AFGI: Towards Accurate and Fast-convergent Gradient Inversion Attack in Federated Learning

Can Liu, Jin Wang,*, *Member, IEEE,* Yipeng Zhou, *Member, IEEE* Yachao Yuan, Quanzheng Sheng, *Member, IEEE* and Kejie Lu, *Senior Member, IEEE*

***Abstract*—Federated learning (FL) empowers privacy-preservation in model training by only exposing users' model gradients. Yet, FL users are susceptible to gradient inversion attacks (GIAs) which can reconstruct ground-truth training data such as images based on model gradients. However, reconstructing high-resolution images by existing GIAs faces two challenges: inferior accuracy and slow-convergence, especially when duplicating labels exist in the training batch. To address these challenges, we present an Accurate and Fast-convergent Gradient Inversion attack algorithm, called AFGI, with two components: Label Recovery Block (LRB) which can accurately restore duplicating labels of private images based on exposed gradients; VME Regularization Term, which includes the total variance of reconstructed images, the discrepancy between three-channel means and edges, between values from exposed gradients and reconstructed images, respectively. The AFGI can be regarded as a white-box attack strategy to reconstruct images by leveraging labels recovered by LRB. In particular, AFGI is efficient that accurately reconstruct ground-truth images when users' training batch size is up to 48. Our experimental results manifest that AFGI can diminish 85% time costs while achieving superb inversion quality in the ImageNet dataset. At last, our study unveils the shortcomings of FL in privacy-preservation, prompting the development of more advanced countermeasure strategies.**

***Index Terms*—Federated learning, gradient inversion attacks, label recovery, privacy.**

I. Introduction

THE Federated Learning (FL) framework was originally proposed by Google [1] for protecting user privacy. In FL, users only expose updated model gradients to the parameter server (PS) for completing model training. Meanwhile, original training data is privately retained by users on local devices to preserve privacy. In FL, the model is trained in an iterative manner, coordinated by the PS. Specifically, the PS distributes the latest global model to a set of selected users at the beginning of each global iteration. Then, selected users train the model with their local data for a few round of local iterations, and then return model gradients or weights back to the PS. The PS aggregates these gradients to update the global model. The above process will be repeated until the global model converges [2]–[14].

However, recent studies have unveiled that it is unrealistic to protect data privacy by FL alone if the PS is honest-but-curious. According to [7]–[16][1], gradients can leak local data privacy such that ground-truth data can be reconstructed if the honest-but-curious PS launches gradient inversion attacks (GIAs) based on collected user gradients. To date, there are mainly two types of GIAs: iteration-based GIAs [7]–[12], [15] and recursion-based GIAs [13], [14]. Iteration-based GIAs reconstruct images by minimizing the distance between gradients of the ground-truth image, denoted by ∇W generated from the image x, and gradients of the reconstructed image, denoted by $\nabla W'$ generated by the reconstructed image $\hat{x}$ [7]–[12], [15]. In essence, iteration-based GIAs' strategies employ various loss functions and auxiliary regularization terms to iteratively minimize the difference between ∇W and $\nabla W'$, continuously updating $\hat{x}$ through model backpropagation [7]–[12], [15]. In particular, the ROG strategy [15] achieved the state-of-the-art results on the ImageNet dataset [17]. ROG utilizes the training outcomes of the encoder, decoder, and pre-trained network as regularization. By narrowing the search range for image reconstruction, it can effectively reconstruct the ground-truth data given that FL trains LeNet, VGG-7, or ResNet-18 models [18]–[20]. Recursion-based GIAs, on the other hand, exploit the relationship between input data, model parameters, and gradients [13], [14] for reconstructing private data. Yet, these strategies are more focused on the trivial case that the batch size of data for local training is 1. Due to this limitation, most GIAs strategies are iteration-based methods because these strategies are applicable in more generic cases.

Nonetheless, reconstructing ground-truth data such as images is non-trivial. There are at least two reasons hindering its practical implementation. First, ground-truth images x and their label y information are invisible in FL. As a result, GIAs algorithms when reconstructing data slowly converge and probably diverge due to the lack of accurate critical information, *e.g.*, labels. For example, it takes 8.5 hours to reconstruct a single ImageNet image under the context that the batch size is 1 by running the strategy in [9] with a single NVIDIA V100 GPU. Considering that a typical FL user may own hundreds of images [2], [3], implementing GIAs for reconstructing so many images is prohibitive. Second, compli-

* Corresponding author: Jin Wang, Email: wjin1985@suda.edu.cn

Can Liu is with the Department of Computer Science and Technology, Soochow University, Suzhou, China.

Jin Wang and Yachao Yuan are with the School of Future Science and Engineering, Soochow University, Suzhou, China.

Yipeng Zhou and Quanzheng Sheng are School of Computing, Macquarie University, Sydney, Australia.

Kejie Lu is with the Department of Computer Science and Engineering, University of Puerto Rico at Mayagüez, Puerto Rico, USA.

[1]the paper [16] is the previous version of the paper we published on the arXiv. This article has been modified and improved on this basis.

TABLE I
VARIOUS ITERATION-BASED GIAS WITHIN HORIZONTAL FEDERATED LEARNING FRAMEWORKS

Methods	GI Types	Image Initialization	Extra Terms Number	Loss Function	Image Resolution	Maximum Batch size	Label Restore	Label Assume
DLG [7]	Iter.	random	0	ℓ_2	64×64	8		-
iDLG [8]	Iter.	random	0	ℓ_2	64×64	8	✓	No-repeat
GGI [9]	Iter.	random	1	cosine	224×224	8		Known
CPL [22]	Iter.	red or green	1	ℓ_2	128×128	8		-
GradInversion [10]	Iter.	random	6	ℓ_2	224×224	48	✓	No-repeat
HGI [12]	Iter.	random	3	ℓ_2	-	8		-
ROG [15]	Iter.	Image	1	ℓ_2	128×128	16		-
AFGI (Ours)	**Iter.**	**gray**	**3**	**cosine**	**224×224**	**48**	✓	**None**

cated FL contexts can substantially deteriorate attack accuracy. As reported in [9], [10], attack accuracy is inferior when the training batch size is much grater than 1 or duplicating labels exist in the batch.

To improve the practicability of GIAs, we propose an **A**ccurate and **F**ast-convergent iteration-based **G**radient **I**nversion attack algorithm, called **AFGI**, with the **LRB** (**L**abel **R**ecovery **B**lock) component and the **VME** (total **V**ariance, three-channel **M**ean and **E**dge detection regularization terms). **LRB** is workable under complicated FL training tasks, *i.e.*, a training batch contains multiple samples with duplicated labels. Specifically, inspired by [10], we exploit the column-wise sum of the last fully connected (FC) layer gradients to serve as the input to the **LRB**, which can enhance the discrimination between repeated and non-repeated labels in the output probability matrix, and thereby facilitate the identification of repeated labels. Then, **VME** reconstructs ground-truth data x based on exposed gradients and labels y produced by **LRB** with a much faster convergence rate due to the following two reasons. First, **LRB** can provide more accurate label information $\hat{y}$, which can effectively avoid learning divergence when reconstructing data. Second, we introduce the three-channel mean of an image to correct the color of $\hat{x}$ in each channel which is more efficient than existing works using the total mean of an image. Then, we introduce the canny edge detection, which is widely used in image subject recognition [21], as a regularization term in **VME** such that we can more accurately recover subject positions in data reconstruction with fewer iterations.

Our experimental results unequivocally demonstrate that **LRB** significantly with up to 20% improvement of label recovery accuracy. **VME** can not only accurately reconstruct ground-truth data but also considerably reduce the time cost. In particular, **AFGI** saves 85% time costs compared to the baseline strategy [9]. **AFGI** excels on reconstructing the widely used ImageNet dataset [17], and faithfully restore individual images at a resolution of 224×224 pixels. **AFGI** attack is still valid even when the batch size is up to 48 images.

