Amplified Gradient Inversion Attacks on Federated Learning Frameworks

DOI: 10.54654/isj.v3i23.1066

Tran Anh Tu*, Dinh Cong Thanh, Tran Duc Su

***Abstract*— Federated Learning (FL) facilitates collaborative model training while safeguarding data privacy, making it ideal for sensitive fields such as finance, education, and healthcare. Despite its promise, FL remains vulnerable to privacy breaches, particularly through gradient inversion attacks that can reconstruct private data from shared model updates. This research introduces a nonlinear amplification strategy that enhances the potency of such attacks, revealing heightened risks of data leakage in FL environments. Additionally, we evaluate the resilience of privacy-preserving mechanisms, such as Differential Privacy (DP) and Homomorphic Encryption (HE), by employing two proposed metrics AvgSSIM and AvgMSE to measure both the severity of attacks and the efficacy of defenses.**

***Tóm tắt*— Học liên kết (Federated Learning - FL) cho phép huấn luyện mô hình chung mà vẫn bảo vệ quyền riêng tư của dữ liệu, đặc biệt phù hợp trong các lĩnh vực nhạy cảm như tài chính, giáo dục và y tế. Tuy nhiên, mặc dù có nhiều tiềm năng, FL vẫn đối mặt với nguy cơ bị xâm phạm quyền riêng tư, đặc biệt thông qua các tấn công dịch ngược gradient, cho phép khôi phục dữ liệu riêng tư từ các cập nhật mô hình được chia sẻ. Nghiên cứu này giới thiệu chiến lược khuếch đại phi tuyến, nhằm tăng cường hiệu quả của các tấn công này, từ đó làm nổi bật rủi ro rò rỉ dữ liệu trong môi trường FL. Bên cạnh đó, nhóm tác giả đã đánh giá khả năng chống chịu của một số cơ chế bảo vệ quyền riêng tư, như Bảo mật vi sai (Differential Privacy - DP) và Mã hóa đồng cấu (Homomorphic Encryption - HE), thông qua hai chỉ số được đề xuất là AvgSSIM và AvgMSE, để đo lường mức độ nghiêm trọng của các cuộc tấn công cũng như hiệu quả của các biện pháp phòng thủ.**

This manuscript was received on October 28, 2024. It was reviewed on December 5, 2024, revised on December 11, 2024 and accepted on December 18, 2024.

* Corresponding author

***Keywords*— Federated learning, inversion attacks, differential privacy, homomorphic encryption.**

***Từ khóa*— Học liên kết, tấn công dịch ngược, bảo mật vi sai, mã hóa đồng cấu.**

I. INTRODUCTION

Deep learning, a subset of machine learning, leverages multi-layered artificial neural networks to automatically discover intricate patterns within large datasets. This approach has demonstrated impressive capabilities across various applications, from computer vision to natural language processing, due to its capacity for learning hierarchical representations. However, training deep learning models often requires access to massive datasets, which are typically centralized to facilitate model updates and performance improvements. This centralized approach raises significant privacy concerns, especially when sensitive or personal data is involved. In response to these concerns, FL has emerged as a widely-used approach for distributed model training on sensitive data without the need for centralized storage [12]. In this framework, the model is hosted on a server, but the server lacks direct access to clients' data. Instead, clients process the model on their local data and return gradient updates to the server. This method is intended to enhance data privacy, as clients only share gradient information rather than raw data.

However, studies such as [9, 19, 20] have demonstrated that servers can still reconstruct training data from gradient updates, undermining the privacy assurances of FL. Recently, several studies have begun applying gradient inversion attack to FL [10, 20]. Salem et al. [13] explored whether the set of updated data could be inferred from changes in a model's output, proposing

four different attacks aimed at reconstructing or inferring information from this set. Hitaj et al. [10] demonstrated that FL is vulnerable to Membership Inference Attacks (MIA), revealing that an adversary involved in the FL process can reconstruct a victim's training data without violating FL protocols. Zhu et al. [20] further revealed that it is possible to reconstruct training data by merely utilizing the gradients shared by FL participants. Although some of these attacks can successfully reconstruct sensitive training samples in specific scenarios, it is important to highlight that these scenarios often involve more relaxed conditions, as the target models are still in the training phase. In contrast, our work addresses a more challenging and general setting, where the target model remains fixed after a certain number of global epochs. This renders the methods employed in [13, 20] ineffective in such a scenario.

These gradient inversion attacks work by optimizing the input space to identify samples whose gradients match the observed gradients, remaining effective even in the presence of secure aggregation techniques [5], which aim to conceal individual gradients [11, 16]. To counter these gradient inversion attacks, previous research has proposed defenses based on differential privacy [1] as well as heuristic methods such as gradient pruning [2] and sign compression [4]. These techniques have been effective, significantly reducing the success of optimization-based attacks [11, 20], while preserving model performance. Consequently, these defenses have helped mitigate the risks posed by gradient inversion attacks in real-world FL implementations.

Relying solely on existing attack evaluations can lead to a false sense of security in FL. To address this gap, we present a novel and potent threat: amplification-based attacks, which we term Amplified Gradient Inversion Attacks (AGIA). In this approach, adversaries exploit shared parameters between participants in FL to generate predictions on a target dataset. By manipulating the model outputs, the attacker amplifies subtle variations in gradients, enabling the extraction of sensitive data with enhanced accuracy. This method can even bypass traditional defenses, such as differential privacy. We assume that the adversary (e.g., the central server) has access to an auxiliary dataset with a similar distribution to the private data. The gradient inversion model is then trained on samples from this auxiliary dataset, with gradients provided by the global model. Our attack method is highly adaptable and proves resilient against a wide range of defense mechanisms, as the introduction of any defense effectively acts as data augmentation for the gradient inversion model, further enhancing its robustness.

