GI-NAS: Boosting Gradient Inversion Attacks Through Adaptive Neural Architecture Search

Wenbo Yu, Hao Fang, Bin Chen, *Member, IEEE*, Xiaohang Sui, Chuan Chen, *Member, IEEE*, Hao Wu, Shu-Tao Xia, *Member, IEEE*, and Ke Xu, *Fellow, IEEE*

***Abstract*—Gradient Inversion Attacks invert the transmitted gradients in Federated Learning (FL) systems to reconstruct the sensitive data of local clients and have raised considerable privacy concerns. A majority of gradient inversion methods rely heavily on explicit prior knowledge (e.g., a well pre-trained generative model), which is often unavailable in realistic scenarios. This is because real-world client data distributions are often highly heterogeneous, domain-specific, and unavailable to attackers, making it impractical for attackers to obtain perfectly matched pre-trained models, which inevitably suffer from fundamental distribution shifts relative to target private data. To alleviate this issue, researchers have proposed to leverage the implicit prior knowledge of an over-parameterized network. However, they only utilize a fixed neural architecture for all the attack settings. This would hinder the adaptive use of implicit architectural priors and consequently limit the generalizability. In this paper, we further exploit such implicit prior knowledge by proposing Gradient Inversion via Neural Architecture Search (GI-NAS), which adaptively searches the network and captures the implicit priors behind neural architectures. Extensive experiments verify that our proposed GI-NAS can achieve superior attack performance compared to state-of-the-art gradient inversion methods, even under more practical settings with high-resolution images, large-sized batches, and advanced defense strategies. To the best of our knowledge, we are the first to successfully introduce NAS to the gradient inversion community. We believe that this work exposes critical vulnerabilities in real-world federated learning by demonstrating high-fidelity reconstruction of sensitive data without requiring domain-specific priors, forcing urgent reassessment of FL privacy safeguards. The source code is available at https://github.com/cswbyu/GI-NAS**

***Index Terms*—Federated learning, gradient inversion attacks, neural architecture search, privacy leakage.**

I. Introduction

FEDERATED Learning (FL) [1], [2], [3] serves as an efficient collaborative learning framework where multiple participants cooperatively train a global model and only the computed gradients are exchanged. By adopting this distributed paradigm, FL systems fully leverage the huge amounts of data partitioned across various clients for enhanced model efficacy and tackle the separateness of data silos [4]. Moreover, since merely the gradients instead of the private data are uploaded to the server, the user privacy seems to be safely guaranteed as the private data is only available at the client side.

However, FL systems are actually not so secure as what people have expected [5], [6], [7], [8], [9], [10], [11], [12], [13]. Extensive studies have discovered that even the transmitted gradients can disclose the sensitive information of users. Early works [5], [6], [7], [8] involve inferring the existence of certain samples in the dataset (i.e., Membership Inference Attacks [14]) or further revealing some properties of the private training set (i.e., Property Inference Attacks [15]) from the uploaded gradients. But unlike the above inference attacks that only partially reveal limited information of the private data, Gradient Inversion Attacks [9], [10], [11], [12], [13] stand out as a more threatening privacy risk as they can completely reconstruct the sensitive data by inverting the gradients.

Zhu et al. [9] first formulate gradient inversion as an optimization problem and recover the private training images by minimizing the gradient matching loss (i.e., the distance between the dummy gradients and the real gradients) with image pixels regarded as trainable parameters. Ensuing works improve the attack performance on the basis of Zhu et al. [9] by designing label extraction techniques [16], switching the distance metric and introducing regularization [10], or considering settings with larger batch sizes [17], but still restrict their optimizations in the pixel space. To fill this gap, recent studies [11], [12], [18] propose to explore various search algorithms within the Generative Adversarial Networks

Received 25 October 2024; revised 22 June 2025; accepted 7 July 2025. Date of publication 15 July 2025; date of current version 25 July 2025. This work was supported in part by the National Science Foundation for Distinguished Young Scholars of China under Grant 62425201; in part by the National Natural Science Foundation of China under Grant 62171248, Grant 62301189, and Grant 62176269; in part by the National Key Research and Development Program of China under Grant 2023YFB2703700; and in part by Shenzhen Science and Technology Program under Grant KJZD20240903103702004, Grant JCYJ20220818101012025, and Grant GXWD20220811172936001. The associate editor coordinating the review of this article and approving it for publication was Dr. Paolo Gasti. *(Wenbo Yu and Hao Fang contributed equally to this work.) (Corresponding author: Bin Chen.)*

Wenbo Yu, Hao Fang, and Shu-Tao Xia are with the Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen, Guangdong 518055, China (e-mail: wenbo.research@gmail.com; ffhibnese@gmail.com; xiast@sz.tsinghua.edu.cn).

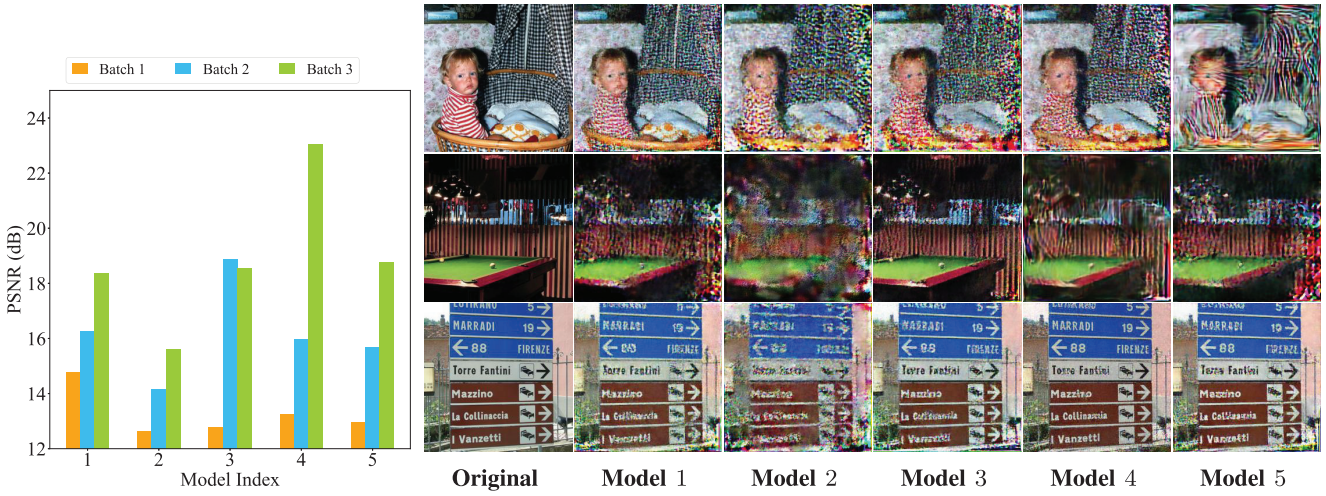
Bin Chen and Xiaohang Sui are with the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, Guangdong 518055, China (e-mail: chenbin2021@hit.edu.cn; suixiaohang@stu.hit.edu.cn).

Chuan Chen is with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, Guangdong 510006, China (e-mail: chenchuan@mail.sysu.edu.cn).

Hao Wu is with Shenzhen ShenNong Information Technology Company Ltd., Shenzhen, Guangdong 518000, China, and also with the Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen, Guangdong 518055, China (e-mail: wu-h22@mails.tsinghua.edu.cn).

Ke Xu is with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China (e-mail: xuke@tsinghua.edu.cn).

Digital Object Identifier 10.1109/TIFS.2025.3589127



(a) Quantitative Results (b) Qualitative Results

Fig. 1. Quantitative and qualitative results when randomly attacking 3 different batches by 5 different models on ImageNet. In Figure 1b, the first, second, and third rows are respectively from Batch 1, Batch 2, and Batch 3.

(GAN) [19], [20] to leverage the rich prior knowledge encoded in the pre-trained generative models. While incorporating these explicit priors indeed improves the attack performance, it is usually tough to pre-prepare such prerequisites in realistic FL scenarios where the data distribution at the client side is likely to be complex or unknown to the attackers. Therefore, Zhang et al. [13] propose to employ an over-parameterized convolutional network for gradient inversion and directly optimize the excessive network parameters on a fixed input, which does not require any explicit prior information but still outperforms GAN-based attacks. The reason behind this is that the structure of a convolutional network naturally captures image statistics prior to any learning [21] and Zhang et al. [13] leverage this characteristic as implicit prior knowledge. However, Zhang et al. [13] only utilize a fixed network for all the attack settings, regardless of the specific batch to recover. An intuitive question naturally arises: *Can we adaptively search the architecture of the over-parameterized network to better capture the implicit prior knowledge for each reconstructed batch?*

In Figure 1, we randomly select 5 over-parameterized networks with different architectures to attack 3 different batches. All the networks are optimized for the same number of iterations on ImageNet [22]. The results indicate that the Peak Signal-to-Noise Ratio (PSNR) performance varies significantly when changing the architectures. For the same batch, various architectures can hold remarkably different implicit priors on it. Besides, since the optimal models for Batch 1, Batch 2, and Batch 3 are respectively Model 1, Model 3, and Model 4, there exists no network that can consistently perform the best on all the given batches. Thus, the rigid use of a single architecture by Zhang et al. [13] lacks optimality under dynamic FL scenarios where different batches require different implicit architectural priors, and it is of great significance to adaptively select the most suitable architecture for each batch.

Inspired by the above phenomena, we propose a novel gradient inversion method, named **G**radient **I**nversion via **N**eural **A**rchitecture **S**earch (GI-NAS), to better match each batch with the optimal model architecture. Specifically, we first enlarge the potential search space for the over-parameterized network by designing different upsampling modules and skip connection patterns. To reduce the computational overhead, we utilize a training-free search strategy that compares the initial gradient matching loss for a given batch over all the candidates and selects the best of them for the final optimization. We further provide substantial experimental evidence that such a metric highly correlates with the real attack performance. We also consider more rigorous and realistic scenarios where the victims may hold high-resolution images and large-sized batches for training, and evaluate advanced defense strategies. Extensive experiments validate that GI-NAS can achieve state-of-the-art performance compared to existing gradient inversion methods. To the best of our knowledge, we are the first to introduce Neural Architecture Search (NAS) to Gradient Inversion Attacks. Our main contributions are as follows:

- We systematically analyze existing methods, emphasize the necessity of equipping image batch recovery with the optimal model structure, and propose GI-NAS to boost gradient inversion through neural architectural search.
- We expand the model search space by considering different upsampling units and skip connection modes, and utilize a training-free search that regards the initial gradient matching loss as the search metric. We also provide extensive experimental evidence that such a metric highly correlates with the real attack performance.
- Numerous experimental results demonstrate that GI-NAS outperforms state-of-the-art gradient inversion methods, even under extreme settings with high-resolution images, large-sized batches, and advanced defense strategies.
- We provide deeper analysis on various aspects, such as computational efficiency, ablation studies, generalizability to more FL global models, robustness to network parameters and latent codes initialization, and implications behind the NAS searched results.

A. Research Purpose and Real-World Significance

This research reveals a critical security vulnerability in federated learning by developing GI-NAS, an adaptive gradient inversion attack that dynamically adjusts to diverse real-world conditions. Unlike existing methods constrained by fixed architectures or unrealistic data assumptions, our approach automatically identifies optimal network configurations through neural architecture search, enabling more effective privacy attacks across varying batch characteristics. The practical significance lies in exposing security risks in actual FL deployments where data resolutions, batch sizes, and defense mechanisms constantly vary. By demonstrating how the architectural adaptation enhances the attack effectiveness, we believe that this work will not only advance attack methodologies but also provide crucial insights for developing stronger defense systems in real-world distributed learning environments where sensitive data protection remains paramount.