Our main contributions are as follows:

- We use part of the training model to build a new network **LRB** for label recovery. The improved label recovery accuracy with **LRB** in the pre-trained ResNet-50 model, without the strong assumption of no-repeated labels, surpasses previous GIAs strategies. More accurate labels help **AFGI** achieve faster convergence in the iteration process.
- We introduce two novel regularization terms in **VME** to accelerate model convergence and reduce time costs. The three-channel mean helps correct the color of $\hat{x}$ and and the edge detection regularization term helps align the position of subjects in $\hat{x}$ with those in the ground truth.
- The reconstructed images of **AFGI** outperform state-of-the-art GIAs strategies. We compare three metrics in our experiments, and **AFGI** performs better in most metrics.

In the following sections, we briefly discuss recent works relevant to **AFGI** in Section II. The detailed **AFGI** process is elaborated in Section III. In Section IV, we present the results of numerous experiments. Finally, the conclusion and future directions are discussed in Section V.

II. RELATED WORK

In this section, we first briefly introduce the classification of federated learning in Section II-A. Then, we comprehensively survey existing gradient inversion attack strategies in Section II-B and introduce the canny edge detection in the Section II-C.

A. Federated Learning

FL can be classified into three main types based on data and feature distribution modes: horizontal federated learning (HFL), vertical federated learning (VFL), and federated transfer learning (FTL) [23], [24]. HFL, which has been extensively investigated, involves users sharing identical data features but exhibiting different data distributions [25]. Conversely, VFL applies when FL users share the same data distribution but own different features, with notable applications in healthcare and finance [26], [27]. FTL addresses the construction of the FL framework when both data characteristics and distribution differ [28], [29]. Among these types, HFL is the most studied one.

B. Gradient Inversion Attacks (GIAs)

Recent studies on GIAs have primarily focused on iteration-based methods in HFL. Zhu et al. [7] propose Deep Leakage from Gradients (DLG) to reconstruct ground-truth data x on small datasets using the ResNet-56 model by minimizing the ℓ_2 distance between reconstructed gradients $\nabla W'$ and ground-truth gradients ∇W. However, DLG initializes with random dummy data $\hat{x}$ and labels, $\hat{y}$, which can lead to convergence failures and poor reconstructed results. To address this issue,

iDLG [8] utilizes the negative cross-entropy value of ∇W to obtain the high accuracy reconstructe labels $\hat{y}$. Geiping et al. [9] propose GGI, which uses cosine similarity as the cost function and adds total variation regularization to accelerate convergence. GGI assumes that labels can be obtained via cross-entropy, which requires 192,000 (192K) iterations to reconstruct high-quality and stable images. User Privacy Leakage (CPL) [22] effectively minimizes the ℓ_2 distance between $\nabla W'$ and ∇W by adding label-based regularization to ensure model convergence, particularly for a batch size of 1. GradInversion [10] introduces five regularization terms to accurately position objects in $\hat{x}$. Building on GradInversion, NVIDIA developed the Divide-and-Conquer Inversion (DCI) [11] algorithm, incorporating new regularization terms under Generative Adversarial Networks (GANs), and then applied these techniques to achieve excellent results with X-ray images [12]. Reconstructed image data from Obfuscated Gradient (ROG) [15] uses an encoder and decoder to accelerate model convergence under shallow training models. Although these strategies achieve visualized reconstructed images in single-label datasets, they incur heavy time costs and rely on the unreliable assumption of batch-norm regularization [10]–[12], [30]. These strategies use the negative cross-entropy value to determine labels in GIAs. However, cross-entropy is ineffective for the acquisition of images' multi-label information. For a succinct overview, a comparative analysis of these strategies is presented in Table I. Images restored by GGI with a batch size of 100 pose difficulty in identification. If the 8th column is empty, it means that the paper does not propose a new label acquisition strategy. The symbol '-' denotes instances where no specific description is provided in the corresponding paper.

In addition to iteration-based strategies, recursive-based methods such as R-GAP [13] reconstruct inputs of each model layer from the last layer by solving linear equations to obtain the optimal solution with the minimal error. COPA [14] provides a deeper understanding of the objective function and underscores the limitations of various models. However, both R-GAP and COPA concentrate on image restoration in single-label datasets with the batch size of 1.

C. Canny Edge Detection

In the context of image classification, it is crucial to detect all subjects present in an image [31]–[33]. Various edge detection algorithms have been proposed for this purpose [**?**], [21], [34]. Among these, the canny edge detection method stands out as a standard approach and remains widely used in computer vision research. Notably, it is recognized for its ability to handle gradient directions, incorporate non-maximum suppression, and utilize two thresholds to define image subject edges. Consequently, the canny edge detection method can assist GIAs in identifying subject edges, thereby potentially accelerating GIAs model convergence.

III. AFGI Algorithm Design

In this section, we discuss the attack model and the workflow of **AFGI**. Furthermore, we elaborate the label recovery and GIAs strategies in **AFGI**, namely **LRB** and **VME**.

TABLE II
IMPORTANT SYMBOLS AND DEFINITIONS

Symbol	Definition
PS	the parameter server
x, y	the ground-truth data, label
$\hat{x}$, $\hat{y}$	the reconstructed data, label
∇W, $\nabla W'$	the gradient of x and $\hat{x}$
$\mathbf{R}$	the regularization terms
α	the scaling factors
G_1, G_2	the summation and variance of ∇W
pro_{LRB}	label probability vector after **LRB**
pro'_{LRB}	pro_{LRB} in a descending order
K	the training batch size
N	the number of image classes
L_k	labels obtained from GradInversion
L_{LRB}	repeated labels obtained from **LRB**
MI	the maximum number of iterations
lr	the learning rate

A. Attack Model

In view of the prevalence of HFL [7]–[10], our study focuses on GIAs within the HFL context. We assume that data are attacked in white-box scenarios [35], [36]. This attack context comprises an honest-but-curious PS and N users u_i, where $i \in \{0, 1, 2, ..., N-1\}$. The PS operates according to FL framework principles but may store data during model training. It has knowledge of the model architecture, parameters, and users' uploaded gradients which is the white-box scenario [37], [38]. Consequently, the PS can construct a threat model identical to the training model and employ iteration-based GIAs to invert users' local data through model backpropagation. In more complex training tasks, *i.e.*, when the training batch size is greater than 1, users' privacy is compromised even if only one training image is reconstructed. The import symbols in this paper are listed in Table II.

B. Overview of AFGI

Inspired by prior research [8]–[10], we design **AFGI** with two novel components: **LRB**, which aims to accurately restore labels of private data by using exposed gradients, and **VME**, which aims to reconstruct ground-truth image data efficiently.

- **LRB**: A notable advantage of **LRB** lies in that labels can be repeating within one local training batch. In contrast, prior works [9]–[11] commonly assume that no repeating labels exist in the same local training batch. In LRB, it leverages a convolutional network (by reusing the last few layers of the FL task model) to enhance the discrimination between repeating and non-repeating labels, and hence facilitate the identification of repeating labels.
- **VME**: Private images are reconstructed by the PS based on recovered label information and the model backpropagation algorithm such that $\hat{x}$ can be as close to x as possible. The **VME** reguarization term is involved to improve both the reconstruction accuracy and speed.

