Unveiling Privacy Risks in Stochastic Neural Networks Training: Effective Image Reconstruction from Gradients

Yiming Chen[1,2] ✉, Xiangyu Yang[1,2], and Nikos Deligiannis[1,2]

[1] ETRO Department, Vrije Universiteit Brussel, Brussels, Belgium
[2] Interuniversitair Micro-electronica Centrum, Leuven, Belgium
{cyiming, xyanga, ndeligia}@etrovub.be

Abstract. Federated Learning (FL) provides a framework for collaborative training of deep learning models while preserving data privacy by avoiding sharing the training data. However, recent studies have shown that a malicious server can reconstruct training data from the shared gradients of traditional neural networks (NNs) in FL, via Gradient Inversion Attacks (GIAs) that emulate the client's training process. Contrary to earlier beliefs that Stochastic Neural Networks (SNNs) are immune to such attacks due to their stochastic nature (which makes the training process challenging to mimic), our findings reveal that SNNs are equally susceptible to GIAs as SNN gradients contain the information of stochastic components, allowing attackers to reconstruct and disclose those uncertain components. In this work, we play the role of an attacker and propose a novel attack method, named *Inverting Stochasticity from Gradients* (ISG), that can successfully reconstruct the training data by formulating the stochastic training process of SNNs as a variant of the traditional NN training process. Furthermore, to improve the fidelity of the reconstructed data, we introduce a feature constraint strategy. Extensive experiments validate the effectiveness of our GIA and suggest that perturbation-based defenses in forward propagation, such as using SNNs, fail to secure models against GIAs inherently.

Keywords: Federated Learning · Gradient Inversion Attacks

1 Introduction

Federated Learning (FL) [22] is a promising decentralized framework for training deep learning models collaboratively to address data governance and privacy challenges. It is applicable to a range of fields, such as healthcare [31] and the Internet of Things (IoT) [23]. Unlike traditional centralized training that collects all training data on a central server, FL eliminates the need to gather data centrally. Initially, a global model is distributed to clients by the server. Each client then locally uses their private data to compute gradients, which are subsequently aggregated by the server to update the global model. It ensures that only gradients rather than sensitive data are shared with the server, thereby preventing any risk of information leakage during model training.

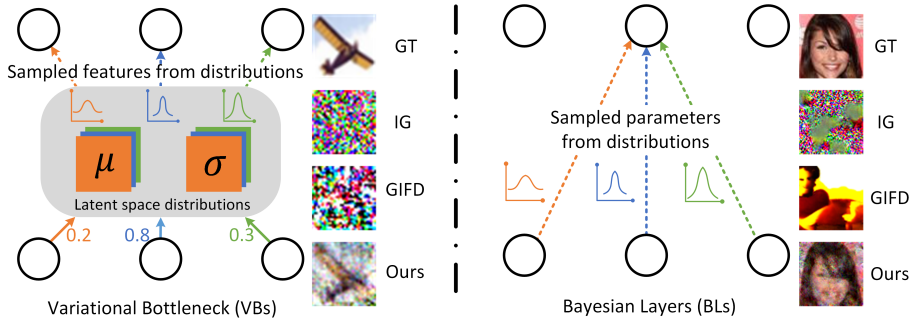

Fig. 1: The architecture of VBs and BLs (see Sec. 3 for more details), alongside the reconstructed images of the CIFAR-10 and CelebA images using IG [9], GIFD [7], and our proposed ISG attack. Circles represent neurons in corresponding neural networks.

However, recent studies have shown that FL is not always resistant to information leakage. In particular, by seeing through the shared gradients, a malicious server could potentially compromise privacy by deducing sensitive attributes (e.g., gender, age) of clients [8]. More seriously, Gradient Inversion Attacks (GIAs) can infer maximal information about clients by directly reconstructing the training data from gradients [9, 32, 35]. It involves initializing dummy data, calculating the corresponding dummy gradients to simulate the training process, and iteratively optimizing the dummy data by minimizing the discrepancy between the true and dummy gradients, eventually leading the dummy data to approximate the original data closely. To protect against GIAs, some works explored defenses by perturbing the gradients, including clipping [27, 29] and noise addition [35]. Although high-level perturbations can effectively defend against privacy breaches, they significantly impact the model performance [5, 10, 35]. To address this, other studies [26] have investigated leveraging the intrinsic properties of Stochastic Neural Networks (SNNs), specifically Variational Bottlenecks (VBs) [1, 15], to counteract GIAs. SNNs also comprise Bayesian Layers (BLs) [3]. As illustrated in Fig. 1, VBs map input to distributions instead of deterministic points, and BLs treat each parameter not as a single-fixed-value but as a distribution over possible values. In each forward propagation, SNNs randomly sample features or parameters from these distributions, leading to the fact that given the same input, the output and gradients are always distinctive. The unique, non-shared stochastic elements within SNNs prevent the server from accurately emulating the training process, providing an effective defense against GIAs [26].

In this work, we show that SNNs are as vulnerable to GIAs as traditional (deterministic) deep learning networks. Our findings indicate that the gradients of SNNs contain the information of both training data and stochastic elements (including both the re-sampled features and model parameters) used in local training, allowing attackers to closely approximate those stochastic elements through the gradients. This enables the malicious server significantly reduces the stochasticity of SNNs when simulating the local training process. From an attacker's

standpoint, we propose a novel attack method termed *Inverting Stochasticity from Gradients* (ISG), which can successfully reconstruct the training data from the gradients of SNNs. Fig. 2 illustrates the overview of our ISG attack. Additionally, we investigate the issue of local minima in GIAs, attributed to convolutional kernels. To address this, we introduce a novel component, namely the feature correction multiplier, to constrain the optimization space, thereby improving the fidelity of reconstructed images. Furthermore, we demonstrate that defense mechanisms relying on stochasticity (perturbation) during forward propagation are ineffective in protecting models against GIAs.

Our main contributions are summarized as follows:

- We evaluate the defense effectiveness of two types of SNNs against state-of-the-art GIAs: VB-based SNNs for feature stochasticity and BL-based SNNs for model parameter stochasticity;
- We propose a novel attack to bypass the defense of SNNs. To the best of our knowledge, we are the first to systemically reconstruct the training data from the gradients of VB- and BL-based SNNs;
- We study the local minima in GIAs caused by convolutional kernels and introduce a feature constraint strategy to achieve higher fidelity in the recovered images;
- We conduct extensive experiments to validate our approach, which shows the superior performance of our attack over current state-of-the-art GIAs.

