered in the end due to the less randomness than $\boldsymbol{\epsilon}$.

As a comparison, we also evaluate the impact of known or unknown ϵ for the reconstruction. Here, it is also assumed that t is known to the server (adversary). The results are shown in Fig. 9. Similarly, in the end, our GIDM is shown to be able to reconstruct images that resembles the original (ground truth in the figure). However, the intermediate output shows different way approaching the end. In general, the middle steps looks more natural perceptually with ϵ known by the

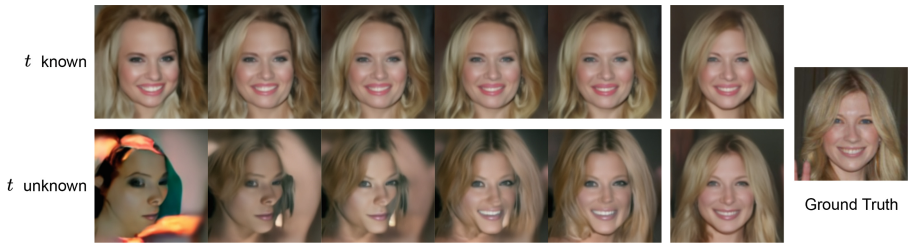


Fig. 8. Step-by-step gradient inversion intermediate results visualization, comparing known or unknown t. We initialize the latent code by 128×128 Gaussian noise. We present some of the representative intermediate outputs through the whole process. The total number of inversion iterations is 4000.

server. We could observe smooth changes towards the final results. Additionally, each output demonstrates high quality in terms of both the color blocks and facial profiles (figures above). As a comparison, unknown ϵ causes some deformed images, represented by both color and profiles (figures below). When it comes to the difference between Fig. 9 and Fig. 8, we find that t influences more on the image content (e.g., natural undistorted face) while ϵ controls more on the image clarity (blurness and random dots).

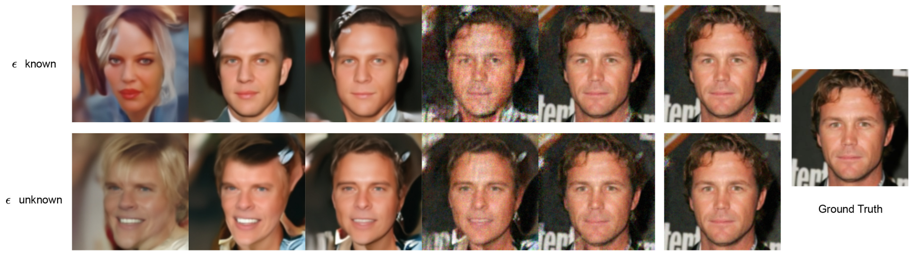


Fig. 9. Step-by-step gradient inversion intermediate results visualization, comparing known or unknown ϵ. We initialize the latent code by 128×128 Gaussian noise. We present some of the representative intermediate outputs through the whole process. The total number of inversion iterations is 4000.

4.7 Effectiveness of Triple-Optimization

In this part, we conduct an ablation study to showcase the significance of our joint triple-optimization. We apply the two-phase fusion model to compare the final recovered image without optimizing $\hat{t}$ or $\hat{\boldsymbol{\epsilon}}$. An example reconstruction can be seen in Fig. 10.

In Fig. 10, we can first observe that it is indeed necessary to optimize the three factors jointly. Without such a joint optimization, the reconstructed images do not resemble the original image closely. The difference in importance between the two terms is notable. Concretely, when $\hat{t}$ is not optimized, the resulting image is very close to random pixels. In contrast, when $\hat{t}$ is optimized but $\hat{\boldsymbol{\epsilon}}$ not, the reconstructed images is clearly that of a human face, though the details are not reconstructed.

Although $\hat{t}$ is only a single value that has a limited range, e.g., integer values between 0 and 999 for the classical DDPM, it decides the specific position on the Markov chain to train on. It guides the training process, similar to the label of classification tasks, which has a huge impact on the gradients. Thus, without having $\hat{t}$ resemble the parameter used during training, we have a very noisy image.

Without optimizing the noise term $\hat{\boldsymbol{\epsilon}}$, which is a float tensor with more randomness, the image quality remains low but the attack is able to recover most color palettes and some outline information. We argue that there might be two reasons. First, $\hat{\boldsymbol{\epsilon}}$ is introduced as a prediction target for training, which is less sensitive to incorrect sampling than $\hat{t}$, which directly instructs the noising process through the position of the chain. Second, the noise term is designed to follow a Gaussian distribution, avoiding unlikely values during inversion avoids extreme differences. In summary, our triple-optimization is indeed required.