II. Related Work

A. Gradient Inversion Attacks and Defenses

Zhu et al. [9] first propose to restore the private samples via iterative optimization for pixel-level reconstruction, yet limited to low-resolution and single images. Geiping et al. [10] empirically decompose the gradient vector by its norm magnitude and updating direction, and succeed on high-resolution ImageNet [22] through the angle-based loss design. Furthermore, Yin et al. [17] extend the attacks to the batch-level inversion through the group consistency regularization and the improved batch label inference algorithm [16]. With strong handcrafted explicit priors (e.g., fidelity regularization, BN statistics), they accurately realize batch-level reconstruction with detailed semantic features allocated to all the individual images in a batch. Subsequent studies [11], [12], [18] leverage pre-trained GAN models as generative priors to enhance the attacks. Current defenses focus on gradient perturbation to alleviate the impact of gradient inversion with degraded gradients [23], [24]. Gaussian Noise [25] and Gradient Clipping [26] are common techniques in Differential Privacy (DP) [27], [28] that effectively constrain the attackers from learning through the released gradients. Gradient Sparsification [29], [30] prunes the gradients through a given threshold, and Soteria [31] edits the gradients from the aspect of learned representations.

B. Neural Architecture Search (NAS)

By automatically searching the optimal model architecture, NAS algorithms have shown significant effectiveness in multiple visual tasks such as image restoration [32], [33], semantic segmentation [34], [35], and image classification [36], [37]. In image classification tasks, Zoph and Le [36] regularize the search space to a convolutional cell and construct a better architecture stacked by these cells using an RNN [38] controller. For semantic segmentation tasks, Liu et al. [35] search for the optimal network on the level of hierarchical network architecture and extend NAS to dense image prediction. In terms of image restoration tasks, Suganuma et al. [32] exploit a better Convolutional Autoencoders (CAE) with standard network components through the evolutionary search, while Chu et al. [33] discover a competitive lightweight model via both micro and macro architecture search.

Existing NAS methods adopt different strategies to exploit the search space, including evolutionary methods [39], [40], [41], Bayesian optimization [42], [43], [44], Reinforcement Learning (RL) [36], [45], [46], [47], and gradient-based search [48], [49], [50]. Different RL-based methods vary in the way to represent the agent's policy and the optimization process. Zoph and Le [36] utilize RNN networks to sequentially encode the neural architecture and train them with the REINFORCE policy gradient algorithm [51]. Baker et al. [45] adopt Q-learning [52], [53] to train the policy network and realize competitive model design. Notably, recent works [54], [55], [56] propose training-free NAS to mitigate the issue of huge computational expenses. Instead of training from scratch, they evaluate the searched networks by some empirically designed metrics that reflect model effectiveness. In this paper, we adopt the initial gradient matching loss as the training-free search metric and provide substantial empirical evidence that such a search metric highly correlates with the real attack performance.

III. Problem Formulation

A. Basics of Gradient Inversion

We consider the training process of a classification model f_θ parameterized by θ in FL scenarios. The real gradients $\mathbf{g}$ are calculated from a private batch (with real images $\mathbf{x}$ and real labels $\mathbf{y}$) at the client side. The universal goal of Gradient Inversion Attacks is to search for some fake images $\hat{\mathbf{x}} \in \mathbb{R}^{B\times H\times W\times C}$ with labels $\hat{\mathbf{y}} \in \{0,1\}^{B\times L}$ so that $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ can be close to $(\mathbf{x}, \mathbf{y})$ as much as possible, where B, H, W, C, and L are respectively batch size, image height, image width, number of channels, and number of classes. This can be realized by minimizing the gradient matching loss [9]:

$$\hat{\mathbf{x}}^*, \hat{\mathbf{y}}^* = \arg\min_{\hat{\mathbf{x}},\hat{\mathbf{y}}} \mathcal{D}(\nabla_\theta \mathcal{L}(f_\theta(\hat{\mathbf{x}}), \hat{\mathbf{y}}), \mathbf{g}), \tag{1}$$

where $\mathcal{D}(\cdot,\cdot)$ is the distance metric (e.g., l_2-norm loss, cosine-similarity loss) for the gradient matching loss and $\mathcal{L}(\cdot,\cdot)$ is the loss function of the global model f_θ.

Previous works [16], [17] in this field have revealed that the ground truth labels $\mathbf{y}$ can be directly inferred from the uploaded gradients $\mathbf{g}$. Therefore, the formulation in (1) can be simplified as:

$$\hat{\mathbf{x}}^* = \arg\min_{\hat{\mathbf{x}}} \mathcal{D}(F(\hat{\mathbf{x}}), \mathbf{g}), \tag{2}$$

where $F(\hat{\mathbf{x}}) = \nabla_\theta \mathcal{L}(f_\theta(\hat{\mathbf{x}}), \hat{\mathbf{y}})$ calculates the gradients of f_θ provided with $\hat{\mathbf{x}}$.

The key challenge of (2) is that gradients only provide limited information of private data and there even exists a pair of different data having the same gradients [57]. To mitigate this issue, subsequent works incorporate various regularizations (e.g., total variation loss [10], group consistency loss [17]) as prior knowledge. Thus, the overall optimization becomes:

$$\hat{\mathbf{x}}^* = \arg\min_{\hat{\mathbf{x}}} \mathcal{D}(F(\hat{\mathbf{x}}), \mathbf{g}) + \lambda \mathcal{R}_{prior}(\hat{\mathbf{x}}), \tag{3}$$

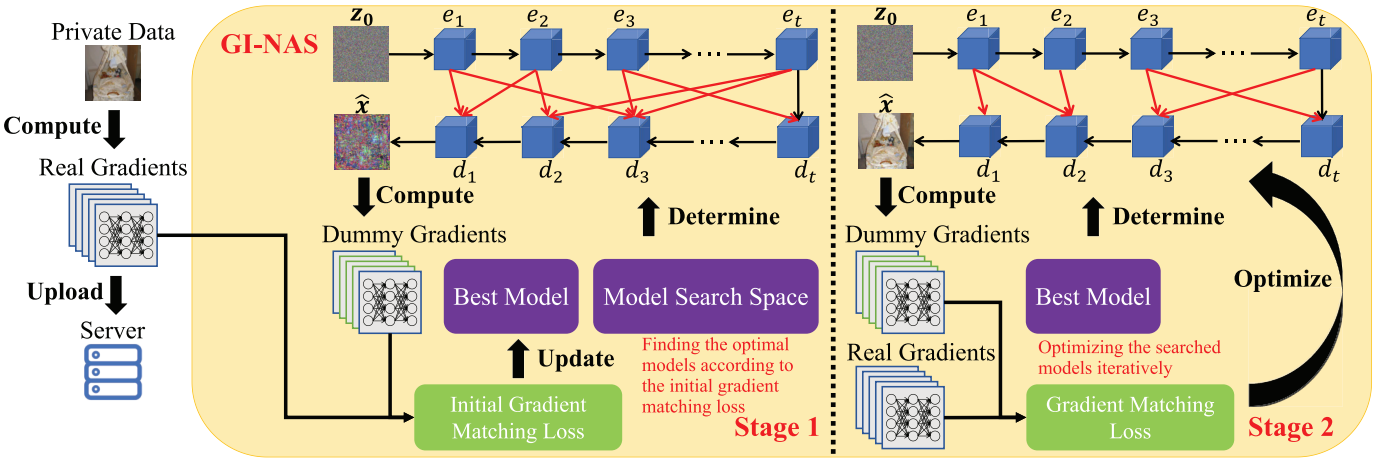


Fig. 2. Overview of the proposed GI-NAS attack. We leverage a two-stage strategy for private batch recovery. In the first stage, we traverse the model search space and calculate the initial gradient matching loss (i.e., our training-free search metric) of each model based on the fixed input $\mathbf{z_0}$. We regard the model that achieves the minimal initial loss as our best model, for its performance at the start can stand out from numerous candidates. In the second stage, we adopt the architecture of the previously found best model and optimize its excessive parameters to reconstruct the private data.

where $\mathcal{R}_{prior}(\cdot)$ is the introduced regularization that can establish some priors for the attacks, and λ is the weight factor.

B. GAN-Based Gradient Inversion

Nevertheless, the optimization of (3) is still limited in the pixel space. Given a well pre-trained GAN, an instinctive idea is to switch the optimization from the pixel space to the GAN latent space:

$$\mathbf{z}^* = \arg\min_{\mathbf{z}} \mathcal{D}(F(G_\omega(\mathbf{z})), \mathbf{g}) + \lambda \mathcal{R}_{prior}(G_\omega(\mathbf{z})), \tag{4}$$

where G_ω and $\mathbf{z} \in \mathbb{R}^{B\times l}$ are respectively the generator and the latent vector of the pre-trained GAN. By reducing the optimization space from $\mathbb{R}^{B\times H\times W\times C}$ to $\mathbb{R}^{B\times l}$, (4) overcomes the uncertainty of directly optimizing the extensive pixels and exploits the abundant prior knowledge encoded in the pre-trained GAN. Based on this, recently emerged GAN-based attacks [11], [12], [18] explore various search strategies within the pre-trained GAN to utilize its expression ability.

C. Gradient Inversion via Over-Parameterized Networks

But as previously mentioned, incorporating such GAN priors is often impractical in realistic scenarios where the distribution of $\mathbf{x}$ is likely to be mismatched with the training data of the pre-trained GAN. Furthermore, it has already been discovered in [13] that explicitly introducing regularization in (3) or (4) may not necessarily result in convergence towards $\mathbf{x}$, as even ground truth images cannot guarantee minimal loss when $\mathcal{R}_{prior}(\cdot)$ is added. To mitigate these issues, Zhang et al. [13] propose to leverage an over-parameterized network as implicit prior knowledge:

$$\phi^* = \arg\min_{\boldsymbol{\phi}} \mathcal{D}(F(G_{over}(\mathbf{z_0}; \boldsymbol{\phi})), \mathbf{g}), \tag{5}$$

where G_{over} is the over-parameterized convolutional network with excessive parameters $\boldsymbol{\phi}$, and $\mathbf{z_0}$ is the randomly generated but fixed latent code. Note that the regularization term is omitted in (5). This is because the architecture of G_{over} itself can serve as implicit regularization, for convolutional networks have been discovered to possess implicit priors that prioritize clean images rather than noise as shown in [21]. Thus, the generated images that highly resemble the ground truth images can be obtained through $\hat{\mathbf{x}}^* = G_{over}(\mathbf{z_0}; \phi^*)$. However, only a changeless over-parameterized network is employed for all the attack settings in [13]. As previously shown in Figure 1, although the network is over-parameterized, the attack performance exhibits significant differences when adopting different architectures. To fill this gap, we propose to further exploit such implicit architectural priors by searching the optimal over-parameterized network G_{opt} for each batch. We will discuss how to realize this in Section IV.

IV. Method

Our proposed GI-NAS attack is carried out in two stages. In the first stage, we conduct our architecture search to decide the optimal model G_{opt}. Given that the attackers may only hold limited resources, we utilize the initial gradient matching loss as the training-free search metric to reduce the computational overhead. In the second stage, we iteratively optimize the parameters of the selected model G_{opt} to recover the sensitive data. Figure 2 illustrates the overview of our method.

A. Threat Model

1) Attacker's Aim: The fundamental aim of an attacker in our method is to reconstruct the original private training data $\mathbf{x}$ from client devices with maximum fidelity by exploiting the exchanged gradients $\mathbf{g}$ in federated learning systems.

2) Attacker's Knowledge: The attacker possesses three key categories of knowledge: complete access to the gradient vectors $\mathbf{g}$ transmitted during federated updates, including their structural organization and numerical values; full architectural specifications of the global model f_θ being trained; and the standard assumptions about input data dimensions $\mathbf{x} \in \mathbb{R}^{B\times H\times W\times C}$ without requiring distributional priors that are needed in previous GAN-based attacks [11], [12], [18].