C. Label Recovery Block (LRB)

Provided that the training batch size is K, where $K \in \{0, 1, ..., N\}$, our task is to recover K labels contained in

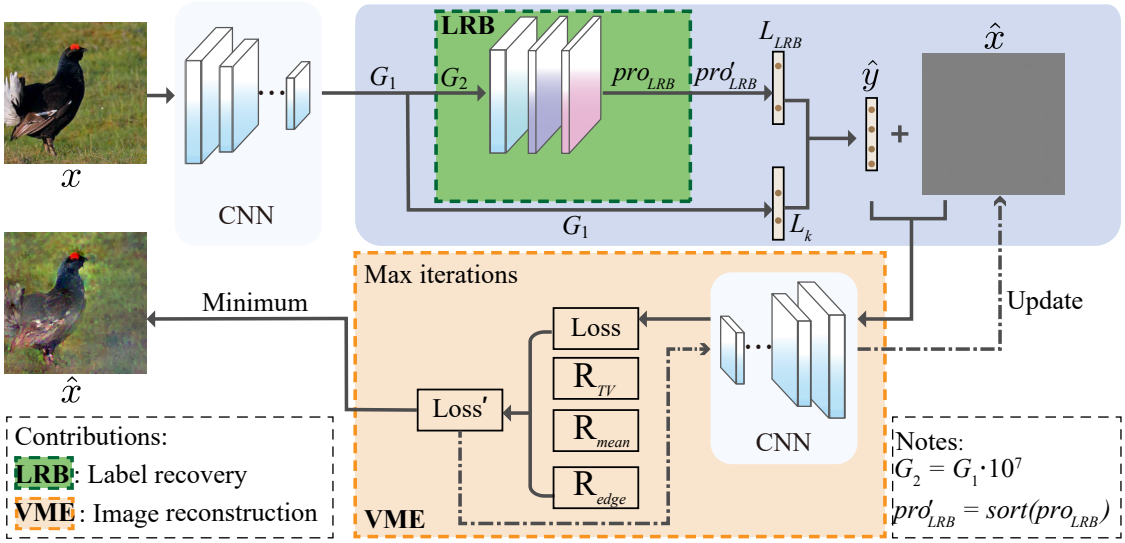


Fig. 1. The workflow of **AFGI**. The initialization of $\hat{x}$ is a gray image and $\hat{y}$ is derived from two sources as G_1 (L_k) and the output of **LRB** (L_{LRB}). The cosine similarity loss function with three regularization terms to compute the loss and gradients values. Finally, the $\hat{x}$ with the minimum loss value is closed to the ground-truth image.

a user's local training batch. The process to recover labels has two major steps: extracting initial labels (denoted by a sequence L_k) , and obtaining duplicating labels (denoted by a sequence L_{LRB}) through **LRB**. Note that both L_k and L_{LRB} are sorted sequences by a descending order of probabilities of these labels contained in the training batch.

Prior works such as GradInversion [10] can extract labels assuming that there is no repeating labels, which can be reused by our works to extract initial labels L_k in the first step. Specifically, it extracts L_k from negative cross-entropy values. This method involves selecting the label corresponding to the minimum K gradient values from the last FC layer. The GradInversion label recovery method is briefly outlined as follows:

$$\hat{y} = argsort(\min_{M} \nabla W^{(FC)}_{M,N} \mathcal{L}(x, y))[:K], \tag{1}$$

where $\min_M \nabla_{WM,N^{(FC)}} \mathcal{L}(x, y)$ represents the minimum ∇W value obtained from the last FC layer along the feature dimension (equivalently by rows). Here, M is the number of embedded features and N is the number of image classes. The function $argsort(\cdot)[:K]$ is utilized to retrieve the indices of the minimum K values along a column. Finally, it outputs a sorted label sequence L_k and a sorted probability sequence p_{init}. Elements in L_k are sorted by a descending order of their probabilities. p_{init} indicates the corresponding probability of each element. For example, $L_k = (c_1, c_2, c_3)$ indicates that recovered labels include c_1, c_2 and c_3. If the corresponding probability of each label in the training batch is 0.9, 0.8, 0.7, respectively, it implies that $p_{init} = (0.9, 0.8, 0.7)$.

Note that there is no repeating labels in L_k. Given the possibility of duplicating labels in the local training batch in practice, it is likely that the number of elements in L_k is less than K, i.e., the number of samples in the training batch. It is represented by $\mid L_k \mid = k \leq K$, where $\mid \cdot \mid$ denotes the number of elements in $\cdot$. Here, $k < K$ indicates that there exists $K-k$ repeating labels in the training batch.

The second step involves obtaining the duplicating label sequence, denoted by L_{LRB}, where $\mid L_{LRB} \mid = K - k$. We design **LRB** as an auxiliary structure for acquiring these duplicating labels. **LRB** consists of the last block, the last average pool (AvgPool) layer, and the last fully connected (FC) layer of the FL training model. **LRB** shares the same model parameters as the corresponding layers of the model trained by FL. In previous label classification tasks [39]–[41], labels are typically identified as the largest values in the model output. Following this intuition, if label is contained in the training batch, the output of **LRB** for this label should be higher than that of other labels.

LRB recovers duplicating labels by leveraging exposed gradients through the following three substeps:

- **Substep 1:** The input to **LRB** is the variance of exposed gradients ∇W. Initially, $G_1 \in \mathbb{R}^{1\times h}$ is defined as the summation of gradients of the last fully connected (FC) layer in a row-wise manner, where h is the number of output_channel before the last FC layer. For convenience, we multiply G_1 by a large positive integer, e.g., 10^7, to get G_2, which will not alter the numerical relationships between gradients. Then, the dimension of G_2 is expanded from $\mathbb{R}^{1\times h}$ to $\mathbb{R}^{1\times h\times 1\times 1}$ to fit the input of **LRB**, which is a common operation in computer vision for adjusting the dimensions of images [42], [43]. Using the expanded G_2 as input of **LRB**, a label probability sequence $p_{LRB} \in \mathbb{R}^{1\times N}$ is produced, where N represents the number of image classes, corresponding to the number of outputs in the last FC layer of the trained FL model.
- **Substep 2:** To identify labels contained in a local training batch, we sort p_{LRB} in a descending order. We select a label from L_{LRB} and insert it to L_k if and only if the

Algorithm 1 Label Restoration in **LRB**

Input: G_1: the column-wise summation of the last FC layer gradients; **LRB**$(\cdot)$: propagation in the LRB; K: the training batch size.

Parameter: G_2: the result of the linear transformation of G_1; $|\cdot|$: the number of elements in a vector or matrix; Con: a large constant; N: the total number of classes; $sort(\cdot)$: sort $\cdot$ in decreasing order; $site(\cdot)$: obtain the index of $\cdot$; $sum(\cdot)[-1]$: summation along the column-wise; $in(\cdot)$: check if an element exists in the vector; L_k:the reconstructed labels $\hat{y}$ from G_1 with $|L_k|$ as k; L_{LRB}: the $\hat{y}$ from **LRB**; pro_{LRB}: the probability of labels after **LRB**; pro'_{LRB}:the result of the order $proLRB$; γ: the threshold of the label probability gap; $merge(\cdot)$: concatenate two vectors into one vector.

Output: $\hat{y}$: the reconstructed labels.