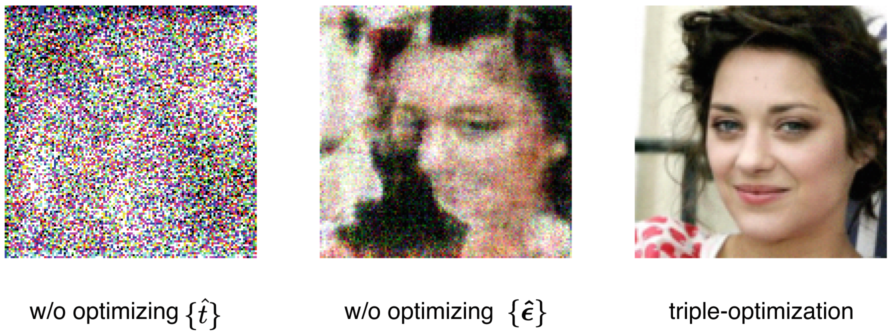

w/o optimizing $\{\hat{t}\}$
w/o optimizing $\{\hat{\boldsymbol{\epsilon}}\}$
triple-optimization

Fig. 10. The reconstruction results of without optimizing $\hat{t}$ (random sampled), without optimizing $\hat{\boldsymbol{\epsilon}}$ (random sampled) and triple-optimization on an example of *CelebA*.

5 Conclusion

In this paper, we are the first to study gradient inversion attacks on diffusion models. Leveraging the trained diffusion models as prior knowledge to constrain the search space, we propose an attack GIDM consisting of two phases. In the first phase, we use the trained diffusion model to approximate the image before fine-tuning the image to closer resemble the origin on a pixel-by-pixel basis in the second phase. The key challenge in designing GIDM is that the noise and sampling step are not known to the attacker, which we solved by the use of a joint optimization algorithm. Our reconstruction results are impressive, almost perfectly recreating images up to 128×128 in size.

In the future, we aim to develop a defense mechanism against the inversion attack, e.g., by using differential privacy [6]. It is unclear how differential privacy can be integrated in such a manner that the quality of the trained diffusion model is not severely impacted.

Acknowledgement. This research was partly funded by the SNSF/DFG project PrivGSyn at 200021E_229204/550302673.

References

1. Abadi, M., et al.: Deep learning with differential privacy. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016, pp. 308–318. ACM (2016)
2. Act, A.: Health insurance portability and accountability act of 1996. Public Law **104**, 191 (1996)
3. Aji, A.F., Heafield, K.: Sparse communication for distributed gradient descent. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, 9–11 September 2017, pp. 440–445. Association for Computational Linguistics (2017)
4. Carlini, N., et al.: Extracting training data from diffusion models. In: Calandrino, J.A., Troncoso, C. (eds.) 32nd USENIX Security Symposium, USENIX Security 2023, Anaheim, CA, USA, 9–11 August 2023, pp. 5253–5270. USENIX Association (2023)
5. Duan, J., Kong, F., Wang, S., Shi, X., Xu, K.: Are diffusion models vulnerable to membership inference attacks? In: International Conference on Machine Learning, ICML 2023, 23–29 July 2023, Honolulu, Hawaii, USA, vol. 202. Proceedings of Machine Learning Research, pp. 8717–8730. PMLR (2023)
6. Dwork, C.: The differential privacy frontier (extended abstract). In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 496–502. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00457-5_29
7. Fang, H., Chen, B., Wang, X., Wang, Z., Xia, S.-T.: GIFD: a generative gradient inversion method with feature domain optimization. In: IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, 1–6 October 2023, pp. 4944–4953. IEEE (2023)
8. Geiping, J., Bauermeister, H., Dröge, H., Moeller, M.: Inverting gradients - how easy is it to break privacy in federated learning? In: Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, 6–12 December 2020, Virtual (2020)
9. Guan, Z., Zhou, Y., Gu, X., Li, B.: GIE: gradient inversion with embeddings. In: IEEE International Conference on Multimedia and Expo, ICME 2024, Niagara Falls, ON, Canada, 15–19 July 2024, pp. 1–6. IEEE (2024)
10. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Improving neural networks by preventing co-adaptation of feature detectors. CoRR, abs/1207.0580 (2012)

11. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. In: Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, 6–12 December 2020, Virtual (2020)
12. Horé, A., Ziou, D.: Image quality metrics: PSNR vs. SSIM. In: 20th International Conference on Pattern Recognition, ICPR 2010, Istanbul, Turkey, 23–26 August 2010, pp. 2366–2369. IEEE Computer Society (2010)
13. Hu, H., Pang, J.: Loss and likelihood based membership inference of diffusion models. In: Athanasopoulos, E., Mennink, B. (eds.) Information Security - 26th International Conference, ISC 2023, Groningen, The Netherlands, 15–17 November 2023, Proceedings. LNCS, vol. 14411, pp. 121–141. Springer (2023)
14. Huang, Y., Gupta, S., Song, Z., Li, K., Arora, S.: Evaluating gradient inversion attacks and defenses in federated learning. In: Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, 6–14 December 2021, Virtual, pp. 7232–7241 (2021)
15. Jeon, J., Kim, J., Lee, K., Oh, S., Ok, J.: Gradient inversion with generative image prior. In: Ranzato, M., Beygelzimer, A., Dauphin, Y.N., Liang, P., Vaughan, J.W. (eds.) Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, 6–14 December 2021, Virtual, pp. 29898–29908 (2021)
16. Jin, X., Chen, P.-Y., Hsu, C.-Y., Chia-Mu, Yu., Chen, T.: CAFE: catastrophic data leakage in vertical federated learning. In: Advances in Neural Information Processing Systems, vol. 34, pp. 994–1006 (2021)
17. Li, D., Xie, W., Wang, Z., Yibing, L., Li, Y., Fang, L.: FedDiff: diffusion model driven federated learning for multi-modal and multi-clients. IEEE Trans. Circuits Syst. Video Technol. **34**(10), 10353–10367 (2024)
18. Li, Z., Zhang, J., Liu, L., Liu, J.: Auditing privacy defenses in federated learning via generative gradient leakage. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, 18–24 June 2022, pp. 10122–10132. IEEE (2022)
19. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: 2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, 7–13 December 2015, pp. 3730–3738. IEEE Computer Society (2015)
20. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, 18–24 June 2022, pp. 10674–10685. IEEE (2022)
21. Saharia, C., et al.: Photorealistic text-to-image diffusion models with deep language understanding. In: Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, 28 November–9 December 2022 (2022)
22. Shankar, A., Brouwer, H., Hai, R., Chen, L.Y.: SiloFuse: cross-silo synthetic data generation with latent tabular diffusion models. CoRR, abs/2404.03299 (2024)
23. Somepalli, G., Singla, V., Goldblum, M., Geiping, J., Goldstein, T.: Diffusion art or digital forgery? Investigating data replication in diffusion models. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, 17–24 June 2023, pp. 6048–6058. IEEE (2023)
24. Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, 3–7 May 2021. OpenReview.net (2021)

25. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
26. Sun, J., Li, A., Wang, B., Yang, H., Li, H., Chen, Y.: Soteria: provable defense against privacy leakage in federated learning from representation perspective. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, Virtual, 19–25 June 2021, pp. 9311–9319. Computer Vision Foundation/IEEE (2021)
27. Sun, X., Ren, X., Ma, S., Wang, H.: meProp: sparsified back propagation for accelerated deep learning with reduced overfitting. In: Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6–11 August 2017, Proceedings of Machine Learning Research, vol. 70, pp. 3299–3308. PMLR (2017)
28. Voigt, P., Von dem Bussche, A.: The EU General Data Protection Regulation (GDPR). A Practical Guide, 1st edn. Springer, Cham (2017). 10(3152676):10–5555
29. Wang, Z., Wang, J., Liu, Z., Qiu, Q.: Binary latent diffusion. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, 17–24 June 2023, pp. 22576–22585. IEEE (2023)
30. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)
31. Wu, R., Chen, X., Guo, C., Weinberger, K.Q.: Learning to invert: simple adaptive attacks for gradient inversion in federated learning. In: Evans, R.J., Shpitser, I. (eds.) Uncertainty in Artificial Intelligence, UAI 2023, 31 July–4 August 2023, Pittsburgh, PA, USA. Proceedings of Machine Learning Research, vol. 216, pp. 2293–2303. PMLR (2023)
32. Wu, Y., Yu, N., Li, Z., Backes, M., Zhang, Y.: Membership inference attacks against text-to-image generation models. CoRR, abs/2210.00968 (2022)
33. Xia, W., Zhang, Y., Yang, Y., Xue, J.-H., Zhou, B., Yang, M.-H.: GAN inversion: a survey. IEEE Trans. Pattern Anal. Mach. Intell. **45**(3), 3121–3138 (2023)
34. Xu, J., Hong, C., Huang, J., Chen, L.Y., Decouchant, J.: AGIC: approximate gradient inversion attack on federated learning. In: 41st International Symposium on Reliable Distributed Systems, SRDS 2022, Vienna, Austria, 19–22 September 2022, pp. 12–22. IEEE (2022)
35. Ye, X., et al.: Accelerating CNN training by pruning activation gradients. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12370, pp. 322–338. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58595-2_20
36. Yin, H., Mallya, A., Vahdat, A., Álvarez, J.M., Kautz, J., Molchanov, P.: See through gradients: image batch recovery via Gradinversion. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, Virtual, 19–25 June 2021, pp. 16337–16346. Computer Vision Foundation/IEEE (2021)
37. Yu, F., Zhang, Y., Song, S., Seff, A., Xiao, J.: LSUN: construction of a large-scale image dataset using deep learning with humans in the loop. CoRR, abs/1506.03365 (2015)
38. Zhang, H., Cissé, M., Dauphin, Y.N., Lopez-Paz., D.: mixup: beyond empirical risk minimization. In: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, 30 April–3 May 2018, Conference Track Proceedings. OpenReview.net (2018)
39. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: 2018 IEEE Conference on

Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, 18–22 June 2018, pp. 586–595. Computer Vision Foundation/IEEE Computer Society (2018)

40. Zhao, B., Mopuri, K.R., Bilen, H.: iDLG: improved deep leakage from gradients. CoRR, abs/2001.02610 (2020)
41. Zhu, L., Liu, Z., Han, S.: Deep leakage from gradients. In: Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8–14 December 2019, Vancouver, BC, Canada, pp. 14747–14756 (2019)