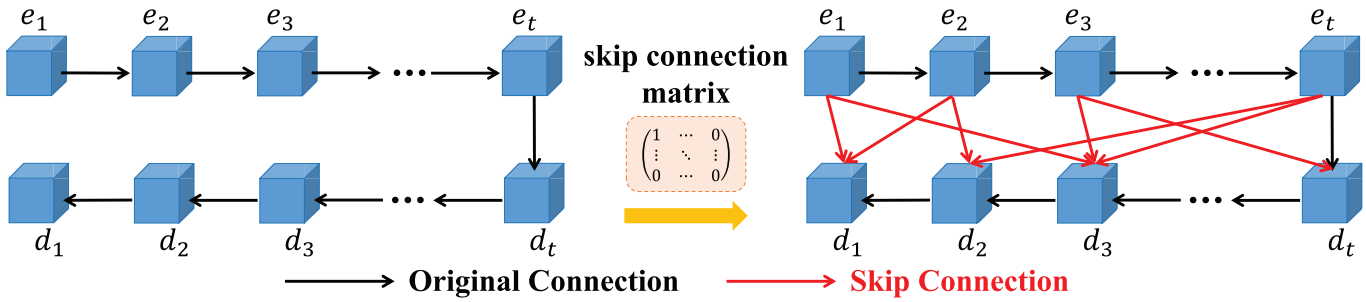


Fig. 3. The design of search space for skip connection patterns. Different skip connection patterns are determined by the skip connection matrix $\mathbf{A} \in \{0,1\}^{t\times t}$. $\mathbf{A}_{ij} = 1$ indicates that there exists a skip connection from e_i to d_j and $\mathbf{A}_{ij} = 0$ means that there is not such a skip connection.

Crucially, the knowledge requirement in our method excludes any direct access to the original training data samples or client-side information beyond the gradients, making our threat model both realistic and concerning for practical deployments.

3) Attacker's Capabilities: The attacker's capabilities encompass executing local computations to optimize reconstruction networks G_{opt}, deploying adaptive architectures through our NAS framework to handle varying batch characteristics, and maintaining attack efficacy even when gradients are protected by common defense mechanisms. The attacker is not allowed to modify the exchanged gradients or transmit malicious data that may compromise the training process, even if doing so can enhance the reconstruction results.

B. Stage 1: Training-Free Optimal Model Search

Before we elaborate on the Stage 1 (i.e., Training-free Optimal Model Search), we outline its key components and objectives as follows:

- Input for this stage: Real gradients $\mathbf{g}$, fixed latent code $\mathbf{z_0}$ sampled from $\mathcal{N}(0,1)$, and the model search space $\mathcal{M} = \{G_1, G_2, \ldots, G_n\}$.
- Output for this stage: The searched optimal architecture G_{opt} with minimal initial loss.
- Purpose for this stage: In this stage, we traverse the model search space and calculate the initial gradient matching loss (i.e., our training-free search metric) of each model based on the fixed input $\mathbf{z_0}$. We regard the model that achieves the minimal initial loss as our best model G_{opt}.

1) Model Search Space Design: One crucial factor for NAS is that the potential search space is large and diverse enough to cover the optimal design. Therefore, we adopt U-Net [58], a typical convolutional neural architecture as the fundamental of our model search, since the skip connection patterns between its encoders and decoders can provide adequate alternatives of model structure. Besides, the configurations of the upsampling modules (e.g., kernel size, activation function) can also enable numerous possibilities when arranged and combined. Similar to previous NAS methods [59], [60], we enlarge the search space of our model from two aspects, namely *Upsampling Modules* and *Skip Connection Patterns*.

a) Search space for upsampling modules: We decompose the upsampling operations into five key components: feature upsampling, feature transformation, activation function, kernel size, and dilation rate. Then, we allocate a series of possible options to each of these components. When deciding on feature upsampling, we choose from commonly used interpolation techniques, such as bilinear interpolation, bicubic interpolation, and nearest-neighbour interpolation. As for feature transformation, we choose from classical convolution techniques, such as 2D convolution, separable convolution, and depth-wise convolution. As regards activation function, we select from ReLU, LeakyReLU, PReLU, etc. Furthermore, we supply kernel size and dilation rate with more choices, such as 1×1, 3×3, or 5×5 for kernel size and 1, 3, or 5 for dilation rate. The combination of these flexible components can contribute to the diversity of upsampling modules.

b) Search space for skip connection patterns: We assume that there are t levels of encoders and decoders in total, and denote them as $e_1, e_2, \ldots, e_t$ and $d_1, d_2, \ldots, d_t$. As shown in Figure 3, We consider different skip connection patterns between encoders and decoders. To represent each of these patterns, we define a skip connection matrix $\mathbf{A} \in \{0,1\}^{t\times t}$ that serves as a mask to determine whether there will be new residual connections [61] between pairs of encoders and decoders. More concretely, $\mathbf{A}_{ij} = 1$ indicates that there exists a skip connection from e_i to d_j and $\mathbf{A}_{ij} = 0$ means that there is not such a skip connection. As the shapes of feature maps across different network levels can vary significantly (e.g., 64×64 for the output of e_i and 256×256 for the input of d_j), we introduce connection scale factors to tackle this inconsistency and decompose all the possible scale factors into a series of $2\times$ upsampling operations or downsampling operations with shared weights. By allocating 0 or 1 to each of the t^2 bits in $\mathbf{A}$, we broaden the search space of skip connection patterns as there are 2^{t^2} possibilities altogether and we only need to sample a portion of them.

2) Optimal Model Selection: We build up our model search space $\mathcal{M}$ by combining the possibilities of the aforementioned upsampling modules and skip connection patterns. We assume that the size of our model search space is n and the candidates inside it are denoted as $\mathcal{M} = \{G_1, G_2, \ldots, G_n\}$. We first sample the latent code $\mathbf{z_0}$ from the Gaussian distribution and freeze its values on all the models for fair comparison. We then traverse the model search space and calculate the initial gradient matching loss of each individual G_r $(1 \le r \le n)$:

$$\mathcal{L}_{grad}(G_r) = \mathcal{D}(\mathcal{T}(F(G_r(\mathbf{z_0}; \boldsymbol{\phi}_r))), \mathbf{g}), \tag{6}$$

where $\mathcal{L}_{grad}(\cdot)$ is the gradient matching loss, $\mathcal{T}(\cdot)$ is the estimated gradient transformation [11], and $\boldsymbol{\phi}_r$ are the parameters of G_r. Here we introduce $\mathcal{T}(\cdot)$ to estimate the gradient transformation following the previous defense auditing work [11], since the victims may apply defense strategies to $\mathbf{g}$ (e.g., Gradient Clipping [25]) and only release disrupted forms of gradients. Empirically, the model that can perform the best at the start and stand out from numerous candidates is likely to have better implicit architectural priors with respect to the private batch. This is because the model would have a superior optimization starting point (i.e., a smaller initial loss) due to its inherent structural advantages. We further provide extensive experimental evidence that such a metric highly correlates with the real attack performance in Section V-D. Therefore, we regard the model that achieves the minimal initial loss as our best model G_{opt} and update our selection during the traversal. Since only the initial loss is calculated and no back-propagation is involved, this search process is training-free and hence computationally efficient.

C. Stage 2: Private Batch Recovery via the Optimal Model

Similar to Section IV-B, we outline the key components and objectives of this stage as follows:

- Input for this stage: The searched optimal architecture G_{opt} from Stage 1, the randomly initialized network parameters γ_1, real gradients $\mathbf{g}$, and the fixed latent code $\mathbf{z_0}$ from Stage 1.
- Output for this stage: The final parameters γ^*, and the reconstructed private data $\hat{\mathbf{x}}^* = G_{opt}(\mathbf{z_0}; \gamma^*)$.
- Purpose for this stage: In this stage, we iteratively optimize the parameters of the selected model G_{opt} to recover the sensitive data.

After deciding on the optimal model G_{opt} with the parameters γ_1, we iteratively conduct gradient decent optimizations on G_{opt} to minimize the gradient matching loss:

$$\boldsymbol{\gamma}_{k+1} = \boldsymbol{\gamma}_k - \eta \nabla_{\boldsymbol{\gamma}_k} \mathcal{L}_{grad}(G_{opt}), \tag{7}$$

where k is the current number of iterations ($1 \le k \le m$), γ_{k+1} are the parameters of G_{opt} after the k-th optimization, and η is the learning rate. Once the above process converges and we obtain $\gamma^* = \boldsymbol{\gamma}_{m+1}$ that satisfy the minimum loss, the private batch can be reconstructed by $\hat{\mathbf{x}}^* = G_{opt}(\mathbf{z_0}; \gamma^*)$. In summary, the pseudocode of GI-NAS is illustrated in Algorithm 1.

Algorithm 1 Gradient Inversion via Neural Architecture Search (GI-NAS)

Input: n: the size of the model search space; $\mathbf{g}$: the uploaded real gradients; m: the maximum iteration steps for the final optimization;

Output: $\hat{\mathbf{x}}^*$: the generated images;

Build up the model search space: $\mathcal{M} = \{G_1, G_2, \ldots, G_n\}$ with the parameters $\Phi = \{\phi_1, \phi_2, \ldots, \phi_n\}$
$\mathbf{z_0} \leftarrow \mathcal{N}(0, 1)$, $loss_{min} \leftarrow +\infty$
for $i \leftarrow 1$ to n **do**
  $loss_i \leftarrow \mathcal{D}(\mathcal{T}(F(G_i(\mathbf{z_0}; \boldsymbol{\phi}_i))), \mathbf{g})$ // calculate the initial gradient matching loss
  if $loss_i < loss_{min}$ **then**
    $loss_{min} \leftarrow loss_i$, $G_{opt} \leftarrow G_i$, $\gamma_1 \leftarrow \boldsymbol{\phi}_i$ // update the selection of G_{opt}
  end if
end for
for $k \leftarrow 1$ to m **do**
  $\boldsymbol{\gamma}_{k+1} \leftarrow \boldsymbol{\gamma}_k - \eta \nabla_{\boldsymbol{\gamma}_k} \mathcal{D}(\mathcal{T}(F(G_{opt}(\mathbf{z_0}; \boldsymbol{\gamma}_k))), \mathbf{g})$
end for
$\gamma^* \leftarrow \boldsymbol{\gamma}_{m+1}$
return $\hat{\mathbf{x}}^* = G_{opt}(\mathbf{z_0}; \gamma^*)$

V. Experiments

A. Experimental Setup

1) Evaluation Settings: We test our method on CIFAR-10 [62] and ImageNet [22] with the resolutions of 32×32 and 256×256. Unlike many previous methods [12], [18] that scale down the ImageNet images to 64×64, here we emphasize that we adopt the high-resolution version of ImageNet. Thus, our settings are more rigorous and realistic. Following previous gradient inversion works [9], [13], we adopt ResNet-18 [61] as our global model and utilize the same preprocessing procedures. We build up the search space and randomly generate the alternative models by arbitrarily changing the options of upsampling modules and skip connection patterns.

2) State-of-the-art Baselines for Comparison: We implement the following gradient inversion baselines: (1) *IG (Inverting Gradients)* [10]: pixel-level reconstruction with angle-based loss function; (2) *GI (GradInversion)* [17]: realizing batch-level restoration via multiple regularization priors; (3) *GGL (Generative Gradient Leakage)* [11]: employing strong GAN priors to produce high-fidelity images under severe defense strategies; (4) *GIAS (Gradient Inversion in Alternative Space)* [18]: searching the optimal latent code while optimizing in the generator parameter space; (5) *GIFD (Gradient Inversion over Feature Domain)* [12]: leveraging intermediate layer optimization for gradient inversion to further exploit pre-trained GAN priors; (6) *GION (Gradient Inversion via Over-parameterized Networks)* [13]: designing an over-parameterized convolutional network with excessive parameters and employing a fixed network architecture as implicit regularization. Note that when implementing GAN-based methods such as GGL and GIAS, we adopt BigGAN [63] pre-trained on ImageNet, which may result in mismatched priors when the target data is from CIFAR-10. This is because there is an inherent distribution bias between the ImageNet and CIFAR-10 datasets. However, such mismatched priors can be very common under realistic FL scenarios as mentioned in previous works [12], [13].