1: $G_2 = G_1 \times Con$
2: $|L_{LRB}| = 0$
3: $|L_k| = k$
4: $pro_{LRB} = sum(LRB(G_2))[-1]$
5: $pro'_{LRB} = sort(pro_{LRB})$
6: $id_{mid} = site(pro'_{LRB})$
7: **if** $k < K$ **then**
8: $G_2 = G_1 \times Con$
9: $|L_k| = 0$
10: $|L_{LRB}| = 0$
11: **for** i = 0, ..., N-1 **do**
12: **if** $G_1[i] < 0$ **then**
13: $L_k \leftarrow arg\ sort(G_1[i])$
14: **end if**
15: **end for**
16: $|L_k| = k$
17: $pro_{LRB} = sum(LRB(G_2))[-1]$
18: $pro'_{LRB} = sort(pro_{LRB})$
19: $id_{mid} = site(pro'_{LRB})$
20: **if** $k < K$ **then**
21: **for** i = 0, ..., N-1 **do**
22: **if** $id_{mid}[i]$ *in* L_k *and* $(pro'_{LRB}[id_{mid}[i]] - pro'_{LRB}[id_{mid}[i+1]]) > \gamma$ **then**
23: $L_{LRB} \leftarrow id_{mid}[i]$.
24: **end if**
25: $k = k + 1$
26: **end for**
27: **end if**
28: **end if**
29: **if** $|L_{LRB}| < K - k$ **then**
30: $L_{LRB} \leftarrow L_k[m]$
31: **end if**

following two conditions are met. First, the label is an element in L_k. The second condition depends on p_{LRB}. Supposing that the label is the n-th element in p_{LRB}. It will be selected if $p_{LRB}[n] - p_{LRB}[n+1] > \gamma$, where γ is an empirically determined threshold and $[n]$ indicates the n-th largest element in a sequence. According to our experiments in Section IV, γ is set as 0.4. The intuition is that a label is repeating only if its probability in the training batch is much greater than other candidate labels.

- **Substep 3:** It is possible that $|\ L_{LRB}\ |$ is still less than $K - k$ after Substep 2. We further duplicate top $K - k$ labels in L_k. Here, top labels are selected based on their probabilities in p_{init}. In this case, we choose the label with the highest probability value as the duplicating label. After these steps, we obtain a reconstructed label sequence $\hat{y}$ in increasing order, which contains K labels.

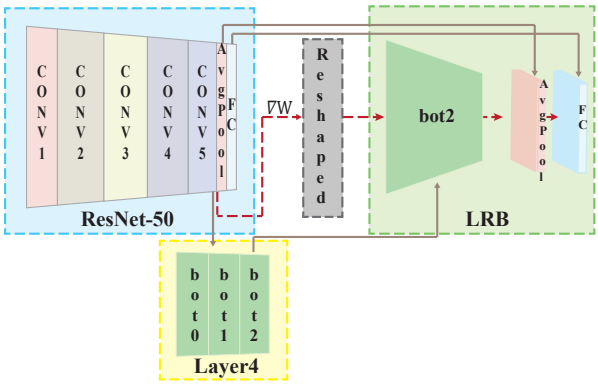


Fig. 2. The layers in the LRB are based on the ResNet-50 model.

The label recovery process is visually illustrated in the green box in Fig. 1, with detailed layers shown in Fig. 2, and the pseudocode provided in Algorithm 1. Lines 4-8 extract L_k, while lines 10-19 manage the L_{LRB} procedure. Ultimately, the merge of L_k and L_{LRB} ensures that the restoration of labels aligns with the batch size K. The repeated label reconstruction requires only one propagation of **LRB**. The computational complexity of **LRB** is $\mathcal{O}((K-k) \cdot N)$.

D. Regularization Terms VME

We aim to reduce time costs and enhance the quality of $\hat{x}$ in white-box attack scenarios. To achieve this goal, we update $\hat{x}$ using model backpropagation, regularized by **VME**. Here, recovered labels $\hat{y}$ and initialized gray images are employed as inputs. According to ablation experiments in GradInversion [10], the absence of regularization terms makes bad results in $\hat{x}$. To improve the quality of $\hat{x}$, we introduce three regularization terms, i.e., total **v**ariance, three-channel **m**ean discrepancy, and **e**dge detection, into the **AFGI** objective function:

$$\hat{x} = \underset{\hat{x},\hat{y}}{argmin}\, 1 - cos(\nabla W', \nabla W) + \mathbf{R}_{reg}, \tag{2}$$

where

$$\cos(\nabla W', \nabla W) = \frac{< \nabla W, \nabla W' >}{\|\nabla W\| \|\nabla W'\|}, \tag{3}$$

and

$$\mathbf{R}_{reg} = \alpha_{TV}\mathbf{R}_{TV} + \alpha_{mean}\mathbf{R}_{mean} + \alpha_{ed}\mathbf{R}_{ed}. \tag{4}$$

Here, Equation 2 employs the cosine similarity distance (denoted by $cos(\cdot)$ in Equation 3) to measure the gap between

TABLE III
THE MOST POPULAR IMAGE DATASETS IN COMPUTER VISION

Dataset	Mean values of three-channel
ImageNet	[0.485, 0.456, 0.406]
CIFAR-10	[0.491, 0.482, 0.447]
CIFAR-100	[0.507, 0.487, 0.441]
PASCAL VOC 2012	[0.485, 0.456, 0.406]
MS COCO	[0.485, 0.456, 0.406]
Average	**[0.491, 0.467, 0.421]**

$\nabla W'$ and ∇W, following the previous work [9]. The computational complexity of Equation 3 is $\mathcal{O}(d)$, where d is the dimension of the image data matrix. When Equation 2 approaches 0, $\hat{x}$ approximates x. In Section IV, we will empirically compare different loss functions to demonstrate that $cos(\cdot)$ is a better choice as the loss function in our problem.

The regularization term $\mathbf{R}_{reg}$ in **VME** comprises three components: $\mathbf{R}_{TV}$, $\mathbf{R}_{mean}$, and $\mathbf{R}_{ed}$. $\mathbf{R}_{TV}$ penalizes the total variance of $\hat{x}$ and has been proven effective in numerous prior works [9], [10]. Our novelty lies in introducing $\mathbf{R}_{mean}$ and $\mathbf{R}_{ed}$.

Mean regularization term $\mathbf{R}_{mean}$: The design of $\mathbf{R}_{mean}$ is inspired by our measurement study on publicly available image datasets. As shown in Table III, we measure three-channel average of images in most popular image datasets, including ImageNet, CIFAR-10, CIFAR-100, PASCAL VOC 2012, and Microsoft Common Objects in Context (MS COCO). From Table III, we can observe that the means of three-channel of these image datasets are very close to each other. Thereby, it is reasonable to assume that the mean of a reconstructed image should not be far away from these mean values. The average mean value of these image datasets is calculating in the last row. By averaging these mean values, we use [0.491, 0.467, 0.421] as the prior-knowledge to construct the term for regularizing the objective function. In other words, the ℓ_2 distance between the three-channel mean of a reconstructed image $\hat{x}$ and [0.491, 0.467, 0.421] should be as small as possible.

Edge regularization term $\mathbf{R}_{ed}$: Prior works [9], [10] have shown that the position of a subject in $\hat{x}$ may deviate from the counterpart in x. To address this problem, $\mathbf{R}_{group}$ is introduced into the objective function of GradInversion [10] to calculate the average of positions of a subject from different initial seeds, which can alleviate the significant deviation. $\mathbf{R}_{group}$ achieves a better quality of $\hat{x}$ through the initialization of more random seeds. However, using $\mathbf{R}_{group}$ can heavily increase the time cost of GIAs.