2 Related Work

Gradient Inversion Attacks. In an FL system, where multiple edge clients collaborate under the coordination of a potentially malicious server to train a model $\mathcal{F}(\theta;\cdot)$ with parameters θ and a given loss function $\mathcal{L}(\cdot)$, GIAs [7,9,20,32,34,35] enable the server to reconstruct the original training image x from the gradients ∇_θ shared by clients. The attack mechanism involves initializing a dummy image $\hat{x}$ and simulating the training process to generate corresponding dummy gradients $\nabla\mathcal{L}(\mathcal{F}(\theta;\hat{x}),y)$ with y being the label that extracted from gradients [32,34]. Through iterative optimization aimed at minimizing the discrepancy, quantified by a distance metric $\mathcal{D}(\cdot)$ (such as Euclidean [35] and Cosine Similarity [9] distance), between ∇_θ and $\nabla\mathcal{L}(\mathcal{F}(\theta;\hat{x}),y)$, the dummy image $\hat{x}$ is refined to approximate the original image x closely. This can be formulated as follows:

$$\hat{x} = \underset{\hat{x}}{\operatorname{argmin}}\, \mathcal{D}(\nabla\mathcal{L}(\mathcal{F}(\theta;\hat{x}),y),\nabla_\theta) + \mathcal{R}_{\texttt{img}}(\hat{x}), \tag{1}$$

where $\mathcal{R}_{\texttt{img}}(\cdot)$ is the regularization function to improve the fidelity of $\hat{x}$, such as TV [9] and GAN [7,20]. However, the efficacy of GIAs depends on mimicking the forward and backward propagation processes occurring at edge clients. When these processes become stochastic, they present a challenge to server emulation, leading to the server's failure in successfully recovering the training data [26,30].

Stochastic Neural Networks. Stochastic Neural Networks (SNNs) include the integration of Bayesian layers (BLs) [3] and Variational Bottlenecks (VBs) [1,15].

VBs are known for their enhanced security features, particularly their effectiveness in defending against GIAs [26]. BLs treat each parameter not as a single-fixed-value but as a distribution over possible values. VBs map input features to distributions instead of deterministic points. The properties of SNNs appear in that parameters or data representations are newly sampled from their respective distributions in each forward propagation. This results in diverse outputs and gradients for the same input data. This stochasticity in SNNs fools GIAs by disrupting attempts to replicate the training process at local clients [26, 30], thereby safeguarding against potential data leakage. Although the work of [2] tried to recover training data from gradients in VB-based models, their approach only applies to fully connected models with bias terms and not to mini-batch training, making it less applicable in practice. In contrast, convolutional SNNs, more prevalent in practice, continue to show robustness against GIAs.

3 Problem Formulation

Consider a generic SNN for image classification tasks with parameters θ. Given a batch of images x and labels y, the predicted class probabilities across L categories for each image is defined by $\mathcal{F}(\theta; \cdot) : x \in \mathbb{R}^{B\times C\times H\times W} \rightarrow \mathbb{R}^{B\times L}$, with B, C, H, W being the batch size, the number of channels, height and width, respectively. The SNN, as illustrated in Fig. 2, includes a feature extractor, along with an FC layer, a VB [15, 26], and a BL [3] (Fig. 1 for visual details of VBs and BLs). Notably, in practical applications, a model supports flexible module placement, allowing for multiple or no specific modules anywhere in the network.

The feature extractor of parameter $\theta_{\texttt{ext}}$ is responsible for learning and extracting task-relevant features $x_{\texttt{ext}} \in \mathbb{R}^{B\times c\times h\times w}$ from the raw input data x with c, h, w being the number of features, their height and width:

$$x_{\texttt{ext}} = f(\theta_{\texttt{ext}}; x). \tag{2}$$

VB comprises a probabilistic encoder and a decoder, with a prior distribution $p(z_{\texttt{vb}})$ of latent variable $z_{\texttt{vb}}$, typically assumed to follow the standard normal distribution [1, 15]. Given an input $x_{\texttt{vb}}$, VB first produces the mean $\mu_{\texttt{vb}}$ and standard deviation $\sigma_{\texttt{vb}}$ standing for multivariate Gaussian distributions to approximate the latent space distribution $p(z_{\texttt{vb}})$ as follows:

$$\mu_{\texttt{vb}}, \sigma_{\texttt{vb}} = f(\theta_{\texttt{enc}}; x_{\texttt{vb}}), \tag{3}$$

where $\theta_{\texttt{enc}}$ denotes the parameters of VB's posterior estimator (encoder). Then, the process of sampling latent features $z_{\texttt{vb}}$ from the posterior distribution $z_{\texttt{vb}} \sim \mathcal{N}(\mu_{\texttt{vb}}, \sigma^2_{\texttt{vb}}\mathbf{I})$ is realized by using the reparameterization trick [1] via intermediate noise $\varepsilon_{\texttt{vb}}$ sampled from a standard normal distribution, that is:

$$\varepsilon_{\texttt{vb}} \sim \mathcal{N}(0, \mathbf{I}), \tag{4}$$

$$z_{\texttt{vb}} = \mu_{\texttt{vb}} + \sigma_{\texttt{vb}} \odot \varepsilon_{\texttt{vb}}, \tag{5}$$

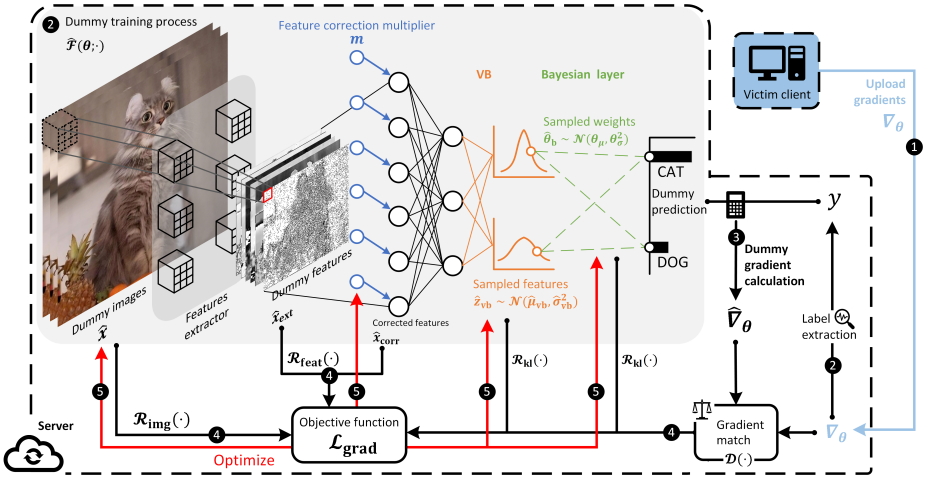

Fig. 2: Our proposed ISG attack starts by randomly initializing $\hat{x}$, m, $\hat{\varepsilon}_{\mathtt{vb}}$, and $\hat{\varepsilon}_{\mathtt{b}}$ (used for constructing $\hat{z}_{\mathtt{vb}}, \hat{\theta}_{\mathtt{b}}$). The process unfolds as follows: a malicious server 1) receives the client's gradients, which is computed using private training images; 2) extracts the label from the gradients (Sec. 4.4) and simulates the client's forward propagation using $\hat{x}$, m, $\hat{\varepsilon}_{\mathtt{vb}}$, and $\hat{\varepsilon}_{\mathtt{b}}$; 3) calculates the dummy gradients per Eq. (9b); 4) computes the ISG attack loss as Eq. (10), and 5) jointly optimizes $\hat{x}$, m, $\hat{\varepsilon}_{\mathtt{vb}}$, $\hat{\varepsilon}_{\mathtt{b}}$ with respect to the attack loss until the algorithm converges.