3) Quantitative Metrics: We utilize four metrics to measure the quality of reconstruction images: (1) *Peak Signal-to-Noise Ratio (PSNR)* ↑; (2) *Structural Similarity Index Measure (SSIM)* ↑; (3) *Feature Similarity Index Measure (FSIM)* ↑; (4) *Learned Perceptual Image Patch Similarity (LPIPS)* ↓ [64]. Note that "↓" indicates that the lower the metric, the better the attack performance while "↑" indicates that the higher the metric, the better the attack performance.

4) Implementation Details: The learning rate η in (7) is set as 1×10^{-3}. We utilize the signed gradient descent and

TABLE I

ATTACK RESULTS UNDER DIFFERENT VALUES OF SEARCH SIZE n. WE RANDOMLY SELECT IMAGES (STRICTLY HAVING NO INTERSECTIONS WITH THE IMAGES USED IN THE REMAINING EXPERIMENTS OF THIS PAPER) FROM IMAGENET WITH THE DEFAULT BATCH SIZE $B = 4$ FOR TESTING

Metric	1	10	50	100	500	1000	2000	3000	4000	**5000**	6000	7000	8000	9000
PSNR ↑	21.5034	21.6179	21.2776	22.0846	22.1451	22.5949	22.7380	23.3836	23.5626	**24.1603**	23.1646	23.3276	23.2485	22.9561
SSIM ↑	0.6314	0.5960	0.6222	0.6069	0.6099	0.6335	0.5960	0.6314	0.6424	**0.6566**	0.6518	0.6395	0.6324	0.6306
FSIM ↑	0.8281	0.7938	0.8019	0.8082	0.8125	0.8269	0.8028	0.8243	0.8337	**0.8439**	0.8334	0.8347	0.8319	0.8228
LPIPS ↓	0.4391	0.5238	0.5212	0.4808	0.4839	0.4540	0.4499	0.4369	0.4317	**0.4103**	0.4289	0.4281	0.4282	0.4913

TABLE II

ATTACK RESULTS UNDER DIFFERENT VALUES OF NETWORK DEPTH t. WE RANDOMLY SELECT IMAGES (STRICTLY HAVING NO INTERSECTIONS WITH THE IMAGES USED IN THE REMAINING EXPERIMENTS OF THIS PAPER) FROM IMAGENET WITH THE DEFAULT BATCH SIZE $B = 4$ FOR TESTING

Metric	3	4	**5**	6	7
PSNR ↑	21.3108	22.4132	**24.1603**	23.1865	22.9935
SSIM ↑	0.6114	0.6314	**0.6566**	0.6441	0.6293
FSIM ↑	0.8249	0.8314	**0.8439**	0.8317	0.8287
LPIPS ↓	0.4478	0.4263	**0.4103**	0.4263	0.4301

TABLE III

QUANTITATIVE COMPARISON OF GI-NAS TO STATE-OF-THE-ART GRADIENT INVERSION METHODS ON CIFAR-10 (32×32) AND IMAGENET (256×256) WITH THE DEFAULT BATCH SIZE $B = 4$

Dataset	Metric	IG	GI	GGL	GIAS	GIFD	GION	**GI-NAS**
CIFAR-10	PSNR ↑	16.3188	15.4613	12.4938	17.3687	18.2325	30.8652	**35.9883**
	SSIM ↑	0.5710	0.5127	0.3256	0.6239	0.6561	0.9918	**0.9983**
	FSIM ↑	0.7564	0.7311	0.6029	0.7800	0.8025	0.9960	**0.9991**
	LPIPS ↓	0.4410	0.4878	0.5992	0.4056	0.3606	0.0035	**0.0009**
ImageNet	PSNR ↑	7.9419	8.5070	11.6255	10.0602	9.8512	21.9942	**23.2578**
	SSIM ↑	0.0815	0.1157	0.2586	0.2408	0.2475	0.6188	**0.6848**
	FSIM ↑	0.5269	0.5299	0.5924	0.5719	0.5767	0.8198	**0.8513**
	LPIPS ↓	0.7194	0.7168	0.6152	0.6563	0.6561	0.4605	**0.3952**

adopt Adam optimizer [65] when updating the parameters of G_{opt}. We choose the negative cosine similarity function as the distance metric $\mathcal{D}(\cdot,\cdot)$ when calculating the gradient matching loss $\mathcal{L}_{grad}(\cdot)$ in (6). We conduct all the experiments on NVIDIA GeForce RTX 3090 GPUs for smaller batch sizes and on A100 GPUs for larger batch sizes.

B. Choices of Search Size n and Network Depth t

To obtain the best reconstruction performance, we need to carefully decide on the values of search size n and network depth t. Note that both these two hyper-parameters reflect the degree of NAS, and there is a trade-off between under-fitting and over-fitting when choosing their values. Thus, we randomly select images *(strictly having no intersections with the images used in the remaining experiments of this paper)* from ImageNet and test the attack results under different values of n and t with the default batch size $B = 4$. From Table I and Table II, we find that when n and t are small (e.g., $n = 100$ and $t = 3$), increasing any one of them indeed improves the attacks, as more optimal alternatives are provided or the models are more complicated to tackle the under-fitting. However, when n and t are large (e.g., $n = 7000$ and $t = 6$), further increasing any one of them cannot improve the attacks and will conversely lead to performance degradation due to the over-fitting. This is because candidates that focus too much on the minimal initial loss while neglecting the fundamental patterns of data recovery may be included in the search space and eventually selected. Thus, we adopt $n = 5000$ and $t = 5$ in the remaining experiments of this paper, as they strike a better balance and acquire the best results in Table I and Table II.

C. Comparison With State-of-the-Art Methods

Firstly, we compare GI-NAS with state-of-the-art gradient inversion methods on CIFAR-10. From Table III, we conclude that GI-NAS achieves the best results with significant performance improvements. For instance, we realize a 5.12 dB PSNR increase than GION, and our LPIPS value is 74.3% smaller than that of GION. These validate that in contrast to GION that optimizes on a fixed network, our NAS strategy indeed comes into effect and better leverages the implicit architectural priors.

We also discover that the GAN-based method GGL underperforms the previous GAN-free methods (i.e., IG and GI). This is because GGL utilizes BigGAN [63] pre-trained on ImageNet for generative priors in our settings, which has an inherent distribution bias with the target CIFAR-10 data domain. Besides, GGL only optimizes the latent vectors and cannot dynamically handle the mismatch between the training data of GAN and the target data. In contrast, GIAS optimizes both the latent vectors and the GAN parameters, which can reduce the distribution divergence and alleviate such mismatch to some extent. Therefore, although also GAN-based, GIAS exhibits much better performance than GGL on CIFAR-10.

1) High-Resolution Images Recovery: We then consider a more challenging situation and compare various methods on ImageNet with the resolution of 256×256. In Table III, most methods encounter significant performance decline when attacking high-resolution images. The amplification of image pixels greatly increases the complexity of reconstruction tasks and thus obstructs the optimization process for optimal images. But GI-NAS still achieves the best attack results, with a PSNR increase of 1.26 dB than GION.

2) Extension to Larger Batch Sizes: As shown in Table IV, we extend GI-NAS to larger batch sizes both on CIFAR-10 and ImageNet, which is more in line with the actual training process of FL systems. We observe that the performance of most methods degrades as batch sizes increase because it would be harder for the attackers to distinguish each individuals in a

Fig. 4. Qualitative comparison of GI-NAS to state-of-the-art gradient inversion methods on ImageNet (256 × 256) with the larger batch size $B = 32$.

TABLE IV

QUANTITATIVE COMPARISON OF GI-NAS TO STATE-OF-THE-ART GRADIENT INVERSION METHODS ON CIFAR-10 (32 × 32) AND IMAGENET (256 × 256) WITH LARGER BATCH SIZES WHEN $B > 4$

Dataset	Metric	Batch Size	IG	GI	GGL	GIAS	GIFD	GION	GI-NAS	Batch Size	IG	GI	GGL	GIAS	GIFD	GION	GI-NAS
CIFAR-10	PSNR ↑	16	10.2647	11.2593	9.9964	11.1952	11.6853	24.8746	**30.5315**	32	9.5802	10.5863	9.4104	9.8669	11.0898	28.6832	**33.0586**
	SSIM ↑		0.2185	0.2431	0.1658	0.2638	0.3222	0.9816	**0.9948**		0.1595	0.2020	0.1481	0.1724	0.2824	0.9892	**0.9974**
	FSIM ↑		0.5254	0.5350	0.4988	0.5569	0.6393	0.9869	**0.9965**		0.5186	0.5325	0.4995	0.5180	0.6015	0.9882	**0.9984**
	LPIPS ↓		0.6254	0.6225	0.6655	0.5804	0.5925	0.0177	**0.0035**		0.6598	0.6626	0.6922	0.6362	0.6305	0.0218	**0.0017**
	PSNR ↑	48	9.5446	10.2000	9.1200	9.6376	10.3709	29.5312	**38.7054**	96	9.2363	9.9500	8.8575	9.2982	10.3227	28.9723	**31.8026**
	SSIM ↑		0.1574	0.1697	0.1552	0.1652	0.2460	0.9932	**0.9983**		0.1402	0.1643	0.1516	0.2180	0.2389	0.9878	**0.9941**
	FSIM ↑		0.5112	0.5167	0.4869	0.5126	0.5824	0.9949	**0.9987**		0.4961	0.5238	0.4801	0.5365	0.5775	0.9865	**0.9941**
	LPIPS ↓		0.6660	0.6571	0.7213	0.6554	0.6504	0.0072	**0.0020**		0.6865	0.6593	0.7248	0.7297	0.6476	0.0227	**0.0071**
ImageNet	PSNR ↑	8	7.6011	8.3419	11.5454	9.6216	8.8377	20.4841	**20.8451**	16	7.7991	8.0233	11.4766	9.2563	8.8755	20.3078	**21.4267**
	SSIM ↑		0.0762	0.1113	0.2571	0.2218	0.2095	0.5681	**0.6076**		0.0704	0.0952	0.2561	0.2189	0.2193	0.5507	**0.6141**
	FSIM ↑		0.5103	0.5365	0.5942	0.5765	0.5726	0.7908	**0.8105**		0.5077	0.5165	0.5872	0.5832	0.5629	0.7840	**0.8157**
	LPIPS ↓		0.7374	0.7251	0.6152	0.6621	0.6701	0.5162	**0.4613**		0.7424	0.7362	0.6165	0.6656	0.6718	0.5313	**0.4613**
	PSNR ↑	24	7.1352	7.9079	10.9910	9.6601	9.2921	20.6845	**21.7933**	32	7.1085	7.8752	10.8987	10.0563	8.4252	16.1275	**20.0011**
	SSIM ↑		0.0549	0.0892	0.2583	0.2270	0.2421	0.5868	**0.6236**		0.0554	0.0763	0.2558	0.2361	0.2013	0.3415	**0.5485**
	FSIM ↑		0.4794	0.4943	0.5811	0.5618	0.5642	0.7978	**0.8163**		0.4940	0.4951	0.5889	0.5762	0.5545	0.6823	**0.7819**
	LPIPS ↓		0.7540	0.7475	0.6261	0.6667	0.6651	0.5219	**0.4651**		0.7493	0.7427	0.6296	0.6674	0.6853	0.6291	**0.5265**

TABLE V

QUANTITATIVE COMPARISON OF GI-NAS TO STATE-OF-THE-ART GRADIENT INVERSION METHODS ON CIFAR-10 (32 × 32) AND IMAGENET (256 × 256) UNDER VARIOUS DEFENSE STRATEGIES WITH THE DEFAULT BATCH SIZE $B = 4$