To overcome the heavy time cost drawback of group regularization, we introduce a superseded regularization term called edge regularization ($\mathbf{R}_{ed}$) based on exposed gradients and the canny edge detection algorithm. Since x is invisible and we can only access ∇W for deriving the edge information, we modify the original canny edge detection algorithm accordingly, and the workflow to obtain $\mathbf{R}_{ed}$ is as follows:

- A dynamic threshold fin related to the maximum and mean values of the gradient before the last FC layer G ($max(G)$, $mean(G)$) is set as:

$$fin = (max(G) - mean(G)) \times \beta. \tag{5}$$

  We utilize the average value ($mean(\cdot)$) to filter out gradients that are extremely small. The parameter β takes the value range for capturing gradients of all edges into account.
- Obtain a set of G matrix coordinates ed_{fin} where the values of ∇W are greater than fin.
- Scale G according to the dimension of x to obtain pixel positions $ed_{fin-reg}$ corresponding to ed_{fin}. Finally, choose the middle pixel position ed_{reg} of $ed_{fin-reg}$ as the x baseline point.
- Apply the canny edge detection algorithm on $\hat{x}$ during each backpropagation iteration with two thresholds of θ_1 and θ_2, obtaining a set of edges denoted as ed_t. As there is a significant gap between $\hat{x}$ and x in the early stage of iterations, set these two thresholds to be sufficiently large to reduce the impact of non-edge pixels. Then, select the middle pixel coordinate $\hat{ed}_{reg}$ from ed_t as the baseline point for $\hat{x}$.
- Calculate ℓ_2 distance between ed_{reg} and $\hat{ed}_{reg}$, penalizing significant differences

Algorithm 2 Edge regularization term $\mathbf{R}_{ed}$

Input: G: the gradient before the last FC layer; $CA(\hat{x}_i, \theta_1, \theta_2)$: canny edge detection of image $\hat{x}$ with two thresholds θ_1 and θ_2.
Parameter: $max(\cdot)$: the maximum value of $\cdot$; $mean(\cdot)$: the mean value of $\cdot$; β: the threshold to select gradients from G; fin: the threshold to obtain edges from G; $ed_{fin-reg}$: the pixel site set where the gradient value is larger than fin; $|\cdot|$: the number of elements in a vector or matrix; img_{row}, img_{col}: the ground-truth image matrix dimensions; $index(\cdot)$: the index of $\cdot$.
Output: R_{ed}: the value of $\mathbf{R}_{ed}$.

1: $fin = (max(G) - mean(G)) \times \beta$
2: **for** i $from$ 0 to $|G|$ **do**
3: **if** $G[i] > fin$ **then**
4: $ed_{fin} \leftarrow G[i]$
5: **end if**
6: **end for**
7: **for** j $from$ 0 to $|ed_{fin}|$ **do**
8: $ed_{fin-reg} \leftarrow [\lfloor ed_{fin}[j] \times img_{row}/|ed_{fin}| \rfloor, \lfloor ed_{fin}[j] \times img_{col}/|ed_{fin}| \rfloor]$
9: **end for**
10: $ed_{reg} \leftarrow ed_{fin-reg}[1/2 \times |ed_{fin}|]$
11: $ed_t \leftarrow CA(\hat{x}, \theta_1, \theta_2)$
12: $\hat{ed}_{reg} \leftarrow ed_t[1/2 \times |ed_t|]$
13: $R_{ed} \leftarrow \|ed_{reg} - \hat{ed}_{reg}\|_2$
14: **return** R_{ed}

The pseudocode for the process to obtain $\mathbf{R}_{ed}$ is presented in Algorithm 2 and **??**. Lines 1-6 depict the process of obtaining edges from ground-truth gradients, lines 7-9 represent the workflow for the base-point of x, while lines 10-12 obtain $\hat{x}$

Algorithm 3 Regularization Terms **VME**

Input: ∇W: the ground-truth gradients; $\nabla W'$: the reconstructed image gradients; $\hat{y}$: the reconstructed labels; MI: the maximum iteration times; $\mathbf{R}_{TV}$: the total variance regularization term; $\mathbf{R}_{mean}$: the three-channel mean regularization term; $\mathbf{R}_{ed}$: the canny edge detection regularization term; α_{TV}: the scaling factor for $\mathbf{R}_{TV}$; α_{mean}: the scaling factor for $\mathbf{R}_{mean}$; α_{ed}: the scaling factor for $\mathbf{R}_{ed}$.
Parameter: $gray$: gray image; $cos(\cdot)$: the cosine similarity function; $\mathbf{R}_{reg}$: the value of three regularization terms; $argmin(\cdot)$: get the minimum value of $\cdot$; $M_{back}(\cdot,\cdot)$: the backpropagation of the model.
Output: $\hat{x}$: the reconstructed image.

1: $\hat{x} = gray$
2: **for** i $from$ 0 to $MI-1$ **do**
3: $\quad \mathbf{R}_{reg} = \alpha_{TV}\mathbf{R}_{TV} + \alpha_{mean}\mathbf{R}_{mean} + \alpha_{ed}\mathbf{R}_{ed}$
4: $\quad \hat{x} = \underset{\hat{x},\hat{y}}{argmin}\, 1 - cos(\nabla W', \nabla W) + \mathbf{R}_{reg}$
5: $\quad M_{back}(\hat{x}, \hat{y})$
6: $\quad$ update $\hat{x}$
7: **end for**
8: **return** $\hat{x}$

edges, and line 13 is the ℓ_2 distance calculation process. The result of $\mathbf{R}_{ed}$ is used to penalize the gap between x and $\hat{x}$.

IV. Experiments

In this section, we first elaborate the settings of experimental environment and hyperparameters. Next, we introduce three evaluation metrics used for the subsequent quantitative comparison of recovered $\hat{x}$. Following this, we compare the accuracy of $\hat{y}$ to demonstrate the superior precision of **LRB**. Finally, we present both visual and quantitative comparisons of $\hat{x}$ between **AFGI** and other typical iteration-based GIA strategies.

A. Experiments Details

We utilize the pre-trained ResNet-50 model as the attack model, implementing the **AFGI** strategy to invert images of size 224×224 pixels from the validation set of the ImageNet ILSVRC 2021 dataset [17]. This setup aligns with that of GradInversion [10]. In the **AFGI** experiment, we conFig.d hyperparameters by setting $\alpha_{TV} = 1e-1$, $\alpha_{mean} = 1e-3$, $\alpha_{ed} = 1e-2$, and the learning rate $lr = 1e-2$ during the maximum iterations $MI = 10K$ at a batch size of 1. The parameters with $\alpha = 0.6$, $\theta_1 = 0.8$ and $\theta_2 = 0.9$ are used in $\mathbf{R}_{ed}$. Fine-tuning is then performed for additional 10K iterations with a batch size larger than 1. The lr decreases with a factor of 0.2 after every 2/7 of the total iterations in our experiments. The most time-saving and cost-effective setting for **AFGI** is to restart only once, while GGI [9] requires eight restarts. The inversion process of **AFGI** is accelerated using one NVIDIA A100 GPU, coupled with the Adam optimizer.

TABLE IV
Label reconstruction accuracy on the ImageNet dataset validation set

Batch size	iDLG [8]	GradInversion [10]	**LRB (Ours)**
1	**100%**	**100%**	**100%**
2	78.45%	87.15%	**98.90%**
4	68.05%	80.50%	**91.95%**
8	58.11%	75.29%	**87.01%**
16	53.42%	72.09%	**83.95%**
32	48.85%	68.85%	**80.96%**
64	46.07%	66.48%	**79.24%**

B. Evaluation Metrics

In this paper, we employ three key metrics for evaluating the quality of $\hat{x}$ from three perspectives, which can overcome the limitations of evaluation relying on a single metric.

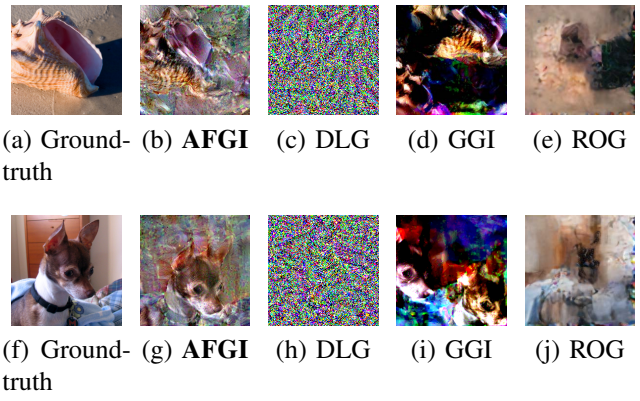
- Peak Signal-to-Noise Ratio (PSNR ↑): This metric measures the ratio of the peak signal strength to the noise level, providing insights into the fidelity of $\hat{x}$. A higher PSNR value indicates a closer match to x.
- Structural Similarity (SSIM ↑): SSIM assesses the structural similarity between x and $\hat{x}$. A higher SSIM value signifies a more faithful reproduction of the structural details in x.
- Learned Perceptual Image Patch Similarity (LPIPS ↓): LPIPS quantifies perceptual similarity between x and $\hat{x}$ by considering learned image features. A lower LPIPS value indicates a higher perceptual similarity to x.
- Time cost (↓): The time cost is obtained by counting the GPU time consumed by the strategy. The less time cost, the faster the convergence speed of the strategy model and the higher the efficiency.