where $\varepsilon_{\mathtt{vb}}, \mu_{\mathtt{vb}}, \sigma_{\mathtt{vb}}$, $z_{\mathtt{vb}}$ are in $\mathbb{R}^{B\times c_{\mathtt{vb}}\times h_{\mathtt{vb}}\times w_{\mathtt{vb}}}$ with $c_{\mathtt{vb}}, h_{\mathtt{vb}}, w_{\mathtt{vb}}$ being the number of latent features, their height and width; $\odot$ and $\mathbf{I}$ denote the element-wise product and the identity matrix, respectively. The sampled features $z_{\mathtt{vb}}$ is then sent to the decoder to generate new features $y_{\mathtt{vb}} \sim p(y_{\mathtt{vb}}|z_{\mathtt{vb}})$, as follows:

$$y_{\mathtt{vb}} = f(\theta_{\mathtt{dec}}; z_{\mathtt{vb}}), \tag{6}$$

with $\theta_{\mathtt{dec}}$ being the parameters of decoder.

BLs maintain a set of means θ_μ and deviations θ_σ as learnable parameters that define the posterior distribution of model parameters. Unlike traditional models where parameters are deterministic, the BL samples a new set of parameters $\theta_{\mathtt{b}}$ by the reparameterization trick from the posterior distribution $\mathcal{N}(\theta_\mu, \theta_\sigma^2\mathbf{I})$ for each forward propagation. Thus, the output $y_{\mathtt{b}}$ of a BL for a given input $x_{\mathtt{b}}$ can be expressed as follows:

$$\varepsilon_{\mathtt{b}} \sim \mathcal{N}(0, \mathbf{I}), \tag{7a}$$
$$\theta_{\mathtt{b}} = \theta_\mu + \theta_\sigma \odot \varepsilon_{\mathtt{b}}, \tag{7b}$$
$$y_{\mathtt{b}} = f(\theta_{\mathtt{b}}; x_{\mathtt{b}}), \tag{7c}$$

where $\varepsilon_{\mathtt{b}}, \theta_\mu, \theta_\sigma, \theta_{\mathtt{b}}$ are in $\mathbb{R}^{h_{\mathtt{b}}\times w_{\mathtt{b}}}$ ($h_{\mathtt{b}}$ and $w_{\mathtt{b}}$ being the height and width), and $\varepsilon_{\mathtt{b}}$ is the intermediate noise sampled from a standard normal distribution.

Overall, the gradients ∇_θ of this SNN is derived as follows:

$$\nabla_\theta = \nabla\mathcal{L}(\mathcal{F}(\theta; x), y), \tag{8}$$

where $\mathcal{L}(\cdot)$ is the loss function for classification tasks, usually cross-entropy loss, incorporates the Kullback-Leibler divergence (KLD) between the posterior and priors distributions for VBs and BLs [3, 15], which ensures the posteriors are closely aligned with the assumed prior distributions.

Therefore, consider an FL system, in which each client computes gradients ∇_θ using its private training data x, y, alongside the SNN $\mathcal{F}(\theta; \cdot)$ and the loss function $\mathcal{L}(\cdot)$ specified by the server. These true gradients are then transmitted to a curious but honest server. The objective is to produce a set of synthetic images $\hat{x}$ that closely resemble the original training images x by utilizing the gradients ∇_θ known at the server. This attack must be achieved without any knowledge of the sampled features $z_{\texttt{vb}}$ or the parameters $\theta_{\texttt{b}}$, as they are discarded by clients after each iteration and are not shared with the server.

4 Methodology

We begin by formulating our optimization objective of ISG attack as follows:

$$\hat{x} = \underset{\hat{x}, m, \hat{\varepsilon}_{\texttt{vb}}, \hat{\varepsilon}_{\texttt{b}}}{\operatorname{argmin}} \mathcal{D}(\hat{\nabla}_\theta, \nabla_\theta) + \mathcal{R}_{\texttt{img}}(\hat{x}) + \mathcal{R}_{\texttt{feat}}(\theta; \hat{x}, m) + \mathcal{R}_{\texttt{kl}}(\hat{\varepsilon}_{\texttt{vb}}) + \mathcal{R}_{\texttt{kl}}(\hat{\varepsilon}_{\texttt{b}}), \tag{9a}$$

$$\hat{\nabla}_\theta = \nabla\mathcal{L}(\hat{\mathcal{F}}(\theta; \hat{x}, m, \hat{\varepsilon}_{\texttt{vb}}, \hat{\varepsilon}_{\texttt{b}}), y), \tag{9b}$$

where $\hat{\nabla}_\theta$ is the dummy gradients and $\hat{\mathcal{F}}(\theta; \cdot)$ represents our defined forward-propagation for mimicking the training process. This process involves the dummy images $\hat{x}$, the feature correction multiplier m, the dummy intermediate noise $\hat{\varepsilon}_{\texttt{vb}}$ and $\hat{\varepsilon}_{\texttt{b}}$. They are jointly optimized together. $\hat{\varepsilon}_{\texttt{vb}}$ and $\hat{\varepsilon}_{\texttt{b}}$ are learned to construct the dummy sampled features $\hat{z}_{\texttt{vb}}$ and parameters $\hat{\theta}_{\texttt{b}}$ as the approximation of $z_{\texttt{vb}}$ and $\theta_{\texttt{b}}$ in VB and BL, respectively. $\mathcal{R}_{\texttt{img}}(\cdot)$, $\mathcal{R}_{\texttt{feat}}(\cdot)$, and $\mathcal{R}_{\texttt{kl}}(\cdot)$ denote the regularization functions for improving the reconstruction quality. The distance measurement $\mathcal{D}(\cdot)$ is based on the MSE loss [32, 35]. For simplicity, we denote the objective function as follows:

$$\hat{x} = \underset{\hat{x}, m, \hat{\varepsilon}_{\texttt{vb}}, \hat{\varepsilon}_{\texttt{b}}}{\operatorname{argmin}} \mathcal{L}_{\texttt{grad}}(\hat{x}, m, \hat{\varepsilon}_{\texttt{vb}}, \hat{\varepsilon}_{\texttt{b}}), \tag{10}$$

where $\mathcal{L}_{\text{grad}}(\hat{x}, m, \hat{\varepsilon}_{\texttt{vb}}, \hat{\varepsilon}_{\texttt{b}})$ encapsulates the objective function in (9a). Fig. 2 depicts the overview of our proposed attack, and we provide detailed explanations for each component in the subsequent subsections.