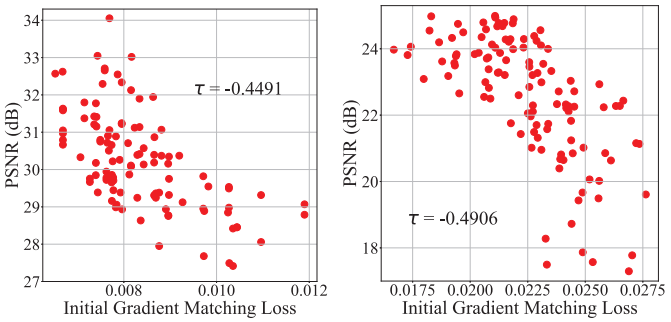
Dataset	Metric	Defense	IG	GI	GGL	GIAS	GIFD	GION	GI-NAS	Defense	IG	GI	GGL	GIAS	GIFD	GION	GI-NAS
CIFAR-10	PSNR ↑	Gaussian Noise	8.8332	8.2797	10.3947	9.8146	9.8441	9.7658	**10.5192**	Gradient Sparsification	9.6183	10.9486	10.4323	12.1328	14.0772	30.3518	**39.8739**
	SSIM ↑		0.1177	0.0976	**0.1856**	0.1301	0.1734	0.0280	0.0375		0.1592	0.2168	0.1862	0.3227	0.4492	0.9920	**0.9987**
	FSIM ↑		0.4844	0.4806	0.5123	0.4843	0.5481	0.5945	**0.6300**		0.5151	0.5329	0.5223	0.6015	0.6984	0.9951	**0.9987**
	LPIPS ↓		0.7258	0.7077	**0.6255**	0.7129	0.7330	0.6752	0.6740		0.6895	0.6855	0.6298	0.5582	0.5292	0.0047	**0.0014**
	PSNR ↑	Gradient Clipping	16.6203	11.5888	10.1373	17.9859	19.0230	32.4744	**36.2490**	Soteria	9.4103	10.2939	10.4217	15.5825	17.6231	31.1003	**34.1804**
	SSIM ↑		0.5952	0.2851	0.1760	0.6703	0.7050	0.9933	**0.9985**		0.1404	0.1707	0.1889	0.5405	0.6319	0.9900	**0.9973**
	FSIM ↑		0.7464	0.4976	0.5070	0.7751	0.8237	0.9966	**0.9992**		0.5086	0.5086	0.5144	0.7133	0.7834	0.9952	**0.9986**
	LPIPS ↓		0.3811	0.6413	0.6205	0.3054	0.3313	0.0029	**0.0008**		0.7008	0.6940	0.6335	0.3748	0.3654	0.0038	**0.0014**
ImageNet	PSNR ↑	Gaussian Noise	7.7020	7.4553	9.7381	6.8520	8.3526	9.2449	**11.2567**	Gradient Sparsification	8.6540	8.6519	9.8019	9.7894	9.8620	13.7358	**15.4331**
	SSIM ↑		0.0311	0.0197	**0.2319**	0.1045	0.1618	0.0277	0.0416		0.0669	0.0627	0.2315	0.2515	0.2424	0.2387	**0.3154**
	FSIM ↑		0.4518	0.3804	0.5659	0.4562	0.5091	0.5714	**0.5846**		0.5073	0.4696	0.5624	0.5664	0.5688	0.6365	**0.6786**
	LPIPS ↓		0.7775	0.8220	**0.6639**	0.7282	0.7085	0.8033	0.7805		0.7480	0.7681	0.6589	0.6604	0.6583	0.6909	**0.6535**
	PSNR ↑	Gradient Clipping	8.0690	7.2342	11.3215	9.7556	9.5759	22.2016	**23.2074**	Soteria	6.5675	6.6693	11.5690	9.7588	9.0612	22.9617	**23.7917**
	SSIM ↑		0.0881	0.0659	0.2531	0.2547	0.2390	0.6266	**0.7023**		0.0323	0.0382	0.2623	0.2425	0.2166	0.6698	**0.7143**
	FSIM ↑		0.5362	0.5434	0.5900	0.5794	0.5731	0.8229	**0.8635**		0.4929	0.4454	0.5964	0.5697	0.5738	0.8430	**0.8658**
	LPIPS ↓		0.7227	0.7230	0.6117	0.6485	0.6653	0.4603	**0.3743**		0.7506	0.7703	0.6031	0.6631	0.6660	0.4253	**0.3655**

batch, while GI-NAS is insusceptible to larger batch sizes and continuously generates high-quality images even at $B = 96$ on CIFAR-10 and $B = 32$ on ImageNet. Besides, GI-NAS is able to acquire consistent and significant performance gains

TABLE VI

CORRELATION INDICATORS BETWEEN VARIOUS TRAINING-FREE SEARCH METRICS AND THE ACTUAL PSNR PERFORMANCE ON CIFAR-10 (32×32) AND IMAGENET (256×256) WITH THE DEFAULT BATCH SIZE $B = 4$. NOTE THAT A CORRELATION INDICATOR CLOSER TO 1 (OR -1) INDICATES A STRONGER POSITIVE (OR NEGATIVE) CORRELATION

Training-Free Search Metric	CIFAR-10			ImageNet		
	Kendall's τ	Pearson Coefficient	Spearman Coefficient	Kendall's τ	Pearson Coefficient	Spearman Coefficient
Gaussian Noise $\mathcal{N} \sim (0, 1)$	0.0531	0.0948	0.0720	0.1187	0.1672	0.1773
Uniform Noise $\mathcal{U} \sim (0, 1)$	0.0885	0.0921	0.1291	0.0486	0.0822	0.0798
Initial Gradient Matching Loss	**-0.4491**	**-0.6128**	**-0.6345**	**-0.4906**	**-0.6633**	**-0.7120**



(a) CIFAR-10 (b) ImageNet

Fig. 5. Correlation between the initial gradient matching loss and the actual PSNR performance on CIFAR-10 (32×32) and ImageNet (256×256) with the default batch size $B = 4$.

on the basis of GION at all the given batch sizes. Figure 4 shows the qualitative comparison on ImageNet with the larger batch size $B = 32$. We notice that GI-NAS exactly realizes pixel-level recovery, while all the other compared methods struggle to perform the attacks and only obtain images with huge visual differences from the original ones. These results further provide evidence for the necessity and effectiveness of our batch-level optimal architecture search.

3) Attacks Under Defense Strategies: Next, we evaluate how these attacks perform when defenses are applied both on CIFAR-10 and ImageNet. Following previous works [9], [11], we consider four strict defense strategies: (1) *Gaussian Noise* [25] with a standard deviation of 0.1; (2) *Gradient Sparsification* [29] with a pruning rate of 90%; (3) *Gradient Clipping* [25] with a clipping bound of 4; (4) *Representative Perturbation (Soteria)* [31] with a pruning rate of 80%. For fair comparison, we apply the estimated gradient transformation $\mathcal{T}(\cdot)$ described in (6) to all the attack methods. From Table V, we discover that although the gradients have been disrupted by the imposed defense strategies, GI-NAS still realizes the best reconstruction effects in almost all the tested cases. The only exception is that GGL outperforms GI-NAS in terms of SSIM and LPIPS when the defense strategy is Gaussian Noise [25]. This is because the Gaussian noise with a standard deviation of 0.1 can severely corrupt the gradients and the information carried inside the gradients is no longer enough for private batch recovery. However, GGL only optimizes the latent vectors and can still generate natural images that possess some semantic features by the pre-trained GAN. Thus, GGL can still obtain not bad performance even though the generated images are quite dissimilar to the original ones.

TABLE VII

TIME COSTS (MINUTE) OF DIFFERENT METHODS AVERAGED OVER EACH BATCH ON CIFAR-10 (32×32) AND IMAGENET (256×256) WITH THE DEFAULT BATCH SIZE $B = 4$

Dataset	Metric	IG	GI	GGL	GIAS	GIFD	GION	GI-NAS
CIFAR-10	NAS Time	-	-	-	-	-	-	**6.6**
	Total Time	46.1	32.9	131.4	214.8	**30.2**	46.9	57.3
	PSNR	16.3188	15.4613	12.4938	17.3687	18.2325	30.8652	**35.9883**
ImageNet	NAS Time	-	-	-	-	-	-	**10.9**
	Total Time	291.1	273.7	244.6	343.2	70.6	**58.1**	84.4
	PSNR	7.9419	8.5070	11.6255	10.0602	9.8512	21.9942	**23.2578**

TABLE VIII

PSNR RESULTS OF GION AND DIFFERENT VARIANTS OF GI-NAS ON CIFAR-10 (32×32) AND IMAGENET (256×256) WITH THE DEFAULT BATCH SIZE $B = 4$

Method	Connection Search	Upsampling Search	CIFAR-10	ImageNet
GION	✗	✗	30.8652	21.9942
GI-NAS*	✗	✗	30.6514	22.0126
GI-NAS†	✓	✗	35.4336	22.4597
GI-NAS‡	✗	✓	35.3542	22.3992
GI-NAS	✓	✓	**35.9883**	**23.2578**

TABLE IX

ATTACK PERFORMANCE WHEN INCREASING THE TRAINING ITERATIONS FOR THE SEARCH METRIC ON IMAGENET (256×256) WITH THE DEFAULT BATCH SIZE $B = 4$. NOTE THAT THE TRAINING ITERATION OF 0 INDICATES THE USE OF OUR TRAINING-FREE SEARCH METRIC

Search Metric Training Iteration	PSNR ↑	SSIM ↑	FSIM ↑	LPIPS ↓
0 (Training-Free Search)	**23.2578**	**0.6848**	0.8513	**0.3952**
5	22.0149	0.6732	**0.8545**	0.3961
10	20.9854	0.6115	0.8393	0.4511
15	21.0370	0.6260	0.8389	0.4358

D. Further Analysis

1) Effectiveness of Our Training-Free Search Metric: We provide extensive empirical evidence that our training-free search metric (i.e., the initial gradient matching loss) highly negatively correlates with the real attack performance. Specifically, we attack the same private batch using models with different architectures and report the PSNR performance as well as the training-free search metric of each model both on CIFAR-10 and ImageNet. Following previous training-free

TABLE X

PSNR RESULTS OF GI-NAS AND STATE-OF-THE-ART GRADIENT INVERSION METHODS WHEN ATTACKING MORE FL GLOBAL MODELS (E.G., LENET-ZHU [9], CONVNET-32 [10]) ON IMAGENET (256×256) WITH THE DEFAULT BATCH SIZE $B = 4$

FL Global Model	IG	GI	GGL	GIAS	GIFD	GION	**GI-NAS**
LeNet-Zhu [9]	7.7175	6.9530	11.8195	16.5229	17.5620	19.2478	**21.1847**
ConvNet-32 [10]	10.9081	10.8886	10.4048	8.6696	24.3937	23.9602	**25.7824**
ResNet-50 [61]	10.9113	10.8914	9.5354	11.0040	9.3521	9.2080	**11.2320**
ViT-Base [69]	9.7616	8.4317	9.8555	9.9113	10.2516	11.5427	**12.6151**
ViT-Large [69]	9.6828	8.8500	9.8445	9.9554	10.3850	11.6536	**12.4354**
ViT-Huge [69]	10.3052	9.1382	10.5788	10.5919	10.4953	11.7839	**13.0380**