C. Label Restoration

In label restoration experiments, we compare **LRB** with the state-of-the-art methods, including GradInversion [10] and iDLG [8]. The latter one was also employed in GGI [9]. Images in each training batch are uniformly sampled, with the initial image index randomly chosen from 1 to 20K, and a uniform random number between 1 and 100 serving as the increment for the next image index. Label accuracy results are summarized in Table IV. Notably, **LRB** consistently outperforms iDLG and GradInversion across all batch sizes. Specifically, with a batch size of 2, **LRB** achieves 98.90% accuracy, remarkably surpassing iDLG by more than 20% and outperforming GradInversion by over 11%.

TABLE V
Label reconstruction using **LRB** under different γ

Batch size	0.3	**0.4**	0.5
1	**100%**	**100%**	**100%**
2	98.90%	**98.90%**	98.75%
4	91.78%	**91.95%**	90.98%
8	**87.25%**	87.01%	86.69%
16	83.57%	**83.95%**	83.83%
32	**81.18%**	80.96%	81.07%
64	79.23%	79.24%	**79.29%**



(a) Ground-truth (b) **AFGI** (c) DLG (d) GGI (e) ROG

(f) Ground-truth (g) **AFGI** (h) DLG (i) GGI (j) ROG

Fig. 3. Comparing the results of $\hat{x}$ under the batch size of 1. Figures (a) and (f) depict ground-truth images x. Figures (b), (c), (d), (e), (g), (h), (i), and (j) showcase $\hat{x}$ under different GIAs strategies.

TABLE VI
THE METRIC VALUES OF FIG. 3

	(b)	(c)	(d)	(e)
PSNR ↑	17.47	3.55	10.07	**19.26**
SSIM ↑	**0.0572**	0.0156	0.0137	0.0417
LPIPS ↓	**0.4909**	1.4242	0.6788	0.6044
	(g)	(h)	(i)	(j)
PSNR ↑	15.87	3.56	11.60	**17.87**
SSIM ↑	**0.1183**	0.0143	0.0378	0.0485
LPIPS ↓	**0.5762**	1.4843	0.6813	0.5970

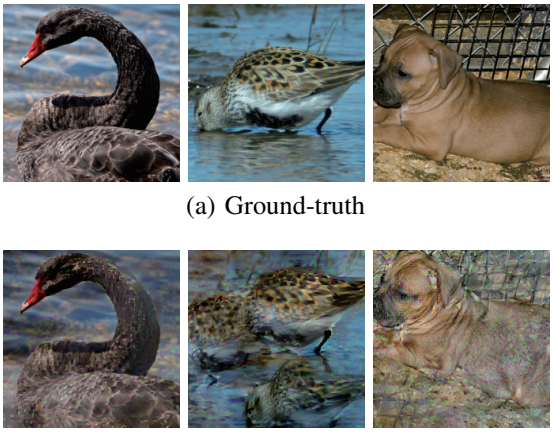
We then compare **LRB** label accuracy under various γ values in Table V. The $\gamma = 0.4$ demonstrates superior performance across different batch sizes in terms of label recovery accuracy. Consequently, we adopted $\gamma = 0.4$ in the final scheme design.

D. AFGI Reconstruct Results

In this subsection, we present the reconstructed results under different batch sizes. When the batch size is 1, we examine the ablation experiments to test the effect of different regularization terms. Next, we compare the time costs of different GIA strategies to provide a comparison of different initialization images and loss functions. Finally, we apply **AFGI** to obtain $\hat{x}$ when the batch size is greater than 1.

1) Reconstructed Results as Batch Size = 1: As DLG [7] is a pioneering work in iteration-based GIAs, GGI [9] utilizes cosine similarity as the loss function, which is the same as **AFGI**. Additionally, ROG [15] achieves state-of-the-art GIA results on the ImageNet dataset. Consequently, we perform a comparative analysis of visual and numerical outcomes among DLG, GGI, ROG, and **AFGI**. To address the mismatch between $\hat{y}$ and x in large batch sizes, we sort $\hat{y}$ in an ascending order, following settings the same as those in GradInversion [10]. The MI for each strategy is set according to their respective previous studies. Visual representations of all $\hat{x}$ are presented in Fig. 3. To quantitatively assess performance, we employ three evaluation metrics: PSNR, SSIM, and LPIPS, which are summarized in Table VI.

From Fig. 3, the findings consistently demonstrate that **AFGI** outperforms other strategies in terms of SSIM and LPIPS. Image objects can be accurately identified in Fig.s 3 (b) and (g). In contrast, DLG [7] (as shown in Fig. 3 (c) and (h)) and ROG (as shown in Fig. 3 (e) and (j)) fail to detect objects, while the color in GGI (as shown in Fig. 3 (d) and (i)) is distorted. Moreover, **AFGI** is of a faster model convergence in reconstructing useful image information (such as image objects) compared to all other strategies. Despite consuming less time in ResNet-50, ROG [15] results are worse in identifying image subjects. These outcomes align with previous findings from GGI and GradInversion [9], [10], which suggest that $\hat{x}$ in shallow models is better than that in deep models.



(a) Ground-truth

(b) **AFGI**

Fig. 4. The results of $\hat{x}$ with a batch size of 1. We reconstruct the 5000-th, 7000-th, and 9000-th images of the ImageNet validation set in **AFGI**.

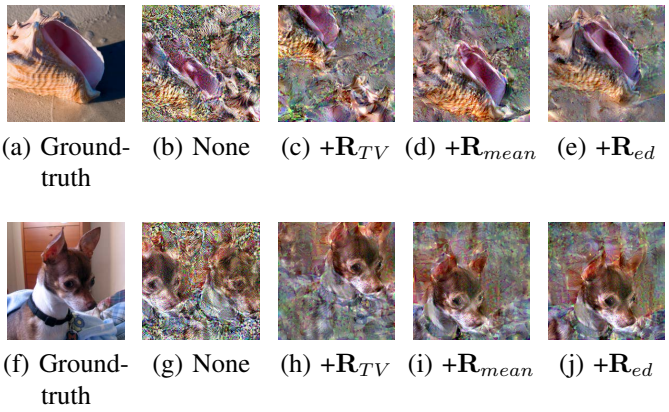
In Table VI, it is evident that ROG fails to detect any objects or useful information in the images (as shown in Fig.s 3 (e) and (j)), though its PSNR values remain higher than those of other methods. This result indicates that smoothing images is helpful for improving PSNR values. This also suggests that solely relying on traditional evaluation metrics, such as PSNR, does not precisely gauge the true quality of images. **AFGI** achieves better SSIM and LPIPS values in all $\hat{x}$. We conjecture that the use of the encoder and decoder to smooth $\hat{x}$ makes the image edges to blurred and immersed with the image background, resulting in low SSIM and LPIPS values.

Finally, we present additional reconstructed results for the indexes of 5000, 7000, and 9000 images in Fig. 4 to illustrate the universality of the **AFGI** strategy.