4.1 Feature Constraint Strategy

When using GIAs to recover training data from gradients in traditional CNNs, we find that the optimization process tends to stall in local minima, unlike in

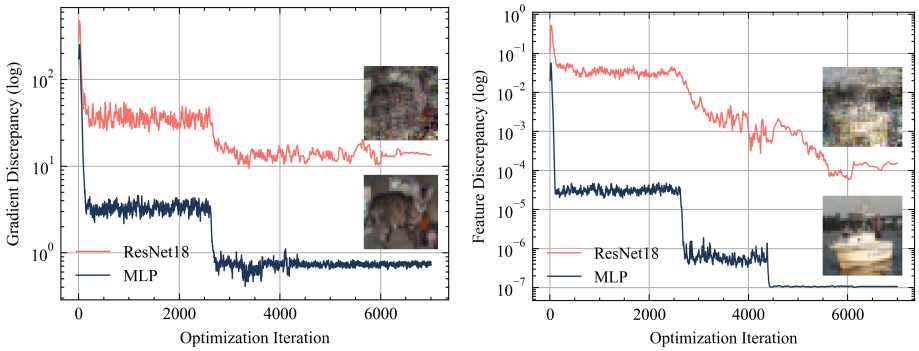

Fig. 3: The image reconstruction by using the gradients from the MLP and ResNet-18. a) The MSE between the true and dummy gradients of the feature extractor, and b) The MSE between the true and dummy features, as the optimization iterations progress.

MLPs. We apply a well-established GIA, named IG [9], to a three-layer MLP (with 1024 hidden neurons) [26] and a ResNet-18 [11] model, each concluding with an FC layer for prediction, to reconstruct the CIFAR-10 [16] data from their gradients. Remarkably, ResNet-18 has over 11 million parameters within its convolutional feature extractor, vastly outnumbering the MLP's mere 4.2 million parameters, which suggests that the extensive gradient count of ResNet-18 could lead to more significant information leakage [9]. However, surprisingly, the images reconstructed from ResNet-18 underperform significantly in visual quality compared to those from MLPs, as depicted in Fig. 3. We further analyze the MSE between the true and dummy gradients of the feature extractor (Fig. 3a), as well as between the true and dummy extracted features (Fig. 3b) across optimization iterations in GIAs. The dummy extracted features $\hat{x}_{\texttt{ext}}$ is calculated per Eq. (2) by using the dummy images $\hat{x}$. According to the results, ResNet-18 consistently demonstrates a higher MSE—almost $10\times$ for dummy gradients and a $1000\times$ for dummy features—when compared to the MLP. This indicates that the shared weights within convolutional kernels may trap the optimization in local minima, thereby hindering the accurate reconstruction of data from gradients.

To mitigate the issue of local minima entrapment and maximize the utility of gradient information for detailed reconstruction of training images, we propose a novel component: a learnable feature correction multiplier $m \in \mathbb{R}^{B\times c\times h\times w}$. This multiplier is positioned after the convolutional feature extractor to correct the dummy features $\hat{x}_{\texttt{ext}}$. The corrected features $\hat{x}_{\texttt{corr}}$ is defined as the element-wise product between the multiplier m and the dummy features $\hat{x}_{\texttt{ext}}$, that is:

$$\hat{x}_{\texttt{corr}} = m \odot \hat{x}_{\texttt{ext}} = m \odot f(\theta_{\texttt{ext}}; \hat{x}). \tag{11}$$

The corrected features are then used for predictions and gradients calculation by the remaining FC layers. Given that m is the partial input to the FC layers, inde-

pendent of the convolutional kernels, and previous discussions have showed that the input of FC layers can be accurately inferred from the gradient, this enables m as a well-estimated discrepancy in the extracted features. Thus, $\hat{x}_{\texttt{corr}}$ rather than $\hat{x}_{\texttt{ext}}$, provides a closer approximation of the true features, $x_{\texttt{ext}}$. Therefore, to better align the dummy features with the true features, we introduce an $l2$ regularization term between the dummy and corrected features as follows:

$$\mathcal{R}_{\texttt{feat}}(\theta; \hat{x}, m) = \alpha_{\texttt{corr}} ||f(\theta_{\texttt{ext}}; \hat{x}) \odot (1 - m)||_2 = \alpha_{\texttt{corr}} ||\hat{x}_{\texttt{ext}} - \hat{x}_{\texttt{corr}}||_2, \quad (12)$$

where $\alpha_{\texttt{corr}}$ is the scale factor. By adding the feature regularization, we impose a constraint on the dummy images $\hat{x}$, effectively narrowing the optimization space. This leads to the achievement of more accurately reconstructed images.

4.2 Inverting Stochasticity

We argue that SNN gradients implicitly retain the information of not only training data but also stochastic elements like randomly sampled features and parameters. This implies that reconstructing stochastic elements alongside the training data from gradients is feasible. By approximating stochastic elements, the attacker can spot the "stochastic" process utilized during training, allowing the recovered stochastic features and parameters to be used in the dummy forward propagation to emulate the client's training process. Consequently, the problem simplifies to reconstructing training data from gradients in a traditional neural network, where stochasticity is no longer a factor.

Specifically, let $\hat{\mu}_{\texttt{vb}}$ and $\hat{\sigma}_{\texttt{vb}}$ denote the mean and standard derivation of the dummy latent space posterior distribution of VB by using dummy training images $\hat{x}$ per Eq. (3). Instead of directly sampling features from the dummy posterior distribution as Eq. (4), a jointly learnable noise $\hat{\varepsilon}_{\texttt{vb}} \in \mathbb{R}^{B \times c_{\texttt{vb}} \times h_{\texttt{vb}} \times w_{\texttt{vb}}}$ is utilized to construct the dummy sampled features $\hat{z}_{\texttt{vb}}$, that is:

$$\hat{z}_{\texttt{vb}} = \hat{\mu}_{\texttt{vb}} + \hat{\sigma}_{\texttt{vb}} \odot \hat{\varepsilon}_{\texttt{vb}}, \quad (13)$$

and the output of VB is calculated by Eq. (6) using $\hat{z}_{\texttt{vb}}$.