TABLE XI

RESULTS WHEN ATTACKING THE SAME BATCH BY DIFFERENT NETWORKS USING DIFFERENT PARAMETER INITIALIZATION SEEDS ON CIFAR-10 (32×32) AND IMAGENET (256×256) WITH THE DEFAULT BATCH SIZE $B = 4$. THE LATENT CODE INITIALIZATION SEEDS ARE FIXED FOR ALL THE MODELS. $G_{C1} \sim G_{C3}$ AND $G_{I1} \sim G_{I3}$ ARE RANDOMLY SAMPLED FROM THE MODEL SEARCH SPACE, WHILE $Seed_{C1} \sim Seed_{C3}$ AND $Seed_{I1} \sim Seed_{I3}$ ARE RANDOMLY SELECTED AND ARE DIFFERENT FROM EACH OTHER. NOTE THAT σ_{loss} AND σ_{psnr} ARE RESPECTIVELY THE STANDARD DEVIATIONS OF THE INITIAL GRADIENT MATCHING LOSSES AND PSNR RESULTS ON EACH NETWORK

Dataset	Model	Random Seed	Initial Gradient Matching Loss ($\times 10^{-3}$)	σ_{loss} ($\times 10^{-4}$)	PSNR	σ_{psnr}
CIFAR-10	G_{C1}	$Seed_{C1}$	8.5647	0.0012	28.7283	0.1211
		$Seed_{C2}$	8.5648		28.8082	
		$Seed_{C3}$	8.5645		28.5703	
	G_{C2}	$Seed_{C1}$	8.4321	0.0006	33.5848	0.2581
		$Seed_{C2}$	8.4323		33.4631	
		$Seed_{C3}$	8.4322		33.9584	
	G_{C3}	$Seed_{C1}$	7.9501	0.0015	49.4290	0.2936
		$Seed_{C2}$	7.9499		49.1171	
		$Seed_{C3}$	7.9502		49.7039	
ImageNet	G_{I1}	$Seed_{I1}$	21.0167	5.6866	22.9013	0.0329
		$Seed_{I2}$	22.1401		22.8912	
		$Seed_{I3}$	21.4250		22.8398	
	G_{I2}	$Seed_{I1}$	24.4600	2.2089	21.1475	0.3040
		$Seed_{I2}$	24.0722		21.6449	
		$Seed_{I3}$	24.0828		21.6989	
	G_{I3}	$Seed_{I1}$	18.5319	6.0290	23.5278	0.1884
		$Seed_{I2}$	18.3819		23.2974	
		$Seed_{I3}$	17.4208		23.6709	

NAS methods [54], [55], [56], we calculate the correlation indicators (i.e., Kendall's τ [66], Pearson Coefficient [67], Spearman Coefficient [68]) to quantitatively measure the relevance. To be more convincing, we introduce Gaussian Noise $\mathcal{N} \sim (0, 1)$ and Uniform Noise $\mathcal{U} \sim (0, 1)$ for comparison, where random noises sampled from $\mathcal{N} \sim (0, 1)$ and $\mathcal{U} \sim (0, 1)$ are regarded as the training-free search metrics. Through this comparative experiment design, we evaluate the performance when using our search method and randomly selecting structures based on Gaussian Noise and Uniform Noise. From Figure 5 and Table VI, we conclude that a smaller initial gradient matching loss is more likely to result in better PSNR performance, while the random noises are far from similar

TABLE XII

RESULTS WHEN ATTACKING THE SAME BATCH BY DIFFERENT NETWORKS USING DIFFERENT LATENT CODE INITIALIZATION SEEDS ON CIFAR-10 (32×32) AND IMAGENET (256×256) WITH THE DEFAULT BATCH SIZE $B = 4$. THE PARAMETER INITIALIZATION SEEDS ARE FIXED FOR ALL THE MODELS. $G_{C4} \sim G_{C6}$ AND $G_{I4} \sim G_{I6}$ ARE RANDOMLY SAMPLED FROM THE MODEL SEARCH SPACE, WHILE $Seed_{C4} \sim Seed_{C6}$ AND $Seed_{I4} \sim Seed_{I6}$ ARE RANDOMLY SELECTED AND ARE DIFFERENT FROM EACH OTHER. NOTE THAT σ_{loss} AND σ_{psnr} ARE RESPECTIVELY THE STANDARD DEVIATIONS OF THE INITIAL GRADIENT MATCHING LOSSES AND PSNR RESULTS ON EACH NETWORK

Dataset	Model	Random Seed	Initial Gradient Matching Loss ($\times 10^{-3}$)	σ_{loss} ($\times 10^{-4}$)	PSNR	σ_{psnr}
CIFAR-10	G_{C4}	$Seed_{C4}$	14.5338	0.0018	26.4506	0.1925
		$Seed_{C5}$	14.5335		26.2082	
		$Seed_{C6}$	14.5336		26.0703	
	G_{C5}	$Seed_{C4}$	13.4751	0.0018	30.1240	0.2675
		$Seed_{C5}$	13.4753		30.2045	
		$Seed_{C6}$	13.4755		30.6222	
	G_{C6}	$Seed_{C4}$	13.2704	0.0015	31.3139	0.0566
		$Seed_{C5}$	13.2701		31.4256	
		$Seed_{C6}$	13.2702		31.3540	
ImageNet	G_{I4}	$Seed_{I4}$	23.1931	4.6280	21.8376	0.1164
		$Seed_{I5}$	23.6335		22.0556	
		$Seed_{I6}$	22.7083		21.8760	
	G_{I5}	$Seed_{I4}$	19.5264	1.1102	23.6033	0.1889
		$Seed_{I5}$	19.5677		23.2439	
		$Seed_{I6}$	19.7360		23.3226	
	G_{I6}	$Seed_{I4}$	25.9479	3.6054	20.4730	0.1655
		$Seed_{I5}$	25.3497		20.8007	
		$Seed_{I6}$	25.9975		20.6776	

effects. Our method significantly outperforms the two random guessing methods based on Gaussian Noise and Uniform Noise. Thus, utilizing the initial gradient matching loss as the training-free search metric is reasonable and meaningful.

2) Computational Efficiency: We next analyze the computational costs. From Table VII, we find that NAS time only takes around 12% of Total Time, yet GI-NAS still achieves significant PSNR gains over GION and reaches state-of-the-art. The additional time costs introduced by our training-free NAS (6.6 or 10.9 minutes per batch) are very minor, especially given that many methods (e.g., GGL, GIAS) consume over 100 or even 200 minutes per batch. Therefore, such minor extra time costs in exchange for significant quantitative improvements are essential and worthwhile. Our method achieves a good trade-off between attack performance and computational efficiency.

3) Ablation Study on Search Type: In Table VIII, we report the performance of GION and different variants of GI-NAS. GI-NAS* optimizes on a fixed over-parameterized network with the search size $n = 1$, which means that it is essentially the same as GION. GI-NAS† only searches the skip connection patterns while GI-NAS‡ only searches the upsampling modules. We observe that the performance of GI-NAS* is very close to the performance of GION, as both

TABLE XIII

AVERAGED ARCHITECTURAL STATISTICS OF THE SEARCHED OPTIMAL MODELS BY OUR NAS STRATEGY UNDER VARIOUS SETTINGS TESTED IN SECTION V-C. THE DEFAULT SETTINGS ARE HIGHLIGHTED IN GRAY COLOR. $\Delta_{connection}$ AND $\Delta_{upsampling}$ ARE RESPECTIVELY THE GAINS OF THE NUMBER OF SKIP CONNECTIONS AND THE KERNEL SIZE OF UPSAMPLING MODULES WHEN COMPARED WITH THE DEFAULT SETTINGS

Dataset	Batch Size	Defense	Number of Skip Connections	Kernel Size of Upsampling Modules	$\Delta_{connection}$	$\Delta_{upsampling}$	Mostly Used Upsampling Operations
CIFAR-10	4	No Defense	11.4	4.8	0.0	0.0	bilinear
	16	No Defense	13.5	4.0	2.1	-0.8	bilinear
	32	No Defense	14.0	5.0	2.6	0.3	bicubic
	48	No Defense	13.0	5.0	1.6	0.3	bicubic
	96	No Defense	12.0	7.0	0.6	2.3	bicubic
	4	Gaussian Noise	12.3	5.5	0.9	0.8	bilinear
	4	Gradient Sparsification	11.8	4.5	0.4	-0.3	pixel shuffle
	4	Gaussian Clipping	12.5	5.0	1.1	0.3	bilinear
	4	Soteria	11.5	5.0	0.1	0.3	bicubic
ImageNet	4	No Defense	11.9	3.8	0.0	0.0	bilinear
	8	No Defense	12.8	5.5	0.9	1.8	bilinear
	16	No Defense	14.0	6.0	2.1	2.3	bilinear
	24	No Defense	16.0	5.0	4.1	1.3	bilinear
	32	No Defense	12.0	7.0	0.1	3.3	bicubic
	4	Gaussian Noise	12.3	5.0	0.4	1.3	pixel shuffle
	4	Gradient Sparsification	12.4	4.3	0.5	0.5	bilinear
	4	Gaussian Clipping	14.1	3.3	2.3	-0.5	bilinear
	4	Soteria	12.6	4.3	0.8	0.5	bilinear

of them adopt a changeless network architecture. GI-NAS† and GI-NAS‡ improve the attacks on the basis of GION or GI-NAS*, which validates the contribution of each search type. GI-NAS combines the above two search types and thus performs the best among all the variants.

4) Ablation Study on Search Metric Training Iteration: GI-NAS utilizes the initial gradient matching loss as the training-free search metric, which implies that its training iteration for each candidate is 0. To explore the rationality behind this design, we gradually increase the training iterations for the search metric. From Table IX, we conclude that further increasing the training iterations cannot improve the attacks and will instead increase the time costs. Hence, utilizing the initial gradient matching loss to find the optimal models is reasonable and computationally efficient.

5) Generalizability to More FL Global Models: We show the performance of different methods when attacking more FL global models (e.g., LeNet-Zhu [9], ConvNet-32 [10]) in Table X. We note that GI-NAS realizes the best reconstruction results in all the tested cases. These results further demonstrate the reliability of GI-NAS. We also observe significant variations in attack results across different FL global models, even when applying the same gradient inversion method (e.g., a 7.85 dB PSNR drop when switching from LeNet-Zhu [9] to ConvNet-32 [10] on GIAS). This implies that future study may have a deeper look into the correlation between the gradient inversion robustness and the architecture of FL global model, and thus design more securing defense strategies from the perspective of adaptive network architecture choice.

6) Robustness to Network Parameters Initialization: Following the general paradigm of previous training-free NAS methods [54], [55], [56] in other areas, the parameters of all the NAS candidates are randomly initialized and there is no special need to tune the parameters before calculating the initial gradient matching losses. Moreover, GI-NAS records both the architectures as well as the initial parameters when finding the optimal models. Therefore, for the model we decide on, its initial gradient matching loss in the NAS time (i.e., Stage 1 as shown in Figure 2) is the same as its initial gradient matching loss in the optimization time (i.e., Stage 2 as shown in Figure 2), and the influences of parameters initialization can be further reduced. To further demonstrate the robustness, we randomly select models with different architectures and apply different parameter initialization seeds to them to attack the same batch. From Table XI, we find that for the same architecture, both the initial gradient matching losses and the final PSNR results stay within a steady range when changing the parameter initialization seeds. However, when the architectures change, both the initial gradient matching losses and the final PSNR results will vary significantly. The impacts of different architectures are much greater than the impacts of different parameter initialization seeds. As a result, we conclude that GI-NAS is stable and insensitive to network parameters initialization. We also observe that the models with smaller initial gradient matching losses often have better final PSNR results in Table XI, which is consistent with our aforementioned findings in Figure 5 and Table VI. This again validates the rationality and usefulness of adopting the initial gradient matching loss as the training-free search metric.