Ablation experiment: To elucidate the effect of each regu-

TABLE VII
THE METRIC VALUES OF FIG. 5

$\mathbf{R}_{reg}$	Metrics		
	PSNR ↑	**SSIM ↑**	**LPIPS ↓**
None	14.51	0.0092	0.7576
	14.94	0.0091	0.7039
$+\mathbf{R}_{TV}$	16.08	0.0591	0.6068
	16.32	0.0820	0.5899
$+\mathbf{R}_{mean}$	16.30	**0.0514**	0.5640
	16.07	0.1065	0.6116
$+\mathbf{R}_{ed}$	**17.47**	0.0572	**0.4909**
	15.87	**0.1183**	**0.5762**



(a) Ground-truth (b) None (c) +$\mathbf{R}_{TV}$ (d) +$\mathbf{R}_{mean}$ (e) +$\mathbf{R}_{ed}$

(f) Ground-truth (g) None (h) +$\mathbf{R}_{TV}$ (i) +$\mathbf{R}_{mean}$ (j) +$\mathbf{R}_{ed}$

Fig. 5. Ablation study of each regularization term at a batch size of 1. Ground-truth images are shown in figure (a) and (f), while the others are reconstructed using different regularization terms.

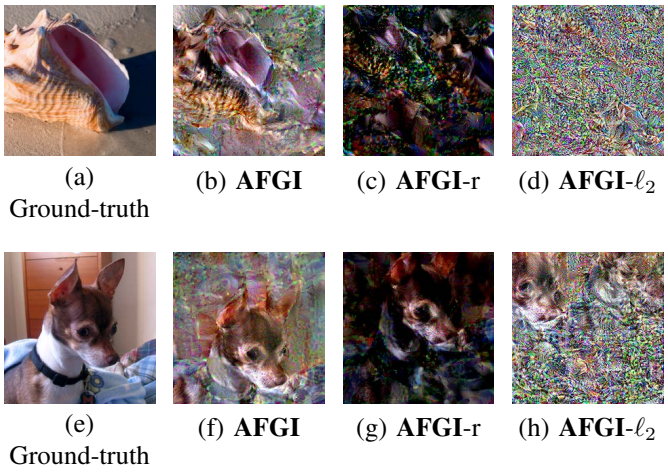
larization term on $\hat{x}$, we sequentially incorporate regularization term one by one into the reconstruction process. A visual and quantitative comparison of $\hat{x}$ is summarized in Fig. 5 and Table VII. If there is no regularization term in GIAs, it is denoted as None in Fig. 5 (b). The $\hat{x}$ results exhibit significant pixel noises. The regularization term $\mathbf{R}_{TV}$ enhances image quality and prohibits noises. However, $\mathbf{R}_{TV}$ introduces a noticeable shift of the subject's position and inaccurate colors in $\hat{x}$, as shown in Fig. 5 (c). In Fig. 5 (d), we introduce $\mathbf{R}_{mean}$ to correct color in $\hat{x}$. This operation can increase the PSNR values of $\hat{x}$. Finally, to rectify the position of the image subject, the objective function incorporates $\mathbf{R}_{ed}$. The results show higher SSIM values and lower LPIPS values in most cases in Table VII, sheding light on the overall improvement of recovered $\hat{x}$.

Impact of image initialization: To examine the impact of $\hat{x}$ initialization on final $\hat{x}$, visual results and evaluation metrics are presented in Fig. 6 (b) (e) and Table VIII. A notable finding is the superior reconstruction achieved when using gray images as the initial value of $\hat{x}$. Comparisons between random initialization of $\hat{x}$ (**AFGI**-r) and gray image initialization of $\hat{x}$ (**AFGI**) highlight the superior reconstruction achieved when using gray images as the initial $\hat{x}$. This underscores the vital role of initial $\hat{x}$ in enhancing the quality of final reconstruction.

TABLE VIII
THE METRIC VALUES FOR FIG. 6

$\hat{x}$	Metrics		
	PSNR ↑	SSIM ↑	LPIPS ↓
AFGI	**17.47**	**0.0572**	**0.4909**
	15.87	**0.1183**	**0.5762**
AFGI-r	12.54	0.0171	0.6656
	10.33	0.0393	0.5922
AFGI-ℓ_2	14.00	0.0089	1.1852
	14.29	0.0081	0.9543

Impact of loss functions: We conduct fine-tuning experiments, as shown in Fig. 6, to examine the impact of different cost functions. Our experimental results indicate that the cosine similarity cost function achieves the highest



(a) Ground-truth (b) **AFGI** (c) **AFGI**-r (d) **AFGI**-ℓ_2

(e) Ground-truth (f) **AFGI** (g) **AFGI**-r (h) **AFGI**-ℓ_2

Fig. 6. The $\hat{x}$ of **AFGI** based on different initializations of x and different loss functions.

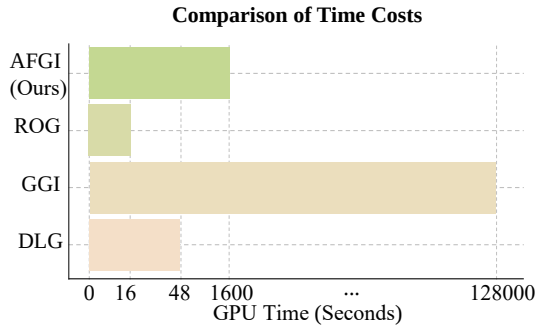


Fig. 7. The time costs of different strategies at the batch size =1.

quality of $\hat{x}$, providing a notable advantage. We consider **AFGI**-ℓ_2 by modifying the cost function to cosine similarity and ℓ_2 distance. Visual results of reconstructed images and corresponding metrics are provided in Fig. 6 (c) and (f), and the last row of Table VIII. Notably, cosine similarity exhibits a stronger tendency to generate high-quality $\hat{x}$. All strategies based on the cosine similarity cost function show higher PSNR values and lower LPIPS values than those using the ℓ_2 distance cost function. This observation deviates from the findings in GradInversion [10]. The discrepancy is attributed to different regularization terms. In our experiments, the choice of the cost function can significantly influence the quality of recovered $\hat{x}$, with cosine similarity showing a substantial advantage.

Time cost comparison: Finally, we compare the time costs of DLG [7], GGI [9], ROG [15], and **AFGI** when the batch size is 1. The results are shown in Fig. 7. Notably, **AFGI** consistently outperforms GGI in terms of reconstruction quality, even with a reduction of over 85% in total time costs. Although ROG and DLG have lower costs, they perform poorly and fail to extract useful information from $\hat{x}$. These two methods are considered ineffective in reconstructing high-resolution images in deep models. **AFGI** strikes a balance between time costs and better GIA final results for $\hat{x}$.

2) Reconstructed Results as Batch Size $>$ 1: Fig. 8 displays the $\hat{x}$ results of **AFGI** with a training batch size of 2. It is easy

(a) Ground-truth

(b) **AFGI**

Fig. 8. The $\hat{x}$ of **AFGI** under a batch size is 2.

(a) Ground-truth (b) batch size = 1 (c) batch size = 2 (d) batch size = 8 (e) batch size = 48

Fig. 9. The $\hat{x}$ results of **AFGI** with batch sizes ranging from 1 to 48.

to observe that the color of $\hat{x}$ in the third column of Fig. 8 (b) is lighter than the corresponding ground-truth data, while the color of the image in the fourth column is darker than its ground-truth counterpart. It is important to note that $\hat{x}$ is influenced by factors such as the color and image subjects of other images in the same training batch.

Next, we evaluate the attack performance of **AFGI** with the batch size varying from 1 to 48. The results are presented in Fig. 9. **AFGI** maintains its ability to visually identify the image subject in $\hat{x}$ even with a large training batch size. However, the color of $\hat{x}$ in a large training batch may shift due to the influence of other images in the same batch. From these results, it is evident that the quality of $\hat{x}$ deteriorates as the batch size increases. This finding also implies that increasing the batch size can better preserve data privacy.