In parallel, for BLs, instead of sampling parameters directly per Eq. (7a), another learnable noise $\hat{\varepsilon}_{\texttt{b}} \in \mathbb{R}^{h_{\texttt{b}} \times w_{\texttt{b}}}$ is used to form the dummy parameters $\hat{\theta}_{\texttt{b}}$:

$$\hat{\theta}_{\texttt{b}} = \theta_{\mu} + \theta_{\sigma} \odot \hat{\varepsilon}_{\texttt{b}}, \quad (14)$$

and the forward propagation of BLs uses $\hat{\theta}_{\texttt{b}}$ as parameters by following Eq. (7c).

Optimizing solely with gradient may result in $\hat{\varepsilon}_{\texttt{vb}}$ and $\hat{\varepsilon}_{\texttt{b}}$ deviating from the standard Gaussian distribution. This deviation can lead to the generation of unrealistic images with artifacts. To mitigate this issue, we incorporate the KLD between $\hat{\varepsilon}_{\texttt{vb}}$, $\hat{\varepsilon}_{\texttt{b}}$, and the standard Gaussian distribution to regularize their distributions [10, 15]. Thus, $\mathcal{R}_{\texttt{kl}}(\cdot)$ is defined as follows:

$$\mathcal{R}_{\texttt{kl}}(\varepsilon) = \alpha_{\texttt{kl}} \frac{1}{2} (-\log \texttt{std}^2(\varepsilon) + \texttt{std}^2(\varepsilon) + \texttt{mean}^2(\varepsilon) - 1), \quad (15)$$

where $\texttt{mean}(\cdot)$ and $\texttt{std}(\cdot)$ are the functions of calculating the mean and standard deviation, respectively, with $\alpha_{\texttt{kl}}$ serving as the scale factor.

4.3 Image Regularization

To generate more realistic images, we follow the works of [9,32] to incorporate an image regularization scaled by factor $\alpha_{\texttt{img}}$ during the optimization. Specifically, since the pattern gradually changes in natural images, Total Variation (TV) [25] is introduced to enforce neighbor-wise smoothness of dummy images $\hat{x}$ [9], and the $l2$ norm of dummy images $\hat{x}$ is also applied to penalize large pixel values [32]. Therefore, image regularization can be formulated as follows:

$$\mathcal{R}_{\texttt{img}}(\hat{x}) = \alpha_{\texttt{img}} TV(\hat{x}) + \alpha_{\texttt{img}} ||\hat{x}||_2. \tag{16}$$

4.4 Label Extraction

Consider the model is generally trained with cross-entropy loss using one-hot labels, and the vector ∇W^i_{pred} represents the gradient of weights W^i_{pred} that is connected to the i^{th} logit in the prediction (last) layer. The ground-truth label y can be analytically identified as checking the sign of ∇W^i_{pred} in single image training [34], as follows:

$$y = i, \quad \text{s.t.} \quad {\nabla W^i_{\text{pred}}}^{\text{T}} \cdot \nabla W^j_{\text{pred}} \leq 0, \quad \forall j \neq i, \tag{17}$$

The study of [32] extends this concept to batch label restoration, demonstrating that utilizing extracted labels, as opposed to learning dummy labels, enhances the stability and efficiency of training data reconstruction.

5 Experiments

5.1 Experimental Setup

FL Tasks and Datasets. We evaluate our ISG attack on four widely discussed FL tasks [9, 19, 20, 26]. (1) 10-class image classification on the CIFAR-10 dataset [16] with 50,000 training images of size 32×32; (2) 100-class image classification on the CIFAR-100 dataset [16] with 50,000 training images of size 32×32; (3) Gender classification on the CelebA dataset [21] with 23,999 images of facial attributes of size 64×64, and (4) 200-class image classification on the Tiny-ImageNet dataset [18] with 100,000 training images of size 64×64. Adhering to FL principles [12, 22], attackers lack prior knowledge of the dataset, including the distribution of training data. We randomly sample 128 training images for each task and calculate the model gradients via a single local update.
Victim Models. We adopt four extensively evaluated models [9, 26] in FL as backbones: (1) ResNet-18 [11], (2) ResNet-34 [11], (3) ConvNet [9], and (4) a 3-layer MLP with 1024 hidden neurons [26]. In VB-based models, a VB with 256 latent dimensions is inserted between the feature extractor and the FC layers—or the final layer in the case of MLPs [26]. In models integrated with BLs, a BL featuring an input channel dimension of 256, with both weights and biases as distributions [15], is appended to the final FC layer to make confidence predictions. All models are randomly initialized.

Implementation. The experiments are performed on a single server with two GeForce RTX 3090 GPUs. Our method leverages the PyTorch library [24], building upon the repository of [9]. Our approach starts the attack by randomly initializing $\hat{x}$, m, $\hat{\varepsilon}_{\texttt{vb}}$, and $\hat{\varepsilon}_{\texttt{b}}$, following standard normal distributions. The hyperparameters $\alpha_{\texttt{corr}}$, $\alpha_{\texttt{kl}}$, and $\alpha_{\texttt{img}}$ are configured to 1, 1e-8, and 1e-6, respectively. The Adam [14] algorithm is adopted to optimize $\hat{x}$, m, $\hat{\varepsilon}_{\texttt{vb}}$, and $\hat{\varepsilon}_{\texttt{b}}$ for 15,000 iterations. The initial learning rate is set to 0.1, which is reduced to every 10% of its value at the $\frac{3}{8}$, $\frac{5}{8}$, and $\frac{7}{8}$ milestones of the total iteration count. Our attack algorithm does not leverage any additional information, such as BN statistics [32] or a GAN pre-trained on IID data [7, 20]. The source code of our work will be made available to the public on GitHub[3].

Evaluation Metrics. To quantitatively assess the quality of image reconstructions, we employ four metrics to compare original images with their reconstructions: (1) *Mean Squared Error* (MSE↓), which calculates the average squared discrepancy between the pixel values of two images; (2) *Learned Perceptual Image Patch Similarity* (LPIPS↓) [33], which measures the similarity of features extracted from two images using a pre-trained Alexnet [17], with values ranging from 0 to 1; (3) *Peak Signal-to-Noise Ratio* (PSNR↑), the ratio of the maximum possible power of an image to the power of corrupting noise, expressed in decibels (dB); and (4) *Structural Similarity Index Measure* (SSIM↑) [28], which computes the similarity between two images based on structural information, luminance, and contrast variations, with values ranging from -1 to 1. To evaluate the quality of the reconstructed stochastic elements, namely the dummy stochastic features and parameters in VB and BLs, we consider two metrics, including (1) MSE↓: the mean squared error between the stochastic elements sampled by the client and those recovered by the attacker; and (2) *Signal-to-Noise Ratio* (SNR↑): the ratio of signal power to noise power. The symbols ↓ and ↑ denote that, for the respective metric, a lower or higher value indicates a better quality of the reconstructed image (in other words, a more effective attack).