7) Robustness to Latent Codes Initialization: The latent code $\mathbf{z_0}$ is randomly sampled from the Gaussian distribution $\mathcal{N}(0, 1)$ and then kept frozen. Thus, we explore the influences of different latent codes by changing their initialization seeds. From Table XII, we discover that although different latent code initialization seeds on the same architecture may result in different initial gradient matching losses, the differences

TABLE XIV

QUANTITATIVE COMPARISON OF GI-NAS TO STATE-OF-THE-ART GRADIENT INVERSION METHODS ON FFHQ (64 × 64), MNIST (28 × 28), AND SVHN (32 × 32) WITH THE DEFAULT BATCH SIZE $B = 4$

Dataset	Metric	IG	GI	GGL	GIAS	GIFD	GION	GI-NAS
FFHQ	PSNR ↑	12.1571	12.4204	10.8660	13.2693	13.7942	31.7174	**36.2522**
	SSIM ↑	0.2226	0.2237	0.1958	0.3418	0.3609	0.9865	**0.9954**
	FSIM ↑	0.6212	0.6128	0.6099	0.6623	0.6763	0.9879	**0.9979**
	LPIPS ↓	0.7448	0.7516	0.6327	0.6160	0.6182	0.0212	**0.0025**
MNIST	PSNR ↑	15.1129	14.1863	9.3894	22.8035	28.8421	31.9038	**37.4773**
	SSIM ↑	0.3497	0.3220	0.0670	0.5962	0.7874	0.9935	**0.9953**
	FSIM ↑	0.5319	0.5115	0.3957	0.7520	0.8959	0.9982	**0.9993**
	LPIPS ↓	0.3898	0.4425	0.5735	0.2240	0.0923	0.0014	**0.0005**
SVHN	PSNR ↑	20.9779	20.9475	16.9157	21.0760	22.8974	44.8053	**46.4780**
	SSIM ↑	0.6702	0.7047	0.4427	0.6704	0.7894	0.9774	**0.9870**
	FSIM ↑	0.7993	0.8121	0.6535	0.8045	0.8531	0.9895	**0.9941**
	LPIPS ↓	0.5145	0.5247	0.6789	0.5148	0.4270	0.0045	**0.0028**

are still very minor (less than 10^{-6} for CIFAR-10 and less than 10^{-3} for ImageNet), and the impacts on the final PSNR results are limited (less than 0.5 dB for CIFAR-10 and less than 0.4 dB for ImageNet). However, when using different architectures, both these two values will change dramatically, and the differences are much greater than those brought by changing the latent code initialization seeds. Hence, GI-NAS is also robust to latent codes initialization.

8) Preferences of NAS Outcomes: To figure out the implications behind NAS preferences, we trace back the searched optimal models from Section V-C and analyze their averaged architectural statistics in Table XIII. Note that the bicubic operation takes more neighbourhood pixels than the bilinear operation (16 > 4) and is thus considered to be more complex. Besides, the pixel shuffle operation rearranges lots of pixels and is also considered to be more complicated than the bilinear operation. From Table XIII, we observe that the optimal models are largely dependent on recovery difficulties. Generally, batches with larger sizes or under defenses are more difficult to recover. Compared to the default settings (highlighted in gray color) that are with the batch size $B = 4$ and under no defenses, the more difficult settings would prefer models with higher complexity (e.g., more skip connections, larger kernel sizes, more complex operations). The complexity gains $\Delta_{connection}$ and $\Delta_{upsampling}$ are often greater than 0 when the settings are $B > 4$ or under defenses. Easier batches prefer simpler, lighter architectures and harder batches prefer more complex, heavier architectures. This might be used to further improve the efficiency. By considering the difficulties ahead of time, we can have a better estimation of the complexity of optimal models. Thus, we only need to focus on models with a certain type of complexity and skip other unrelated models even when there is not enough time for NAS.

VI. CONCLUSION

In this paper, we propose GI-NAS, a novel gradient inversion method that makes deeper use of the implicit architectural priors for gradient inversion. We first systematically analyze existing gradient inversion methods and emphasize the necessity of adaptive architecture search. We then build up our model search space by designing different upsampling modules and skip connection patterns. To reduce the computational overhead, we leverage the initial gradient matching loss as the training-free search metric to select the optimal model architecture and provide extensive experimental evidence that such a metric highly correlates with the real attack performance. Extensive experiments show that GI-NAS can achieve state-of-the-art performance compared to existing methods, even under more practical FL scenarios with high-resolution images, large-sized batches, and advanced defense strategies. We also provide deeper analysis on various aspects, such as time costs, ablation studies, generalizability to more FL global models, robustness to network parameters initialization, robustness to latent codes initialization, and implications behind the NAS searched results. We hope that the remarkable attack improvements of GI-NAS over existing gradient inversion methods may help raise the public awareness of such privacy threats, as the sensitive data would be more likely to be revealed or even abused. Moreover, we hope that the idea of this paper may shed new light on the gradient inversion community and facilitate the research in this field.

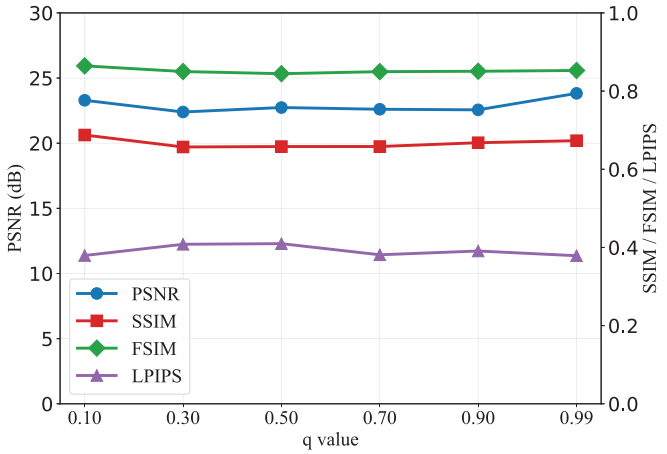


Fig. 6. Attack performance of GI-NAS when changing the values of q (the probability of assigning a training example with a certain label to a particular client) on ImageNet (256 × 256) with the default batch size $B = 4$. Note that there are $N_{client} = 10$ clients in total, which indicates that if $q = \frac{1}{10}$, the distribution is IID. These settings strictly adhere to the settings in previous works [77], [78] considering data heterogeneity.

A. Limitations and Future Directions

Although we have empirically demonstrated that the initial gradient matching loss is highly negatively correlated to the final attack performance, it is still of great significance to further prove the effectiveness of this search metric with rigorous theoretical analysis. In the future work, we hope to provide more insightful theoretical results with regard to this search metric from various perspectives, such as the frequency spectrum [60], [70] or the implicit neural architectural priors [21], [71]. Furthermore, inspired by the effectiveness of NAS for attacks in this paper, a promising defensive direction involves exploring victim-side neural architecture search to discover model structures inherently more resistant to gradient inversion while maintaining model accuracy, or developing

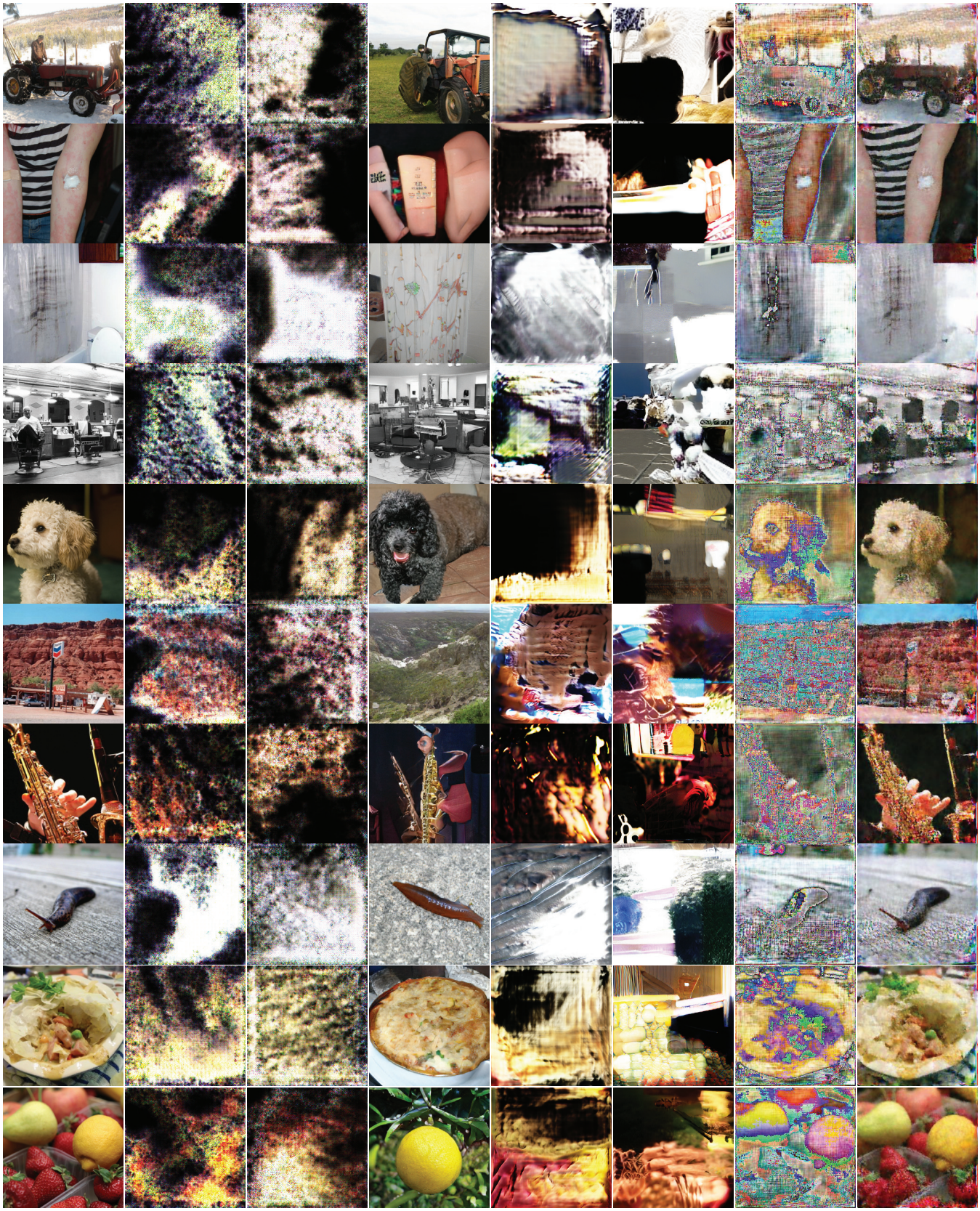


Fig. 7. More qualitative results of GI-NAS and state-of-the-art gradient inversion methods on ImageNet (256×256) with the larger batch size $B = 32$.

adaptive gradient perturbation techniques on the FL global model guided by NAS principles.

APPENDIX A. ATTACK RESULTS ON MORE DATASETS

We further compare the attack effectiveness of GI-NAS with other methods on more datasets, including FFHQ [72], MNIST [73], and SVHN [74]. These datasets are also considered in previous gradient inversion methods [12], [75]. From Table XIV, we observe that GI-NAS consistently outperforms all the compared methods across diverse datasets. This consistent superiority suggests that GI-NAS is a highly generalizable and effective method for privacy attacks in federated learning.

APPENDIX B. ATTACK RESULTS ON HETEROGENEOUS DATA

FL systems would commonly face the problem of heterogeneous data [76]. Thus, to further validate the robustness of our method, we strictly align with the prior works [77], [78] to design the data heterogeneity experiments. To be specific, we randomly allocate the classes of datasets into N_{client} clients with a probability q of assigning a training example with a certain label to a particular client and a probability $\frac{1-q}{N_{client}-1}$ of assigning it to other clients. When $q = \frac{1}{N_{client}}$, the client's local training data are independent and identically distributed (IID), otherwise the client's local training data are Non-IID.

We set $N_{client} = 10$, which means that if $q = \frac{1}{10}$, the distribution is IID. We select $q = 0.10, 0.30, 0.50, 0.70, 0.90, 0.99$, and test the performance of GI-NAS under these settings. From Figure 6, we notice that the attack performance of GI-NAS stays within a steady range when changing the values of q. Therefore, we conclude that GI-NAS is also robust to the influences of data heterogeneity.