V. Conclusion

We introduce **AFGI**, an effective iteration-based gradient inversion strategy that combines two novel approaches: **LRB** to enhance label accuracy and **VME** to improve the quality of reconstructed images. **LRB** utilizes a convolutional network to accurately reconstruct labels without the strong assumption of no-repeating labels in a training batch. Experimental results highlight the efficiency of **AFGI** in significantly reducing time costs and consistently achieving superior reconstructed results. These findings manifest the practicality of gradient inversion in real-world applications and underscore the necessity to safeguard gradients.

In our future work, we will extend our algorithm into the domain of text reconstruction. Besides, we will also explore defense strategies to counteract these gradient inversion attacks so as to boost user privacy preservation.

References

[1] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[2] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," *arXiv preprint arXiv:1905.10497*, 2019.

[3] J. Chen, X. Pan, R. Monga, S. Bengio, and R. Jozefowicz, "Revisiting distributed synchronous sgd," *arXiv preprint arXiv:1604.00981*, 2016.

[4] Z. Zhou, C. Xu, M. Wang, X. Kuang, Y. Zhuang, and S. Yu, "A multi-shuffler framework to establish mutual confidence for secure federated learning," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 5, pp. 4230–4244, 2023.

[5] Z. Tian, R. Zhang, X. Hou, L. Lyu, T. Zhang, J. Liu, and K. Ren, "federboost: Private federated learning for gbdt," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 3, pp. 1274–1285, 2024.

[6] X. Lin, J. Wu, J. Li, C. Sang, S. Hu, and M. J. Deen, "Heterogeneous differential-private federated learning: Trading privacy for utility truthfully," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 6, pp. 5113–5129, 2023.

[7] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019.

[8] B. Zhao, K. R. Mopuri, and H. Bilen, "idlg: Improved deep leakage from gradients," *arXiv preprint arXiv:2001.02610*, 2020.

[9] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients - how easy is it to break privacy in federated learning?" in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 16 937–16 947.

[10] H. Yin, A. Mallya, A. Vahdat, J. M. Alvarez, J. Kautz, and P. Molchanov, "See through gradients: Image batch recovery via gradinversion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 16 337–16 346.

[11] X. Dong, H. Yin, J. M. Alvarez, J. Kautz, and P. Molchanov, "Deep neural networks are surprisingly reversible: a baseline for zero-shot inversion," *arXiv preprint arXiv:2107.06304*, 2021.

[12] A. Hatamizadeh, H. Yin, P. Molchanov, A. Myronenko, W. Li, P. Dogra, A. Feng, M. G. Flores, J. Kautz, D. Xu, and H. R. Roth, "Do gradient inversion attacks make federated learning unsafe?" *IEEE Transactions on Medical Imaging*, vol. 42, no. 7, pp. 2044–2056, 2023.

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

[14] C. Chen and N. D. Campbell, "Understanding training-data leakage from gradients in neural networks for image classification," *arXiv preprint arXiv:2111.10178*, 2021.

[15] K. Yue, R. Jin, C.-W. Wong, D. Baron, and H. Dai, "Gradient obfuscation gives a false sense of security in federated learning," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 6381–6398.

[16] C. Liu, J. Wang, and D. Yu, "Raf-gi: Towards robust, accurate and fast-convergent gradient inversion attack in federated learning," *arXiv preprint arXiv:2403.08383*, 2024.

[17] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[18] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[19] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[21] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[22] W. Wei, L. Liu, M. Loper, K.-H. Chow, M. E. Gursoy, S. Truex, and Y. Wu, "A framework for evaluating gradient leakage attacks in federated learning," *arXiv preprint arXiv:2004.10397*, 2020.

[23] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, 2019.

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

[25] D. C. Nguyen, M. Ding, Q.-V. Pham, P. N. Pathirana, L. B. Le, A. Seneviratne, J. Li, D. Niyato, and H. V. Poor, "Federated learning meets blockchain in edge computing: Opportunities and challenges," *IEEE Internet of Things Journal*, vol. 8, no. 16, pp. 12 806–12 825, 2021.
[26] S. Yang, B. Ren, X. Zhou, and L. Liu, "Parallel distributed logistic regression for vertical federated learning without third-party coordinator," *arXiv preprint arXiv:1911.09824*, 2019.
[27] T. Chen, X. Jin, Y. Sun, and W. Yin, "Vafl: a method of vertical asynchronous federated learning," *arXiv preprint arXiv:2007.06081*, 2020.
[28] Y. Chen, W. Lu, J. Wang, and X. Qin, "Fedhealth 2: Weighted federated transfer learning via batch normalization for personalized healthcare," *arXiv preprint arXiv:2106.01009*, 2021.
[29] C. He, M. Annavaram, and S. Avestimehr, "Group knowledge transfer: Federated learning of large cnns at the edge," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 14 068–14 080.
[30] J. Xu, C. Hong, J. Huang, L. Y. Chen, and J. Decouchant, "Agic: Approximate gradient inversion attack on federated learning," in *2022 41st International Symposium on Reliable Distributed Systems (SRDS)*, 2022, pp. 12–22.
[31] T. Chen, M. Xu, X. Hui, H. Wu, and L. Lin, "Learning semantic-specific graph representation for multi-label image recognition," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 522–531.
[32] S. Yun, S. J. Oh, B. Heo, D. Han, J. Choe, and S. Chun, "Re-labeling imagenet: from single to multi-labels, from global to localized labels," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2340–2350.
[33] B.-B. Gao and H.-Y. Zhou, "Learning to discover multi-class attentional regions for multi-label image recognition," *IEEE Transactions on Image Processing*, vol. 30, pp. 5920–5932, 2021.
[34] Y. Li and B. Liu, "Improved edge detection algorithm for canny operator," in *2022 IEEE 10th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*, vol. 10, 2022, pp. 1–5.
[35] G. Apruzzese and V. S. Subrahmanian, "Mitigating adversarial gray-box attacks against phishing detectors," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 5, pp. 3753–3769, 2023.
[36] Y. Wei, Z. Ma, Z. Ma, Z. Qin, Y. Liu, B. Xiao, X. Bi, and J. Ma, "Effectively improving data diversity of substitute training for data-free black-box attack," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 4, pp. 4206–4219, 2024.
[37] L. Bu, Z. Zhao, Y. Duan, and F. Song, "Taking care of the discretization problem: A comprehensive study of the discretization problem and a black-box adversarial attack in discrete integer domain," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 5, pp. 3200–3217, 2022.
[38] H. Sun, T. Zhu, J. Li, S. Ji, and W. Zhou, "Attribute-based membership inference attacks and defenses on gans," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 4, pp. 2376–2393, 2024.
[39] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.
[40] S.-C. Huang, A. Pareek, M. Jensen, M. P. Lungren, S. Yeung, and A. S. Chaudhari, "Self-supervised learning for medical image classification: a systematic review and implementation guidelines," *NPJ Digital Medicine*, vol. 6, no. 1, p. 74, 2023.
[41] Y. Yang, A. Panagopoulou, S. Zhou, D. Jin, C. Callison-Burch, and M. Yatskar, "Language in a bottle: Language model guided concept bottlenecks for interpretable image classification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 19 187–19 197.
[42] C. Liu, H. Yang, J. Fu, and X. Qian, "4d lut: learnable context-aware 4d lookup table for image enhancement," *IEEE Transactions on Image Processing*, vol. 32, pp. 4742–4756, 2023.
[43] Y. Zhang, D. Zhou, B. Hooi, K. Wang, and J. Feng, "Expanding small-scale datasets with guided imagination," in *Advances in Neural Information Processing Systems*, vol. 36, 2023, pp. 76 558–76 618.