5.2 Comparison against State-of-the-Art GIAs

GIAs baseline. We compare our proposed ISG attack against several state-of-the-art GIAs, including (1) *Deep Leakage from Gradient* (DLG) [35]: employs MSE as the gradient matching loss and utilizes the L-BFGS optimizer for 1,200 iterations; (2) *improved Deep Leakage from Gradient* (iDLG) [34]: integrates the label extraction to DLG; (3) *Inverting Gradient* (IG) [9]: adopts a negative cosine similarity measure of gradients with TV regularization. The Adam optimizer is used to optimize for 24,000 iterations. The results report the best outcome from four trials with different random seeds; (4) *Gradient Inversion* (GI) [32]: leverages MSE with the knowledge of BatchNorm [13] statistics. We report the best reconstruction of four restarts as well; (5) *Generative Gradient Leakage* (GGL) [20]: uses a BigGAN [4] (pre-trained on ImageNet [6]) as the image prior

[3] https://github.com/SillyPuffo/ISG

and optimizes the GAN's latent code via the Adam optimizer for 2,500 iterations, and (6) *Generative Gradient Leakage with Feature Domain Optimization* (GIFD) [7]: optimizes the features of the first 13 layers in BigGAN, conducting 1,000 iterations of optimization for each layer.

Experimental Results. Tab. 1 presents the quality assessments of reconstructed images $\hat{x}$ and reconstructed stochastic elements. For the CIFAR-10 classification task, we evaluate the quality of images and reconstructed features $\hat{z}_{\mathtt{vb}}$, derived from the gradients of a VB-ResNet18 model. Concurrently, for the gender classification FL task, we report the quality of reconstructed CelebA images and the reconstructed model parameters $\hat{\theta}_{\mathtt{b}}$, obtained from the gradients of a Bayesian ConvNet model. Bold numbers denote the best performance of attacks. These results show that existing GIAs fail to reconstruct training data through the gradients of SNNs due to their inability to mimic forward propagation. In contrast, our ISG attack successfully recovers the training data by effectively reconstructing the stochastic elements, thus circumventing the defenses established by either VBs or BLs. Specifically, our ISG attack improves SSIM by approximately 5× on VB-ResNet18 and reduces MSE of reconstructed images by nearly 20× on Bayesian ConvNet. Fig. 4 depicts a comparative visualization of original and reconstructed images for the CIFAR-10 and CelebA datasets. The images reconstructed by existing attacks appear predominantly as noise, lacking discernible visual information. In contrast, images recovered through our ISG attack preserve significant visual information, revealing considerable privacy of the clients.

Table 1: Comparative Reconstruction Metrics for VB-ResNet18 and Bayesian ConvNet in Image Classification on CIFAR-10 and Gender Classification on CelebA.

Dataset	Model	Methods	Reconstructed Images $\hat{x}$				$\hat{z}_{\mathtt{vb}}$ or $\hat{\theta}_{\mathtt{b}}$	
			MSE ↓	LPIPS ↓	PSNR ↑	SSIM ↑	MSE ↓	SNR ↑
CIFAR10	VB-ResNet18	DLG [35]	0.2390	0.4051	3.1322	0.0059	-	-
		iDLG [34]	0.1078	0.3336	4.9769	0.0299	-	-
		IG [9]	0.2275	0.4084	3.2456	0.0163	-	-
		GI [32]	0.1057	0.3471	4.9973	0.0341	-	-
		GGL [20]	0.2160	0.6475	6.7497	0.1322	-	-
		GIFD [7]	0.2376	0.6972	6.2816	0.0474	-	-
		Ours	**0.0233**	**0.0632**	**8.4871**	**0.5831**	**0.0028**	**26.81**
CelebA	BL-ConvNet	DLG [35]	0.2102	0.9763	3.4137	0.0119	-	-
		iDLG [34]	0.2404	0.9402	3.1118	0.0049	-	-
		IG [9]	0.2267	0.8926	3.2541	0.0112	-	-
		GI [32]	0.2223	0.9301	3.2747	0.0054	-	-
		GGL [20]	0.2296	0.8746	6.5874	0.1174	-	-
		GIFD [7]	0.2271	0.8592	6.5855	0.0805	-	-
		Ours	**0.0148**	**0.2997**	**9.3730**	**0.4736**	**0.2449**	**6.0964**

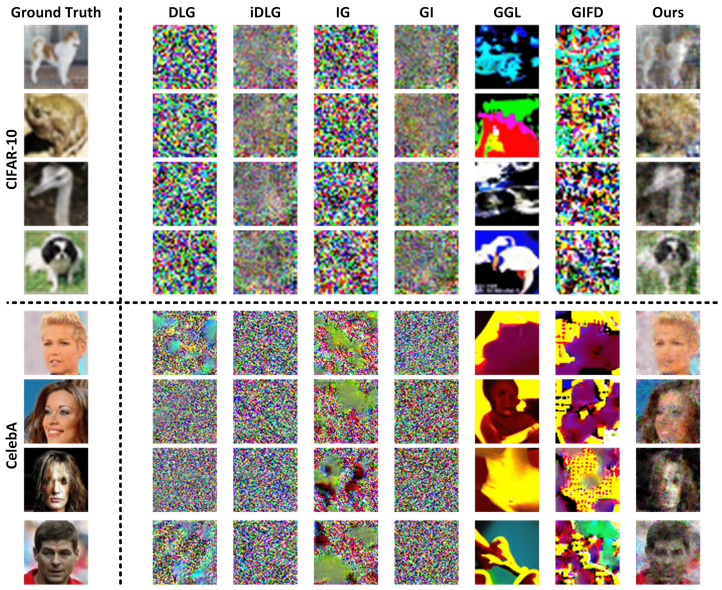

Fig. 4: Visualization of different attack results for VB-ResNet18 and Bayesian ConvNet in the CIFAR-10 classification and gender classification FL tasks, respectively.

5.3 Batch Images reconstruction

We evaluate the efficacy of our ISG attack in reconstructing a batch of images from an averaged gradient in SNNs. Specifically, in the Tiny-ImageNet classification task, we compute the VB-MLP gradients via a single local training step across varying batch sizes. Table 2 and Figure 5 illustrate the quantitative analysis and visual reconstructions of batch images from an averaged gradient, respectively. The findings indicate that our method successfully recovers batches of images from an averaged gradient, with the reconstructed images retaining significantly more fine detail compared to reconstructions from convolutional model gradients as presented in Sec. 5.2. This observation aligns with our earlier insights in Sec. 4.1 that the shared weights in convolutional kernels can potentially cause optimization to be trapped in local minima.