APPENDIX C. MORE VISUALIZED RESULTS

We provide more visualized comparisons in Figure 7. We observe that all the other methods struggle to conduct the reconstruction attacks, while our proposed GI-NAS again realizes the best performance in terms of visual fidelity.

REFERENCES

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, vol. 54, A. Singh and J.Zhu., Eds., Fort Lauderdale, FL, USA, 2017, pp. 1273–1282.

[2] K. Wei et al., "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 3454–3469, 2020.

[3] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," *Knowl.-Based Syst.*, vol. 216, Jan. 2021, Art. no. 106775.

[4] J. Li, C. Zhang, Y. Zhao, W. Qiu, Q. Chen, and X. Zhang, "Federated learning-based short-term building energy consumption prediction method for solving the data silos problem," *Building Simul.*, vol. 15, no. 6, pp. 1145–1159, Jun. 2022.

[5] L. Melis, C. Song, E. D. Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 691–706.

[6] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 739–753.

[7] L. Lyu, H. Yu, and Q. Yang, "Threats to federated learning: A survey," 2020, *arXiv:2003.02133*.

[8] X. Luo, Y. Wu, X. Xiao, and B. C. Ooi, "Feature inference attack on model predictions in vertical federated learning," in *Proc. IEEE 37th Int. Conf. Data Eng. (ICDE)*, Apr. 2021, pp. 181–192.

[9] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 14774–14774.

[10] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients-how easy is it to break privacy in federated learning?," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 16937–16947.

[11] Z. Li, J. Zhang, L. Liu, and J. Liu, "Auditing privacy defenses in federated learning via generative gradient leakage," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 10132–10142.

[12] H. Fang, B. Chen, X. Wang, Z. Wang, and S.-T. Xia, "GIFD: A generative gradient inversion method with feature domain optimization," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 4944–4953.

[13] C. Zhang et al., "Generative gradient inversion via over-parameterized networks in federated learning," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 5103–5112.

[14] S. Saeidian, G. Cervia, T. J. Oechtering, and M. Skoglund, "Quantifying membership privacy via information leakage," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 3096–3108, 2021.

[15] K. Ganju, Q. Wang, W. Yang, C. A. Gunter, and N. Borisov, "Property inference attacks on fully connected neural networks using permutation invariant representations," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2018, pp. 619–633.

[16] B. Zhao, K. Reddy Mopuri, and H. Bilen, "IDLG: Improved deep leakage from gradients," 2020, *arXiv:2001.02610*.

[17] H. Yin, A. Mallya, A. Vahdat, J. M. Alvarez, J. Kautz, and P. Molchanov, "See through gradients: Image batch recovery via gradinversion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2021, pp. 16337–16346.

[18] J. Jeon, J. Kim, K. Lee, S. Oh, and J. Ok, "Gradient inversion with generative image prior," in *Proc. Adv. Neural Inf. Process. Syst.*, Jan. 2021, pp. 29898–29908.

[19] I. Goodfellow et al., "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, Oct. 2020.

[20] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative adversarial networks: An overview," *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 53–65, Jan. 2018.

[21] V. Lempitsky, A. Vedaldi, and D. Ulyanov, "Deep image prior," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9446–9454.

[22] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Miami, FL, USA, Jun. 2009, pp. 248–255.

[23] Y. Huang, S. Gupta, Z. Song, K. Li, and S. Arora, "Evaluating gradient inversion attacks and defenses in federated learning," in *Proc. Annu. Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2021, pp. 7232–7241.

[24] R. Zhang, S. Guo, J. Wang, X. Xie, and D. Tao, "A survey on gradient inversion: Attacks, defenses and future directions," 2022, *arXiv:2206.07284*.

[25] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," 2017, *arXiv:1712.07557*.

[26] W. Wei, L. Liu, Y. Wut, G. Su, and A. Iyengar, "Gradient-leakage resilient federated learning," in *Proc. IEEE 41st Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2021, pp. 797–807.

[27] T. Zhu, P. Xiong, G. Li, and W. Zhou, "Correlated differential privacy: Hiding information in non-IID data set," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 2, pp. 229–242, Feb. 2015.

[28] K. Wei et al., "Personalized federated learning with differential privacy and convergence guarantee," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 4488–4503, 2023.

[29] A. Fikri Aji and K. Heafield, "Sparse communication for distributed gradient descent," 2017, *arXiv:1704.05021*.

[30] Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally, "Deep gradient compression: Reducing the communication bandwidth for distributed training," in *Proc. Int. Conf. Learn. Represent.*, Jan. 2017, pp. 1–14.

[31] J. Sun, A. Li, B. Wang, H. Yang, H. Li, and Y. Chen, "Soteria: Provable defense against privacy leakage in federated learning from representation perspective," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2021, pp. 9311–9319.

[32] M. Suganuma, M. Özay, and T. Okatani, "Exploiting the potential of standard convolutional autoencoders for image restoration by evolutionary search," in *Proc. Int. Conf. Mach. Learn.*, Jul. 2018, pp. 4771–4780.

[33] X. Chu, B. Zhang, H. Ma, R. Xu, and Q. Li, "Fast, accurate and lightweight super-resolution with neural architecture search," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 59–64.

[34] V. Nekrasov, H. Chen, C. Shen, and I. Reid, "Fast neural architecture search of compact semantic segmentation models via auxiliary cells," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9118–9127.

[35] C. Liu et al., "Auto-DeepLab: Hierarchical neural architecture search for semantic image segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 82–92.

[36] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," in *Proc. Int. Conf. Learn. Represent.*, Jan. 2016, pp. 1–16.

[37] H. Qin, C. Fan, S. Deng, Y. Li, M. A. El-Yacoubi, and G. Zhou, "AG-NAS: An attention GRU-based neural architecture search for finger-vein recognition," *IEEE Trans. Inf. Forensics Security*, vol. 19, pp. 1699–1713, 2024.

[38] F. Wang and D. M. J. Tax, "Survey on the attention based RNN model and its applications in computer vision," 2016, *arXiv:1601.06823*.

[39] H. Liu, K. Simonyan, O. Vinyals, C. Fernando, and K. Kavukcuoglu, "Hierarchical representations for efficient architecture search," 2017, *arXiv:1711.00436*.

[40] R. Miikkulainen et al., "Evolving deep neural networks," in *Proc. Artif. Intell. Age Neural Netw. Brain Comput.*, Oct. 2023, pp. 269–287.

[41] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *Proc. 32nd Int. Conf. Mach. Learn.*, Jun. 2015, pp. 2342–2350.

[42] J. Bergstra, D. Yamins, and D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *Proc. 30th Int. Conf. Mach. Learn.*, vol. 28, S. Dasgupta and D. McAllester, Eds., Atlanta, GA, USA: PMLR, Feb. 2013, pp. 115–123.

[43] H. Mendoza, A. Klein, M. Feurer, J. T. Springenberg, and F. Hutter, "Towards automatically-tuned neural networks," in *Proc. Workshop Autom. Mach. Learn.*, Dec. 2016, pp. 58–65.

[44] T. Domhan, J. T. Springenberg, and F. Hutter, "Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves," in *Proc. 24th Int. Joint Conf. Artif. Intell.*, Jul. 2015, pp. 3460–3468.

[45] B. Baker, O. Gupta, N. Naik, and R. Raskar, "Designing neural network architectures using reinforcement learning," 2016, *arXiv:1611.02167*.

[46] H. Cai, T. Chen, W. Zhang, Y. Yu, and J. Wang, "Efficient architecture search by network transformation," in *Proc. AAAI Conf. Artif. Intell.*, vol. 32, Apr. 2018, pp. 2787–2794.

[47] Z. Zhong, J. Yan, W. Wu, J. Shao, and C.-L. Liu, "Practical block-wise neural network architecture generation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2423–2432.

[48] S. Xie, H. Zheng, C. Liu, and L. Lin, "SNAS: Stochastic neural architecture search," 2018, *arXiv:1812.09926*.

[49] H. Cai, L. Zhu, and S. Han, "ProxylessNAS: Direct neural architecture search on target task and hardware," 2018, *arXiv:1812.00332*.

[50] K. Ahmed and L. Torresani, "MaskConnect: Connectivity learning by gradient descent," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 349–365.

[51] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 229–256, May 1992.

[52] C. J. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 279–292, May 1992.

[53] J. Clifton and E. B. Laber, "Q-learning: Theory and applications," *Annu. Rev. Statist. Appl.*, vol. 7, no. 1, pp. 279–301, Mar. 2020.

[54] Q. Zhou et al., "Training-free transformer architecture search," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2022, pp. 10894–10903.

[55] Q. Zhou et al., "Training-free transformer architecture search with zero-cost proxy guided evolution," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 10, pp. 6525–6541, Oct. 2024.

[56] A. Serianni and J. Kalita, "Training-free neural architecture search for RNNs and transformers," in *Proc. 61st Annu. Meeting Assoc. Comput. Linguistics*, 2023, pp. 2522–2540.

[57] J. Zhu and M. Blaschko, "R-GAP: Recursive gradient attack on privacy," 2020, *arXiv:2010.07733*.

[58] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. 18th Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.*, vol. 9351. Cham, Switzerland: Springer, 2015, pp. 234–241.

[59] Y. Chen, C. Gao, E. Robb, and J. Huang, "NAS-DIP: Learning deep image prior with neural architecture search," in *Proc. ECCV*, Jan. 2020, pp. 442–459.

[60] M. E. Arican, O. Kara, G. Bredell, and E. Konukoglu, "ISNAS-DIP: Image-specific neural architecture search for deep image prior," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 1950–1958.

[61] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[62] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Master's thesis, Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, 2009.

[63] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," in *Proc. Int. Conf. Learn. Represent.*, Jan. 2018, pp. 1–35.

[64] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 586–595.

[65] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[66] M. G. Kendall, "A new measure of rank correlation," *Biometrika*, vol. 30, no. 1, p. 81, Jun. 1938.

[67] G. Nahler, "Pearson correlation coefficient," in *Noise Reduction in Speech Processing*. Berlin, Germany: Springer, Jan. 2009, p. 132.

[68] C. Spearman, "The proof and measurement of association between two things," *Int. J. Epidemiology*, vol. 39, no. 5, pp. 1137–1150, Oct. 2010.

[69] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," in *Proc. Int. Conf. Learn. Represent.*, Jan. 2020, pp. 1–21.

[70] Y. Liu, J. Li, Y. Pang, D. Nie, and P.-T. Yap, "The devil is in the upsampling: Architectural decisions made simpler for denoising with deep image prior," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 12374–12383.

[71] Z. Cheng, M. Gadelha, S. Maji, and D. Sheldon, "A Bayesian perspective on the deep image prior," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5438–5446.

[72] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4401–4410.

[73] L. Deng, "The MNIST database of handwritten digit images for machine learning research [Best of the Web]," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 141–142, Nov. 2012.

[74] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *Proc. NIPS Workshop Deep Learn. Unsupervised Feature Learn.*, Jan. 2011, p. 4.

[75] H. Liang, Y. Li, C. Zhang, X. Liu, and L. Zhu, "EGIA: An external gradient inversion attack in federated learning," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 4984–4995, 2023.

[76] M. Ye, X. Fang, B. Du, P. C. Yuen, and D. Tao, "Heterogeneous federated learning: State-of-the-art and research challenges," *ACM Comput. Surveys*, vol. 56, no. 3, pp. 1–44, Mar. 2024.

[77] J. Wei et al., "Extracting private training data in federated learning from clients," *IEEE Trans. Inf. Forensics Security*, vol. 20, pp. 4525–4540, 2025.

[78] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "FLTrust: Byzantine-robust federated learning via trust bootstrapping," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2021, pp. 1–18.