5.4 Ablation Study

In this section, we systematically analyze the individual contributions of key components within our proposed ISG attack on SNNs, to quantify their respective impacts on the attack's overall effectiveness. The results of this analysis, focusing on the CIFAR-100 image classification task utilizing VB-ResNet32, are detailed in Table 3. We adjust several critical components: the process of Inverting Stochasticity denoted by "IS", the regularization term $\mathcal{R}_{\texttt{kl}}(\cdot)$, the feature correction multiplier m, and the feature regularization term $\mathcal{R}_{\texttt{feat}}(\cdot)$. Symbols ✓ and

Table 2: Quality of Batch Reconstruction by Using Gradients of VB-MLP.

Dataset	Batch Size	Reconstructed Images $\hat{x}$				$\hat{z}_{\text{vb}}$	
		MSE ↓	LPIPS ↓	PSNR ↑	SSIM ↑	MSE ↓	SNR ↑
Tiny-ImageNet	1	9.4e-5	0.0004	22.2502	0.9917	0.0024	28.7953
	8	0.0028	0.0083	19.4303	0.9747	0.0354	27.1164
	16	0.0058	0.0197	18.0602	0.9556	0.1059	24.6211
	32	0.0071	0.0276	15.8763	0.9256	0.1947	21.8593
	64	0.0124	0.0426	14.0831	0.8885	0.3311	19.6521
	128	0.0294	0.1537	10.4532	0.7030	1.8609	10.2723

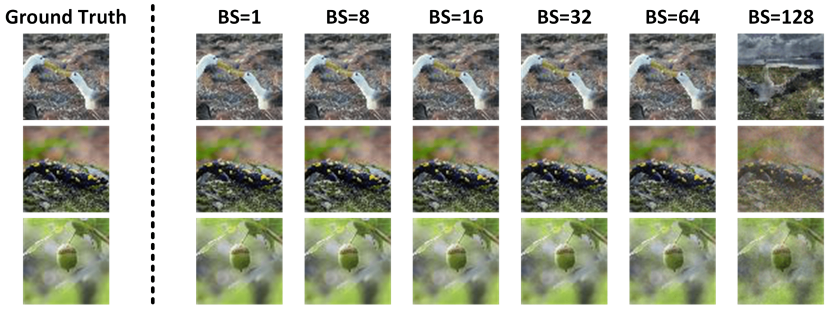

Fig. 5: Visual results of VB-MLP in the Tiny-ImageNet classification task

− are used to indicate the inclusion and exclusion, respectively, of these components. The reconstruction process for stochastic elements shows the critical importance when adopting GIAs to SNNs. GIA cannot accurately replicate the authentic training process without knowledge of the stochastic elements, thus hindering its ability to reconstruct the training data. $\mathcal{R}_{\texttt{kl}}(\cdot)$ contributes to a better reconstruction of stochastic features by constraining the distribution of those features, which results in a reduced MSE of the reconstructed stochastic features and an improved SNR.

Our proposed feature correction multiplier m, in conjunction with the feature regularization term $\mathcal{R}_{\texttt{feat}}(\cdot)$, plays a pivotal role in enhancing the fidelity of reconstructed images. Incorporating them together resulted in nearly a 50% reduction in the LPIPS index as reported in Tab. 3. To delve deeper into their contributions, Fig. 6 shows the MSE between the actual and dummy gradients across iterations under different configurations. These include scenarios without incorporating m and $\mathcal{R}_{\texttt{feat}}(\cdot)$ (denoted as "w/o m", as the baseline) and with different scale factors ($\alpha_{\texttt{feat}}$) for $\mathcal{R}_{\texttt{feat}}(\cdot)$. The curves demonstrate that utilizing our proposed feature constraint strategy significantly narrows the gap between the dummy and actual gradients — achieving nearly a 4× improvement over the baseline. This suggests that our proposed strategy effectively limits the optimization space. Additionally, a visual comparison on the right side of Fig. 6 shows that when $\alpha_{\texttt{feat}} = 1$, the reconstructed images reveal considerably more

Table 3: Ablation Study Results on CIFAR-100 Image Reconstruction from VB-ResNet34 Gradients. IS denotes the Process of Inverting Stochasticity (see Sec. 4.2).

IS	$\mathcal{R}_{\text{kl}}$	m	$\mathcal{R}_{\text{feat}}$	Reconstructed Images				$\hat{z}_{\text{vb}}$	
				MSE ↓	LPIPS ↓	PSNR ↑	SSIM ↑	MSE ↓	SNR ↑
–	–	✓	✓	0.1195	0.2237	4.8976	0.1518	2.0487	-3.248
✓	–	–	–	0.0274	0.0435	8.1699	0.5639	0.0584	12.8174
✓	✓	–	–	0.0313	0.0536	7.9803	0.5326	0.0524	12.8780
✓	✓	✓	–	0.0253	0.0482	8.2175	0.5770	0.0731	11.1999
✓	✓	✓	✓	**0.0215**	**0.0273**	**8.6080**	**0.6295**	**0.0441**	**13.1634**

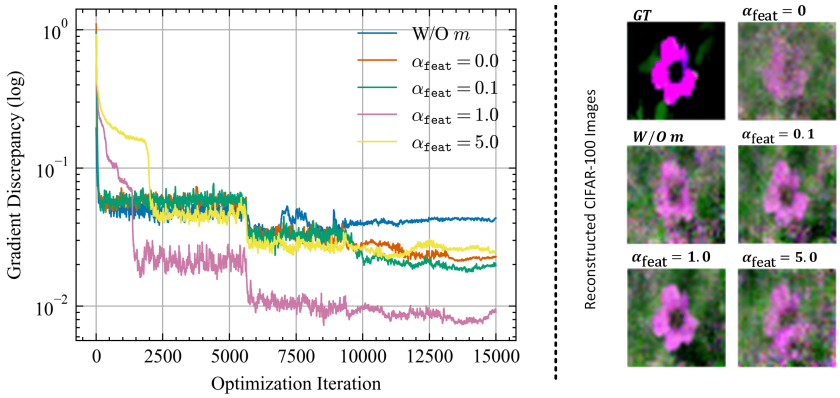

Fig. 6: The left part shows the MSE between the true and dummy gradients as the optimization progresses. The right part depicts the reconstructed images. Both parts are evaluated on the CIFAR-100 dataset with different hyperparameter settings.

details than the baseline, aligning with the curve results. All experimental results substantially support our analyses and motivations discussed in Sec. 4.1.

6 Conclusion

In this work, we overturned the assumption that SNNs are inherently resistant to GIAs due to their stochastic nature. Our findings exposed their vulnerability, by showing that SNN gradients retain information about stochastic components, allowing attackers to reconstruct these components and reduce their stochastic impact. We demonstrated that existing defense strategies relying on stochastic perturbations during forward propagation fall short against GIAs. This suggests a pressing need for more robust defense mechanisms. Alternatively, we suggest exploring gradient perturbation-based defenses, such as Gradient Sparsification and Noisy Gradient, which, despite their trade-off between model performance and privacy [12, 35].

Acknowledgements

This work was supported in part by the Research Foundation - Flanders (FWO) through the Project under Grant G014718N and in part by the Flemish Government, under the "Onderzoeksprogramma Artificieële Intelligentie (AI) Vlaanderen" Programme.

References

1. Alemi, A.A., Fischer, I., Dillon, J.V., Murphy, K.: Deep variational information bottleneck. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net (2017), https://openreview.net/forum?id=HyxQzBceg
2. Balunovic, M., Dimitrov, D.I., Staab, R., Vechev, M.: Bayesian framework for gradient leakage. In: International Conference on Learning Representations (2022), https://openreview.net/forum?id=f2lrIbGx3x7
3. Blundell, C., Cornebise, J., Kavukcuoglu, K., Wierstra, D.: Weight uncertainty in neural network. In: Bach, F., Blei, D. (eds.) Proceedings of the 32nd International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 37, pp. 1613–1622. PMLR, Lille, France (07–09 Jul 2015), https://proceedings.mlr.press/v37/blundell15.html
4. Brock, A., Donahue, J., Simonyan, K.: Large scale GAN training for high fidelity natural image synthesis. In: International Conference on Learning Representations (2019), https://openreview.net/forum?id=B1xsqj09Fm
5. Chen, H., Zhu, T., Zhang, T., Zhou, W., Yu, P.S.: Privacy and fairness in federated learning: On the perspective of tradeoff. ACM Comput. Surv. **56**(2) (sep 2023). https://doi.org/10.1145/3606017, https://doi.org/10.1145/3606017
6. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. 2009 IEEE Conference on Computer Vision and Pattern Recognition pp. 248–255 (2009), https://api.semanticscholar.org/CorpusID:57246310
7. Fang, H., Chen, B., Wang, X., Wang, Z., Xia, S.T.: Gifd: A generative gradient inversion method with feature domain optimization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 4967–4976 (2023)
8. Ganju, K., Wang, Q., Yang, W., Gunter, C.A., Borisov, N.: Property inference attacks on fully connected neural networks using permutation invariant representations. Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (2018)
9. Geiping, J., Bauermeister, H., Dröge, H., Moeller, M.: Inverting gradients-how easy is it to break privacy in federated learning? Advances in Neural Information Processing Systems **33**, 16937–16947 (2020)
10. Gong, H., Jiang, L., Liu, X.Y., Wang, Y., Gastro, O., Wang, L., Zhang, K., Guo, Z.: Gradient leakage attacks in federated learning. Artificial Intelligence Review **56**, 1337–1374 (2023), https://api.semanticscholar.org/CorpusID:260123471
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
12. Huang, Y., Gupta, S., Song, Z., Li, K., Arora, S.: Evaluating gradient inversion attacks and defenses in federated learning. Advances in Neural Information Processing Systems **34**, 7232–7241 (2021)

13. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning. pp. 448–456. pmlr (2015)
14. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (ICLR). San Diega, CA, USA (2015)
15. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings (2014), available at: http://arxiv.org/pdf/1312.6114v10
16. Krizhevsky, A.: Learning multiple layers of features from tiny images. Tech. rep., University of Toronto (2009)
17. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems **25** (2012)
18. Le, Y., Yang, X.: Tiny imagenet visual recognition challenge. CS 231N **7**(7), 3 (2015)
19. Li, Q., He, B., Song, D.: Model-contrastive federated learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2021)
20. Li, Z., Zhang, J., Liu, L., Liu, J.: Auditing privacy defenses in federated learning via generative gradient leakage. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10132–10142 (2022)
21. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: Proceedings of the IEEE international conference on computer vision. pp. 3730–3738 (2015)
22. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial intelligence and statistics. pp. 1273–1282. PMLR (2017)
23. Nguyen, D.C., Ding, M., Pathirana, P.N., Seneviratne, A., Li, J., Poor, H.V.: Federated learning for internet of things: A comprehensive survey. IEEE Communications Surveys & Tutorials **23**(3), 1622–1658 (2021)
24. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch. In: NIPS-W (2017)
25. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D: Nonlinear Phenomena **60**(1), 259–268 (1992). https://doi.org/https://doi.org/10.1016/0167-2789(92)90242-F, https://www.sciencedirect.com/science/article/pii/016727899290242F
26. Scheliga, D., Mäder, P., Seeland, M.: Precode - a generic model extension to prevent deep gradient leakage. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 1849–1858 (January 2022)
27. Sun, J., Li, A., Wang, B., Yang, H., Li, H., Chen, Y.: Soteria: Provable defense against privacy leakage in federated learning from representation perspective. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 9307–9315 (2020), https://api.semanticscholar.org/CorpusID:228375828
28. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing **13**, 600–612 (2004), https://api.semanticscholar.org/CorpusID:207761262
29. Wei, W., Liu, L., Wu, Y., Su, G., Iyengar, A.: Gradient-leakage resilient federated learning. In: 2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS). pp. 797–807. IEEE (2021)

30. Xie, X., Hu, C., Ren, H., Deng, J.: A survey on vulnerability of federated learning: A learning algorithm perspective. Neurocomputing **573**, 127225 (2024). https://doi.org/https://doi.org/10.1016/j.neucom.2023.127225, https://www.sciencedirect.com/science/article/pii/S0925231223013486
31. Xu, J., Glicksberg, B.S., Su, C., Walker, P., Bian, J., Wang, F.: Federated learning for healthcare informatics. Journal of Healthcare Informatics Research **5**, 1–19 (2021)
32. Yin, H., Mallya, A., Vahdat, A., Alvarez, J.M., Kautz, J., Molchanov, P.: See through gradients: Image batch recovery via gradinversion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16337–16346 (2021)
33. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018)
34. Zhao, B., Mopuri, K.R., Bilen, H.: idlg: Improved deep leakage from gradients. arXiv preprint arXiv:2001.02610 (2020)
35. Zhu, L., Liu, Z., Han, S.: Deep leakage from gradients. Advances in neural information processing systems **32** (2019)