Our experiments across multiple datasets demonstrate the effectiveness of this approach, underscoring the urgent need for stronger defenses to protect against privacy breaches in FL. Given the strong empirical performance of this approach and its adaptability to different learning tasks and defense methods, we recommend its use as a simple baseline for future research on gradient inversion attacks in FL.

This paper is organized as follows. After this introduction, we delve into the technical details of the AGIA in the next section. We then evaluate the effectiveness of AGIA under various scenarios, including an assessment of defense mechanisms such as DP and HE. Finally, we conclude the report by summarizing key findings and outlining directions for future research aimed at enhancing the security and privacy of FL systems.

II. Amplified Gradient Inversion Attacks

A. Threat Model

In our threat model, we assume that n participants train a model using federated learning. After completing local epochs, each participant sends their locally trained models to a semi-trusted aggregation server. The adversary is an honest-but-curious server that follows the protocol but seeks to infer private data from observed gradients. At each global epoch t, the adversary accesses both the global model parameters, W^t, and the participants' local parameters, W_i^t.

Additionally, we assume the adversary has an auxiliary dataset, D_{aux}, which could follow the same or a mixed distribution as the private data. This setup is consistent with previous work [11] and common in privacy attack studies like membership inference [7], making our threat model realistic and aligned with current research.

In this threat model, the adversary aims to build a Global Attack Network A_θ^t to target the global model W_θ^t, along with multiple Local Attack Networks A_i^t to target the local models W_i^t, with the ultimate objective of reconstructing private training data. This is achieved by leveraging the prediction vectors produced by the target models, applying the direct inversion technique. For the sake of notation, we define the model being attacked as the Target Model T_θ, and the model executing the attack as the Attack Model A_θ.

In this scenario, the Target Network T_θ is trained using a private dataset $D_{train} = \{x_{train}, y_{train}\}$ and evaluated on a separate private test set $D_{test} = \{x_{test}, y_{test}\}$ for a classification task. The objective of the classification task is to find a function T_θ that maps each sample to its correct class by minimizing the cross-entropy loss:

$$\widehat{T_\theta} = argmin_{T_\theta} Cross\ Entropy(T_\theta(x_{train}), y_{train}) \quad (1)$$

To carry out the attack, the adversary creates an Attack Network A_θ, which is trained on auxiliary dataset, $D_{aux} = \{x_{attack}, y_{attack}\}$ that follows the same distribution as the private training data D_{train}. In this attack scenario, the adversary has access to the prediction vectors of the private images and can query the Target Network using the attack dataset. The adversary trains the Attack Network by minimizing the following objective:

$$\widehat{A_\theta} = argmin_{A_\theta} MSE\left(A_\theta\left(T_\theta(x_{attack})\right), x_{attack}\right) \quad (2)$$

In this equation, MSE refers to the mean-square error. After training, the objective is for the Attack Network to be capable of reconstructing the private training samples from the Target Network's outputs:

$$A_\theta(T_\theta(x_{train})) \approx x_{train}$$

B. Hidden Patterns in Model Predictions

To examine the information contained within prediction vectors, we utilize a method similar to the approach proposed by Yang et al. [15] and Zhang et al. [18], known as the direct inversion technique. In this method, an adversary first trains an Attack Network using an auxiliary dataset. The goal is to reconstruct the private training data by directly analyzing the prediction vectors outputted by the Target Network. As depicted in Figure 1, the original data encompasses various attributes, such as shape, texture, color, and other unique details. The Target Network extracts features from the input data and produces a prediction vector based on the learned knowledge. However, during this process, certain features not recognized by the Target Network are lost. Typically, in a well-trained network, the prediction vector consists of a dominant entry and several smaller entries, with the largest entry dictating the classification result. Furthermore, prediction vectors belonging to the same category still exhibit slight variations (discussed below). As a result, we can divide the information within the prediction vector into two components: category information and hidden information. The category information (shown in the blue) represents the classification result, contained within the largest entry. The hidden information (red) reflects the subtle differences between prediction vectors of the same class, distributed across all entries. Lastly, the lost information (dashed cube) represents details lost during the classification process.

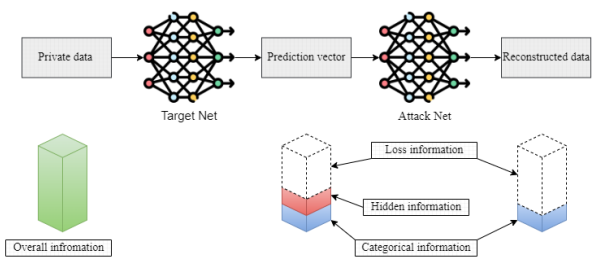

Figure 1. Visualization of Information Loss in Direct Inversion

For the Attack Network, learning the category information is straightforward, but extracting hidden information is far more challenging. This is because the prediction vector from a well-trained model predominantly

consists of one large entry, which is nearly 1, and several small entries close to 0. Additionally, the differences between prediction vectors within the same class are minimal, making it difficult for the Attack Network to distinguish between them and reconstruct unique samples. A statistical analysis of the prediction vector entries that do not influence the classification result reveals that 98% of these entries are smaller than 10^{-2} [18]. When both the large and small entries are inputted into the Attack Network, it struggles to detect these subtle values, making it difficult to extract hidden information. As a result, the network primarily learns the category information, along with only a small amount of hidden information, as depicted in Figure 1.

This phenomenon stems from how machine learning algorithms work, especially in classification tasks. These models extract common features among samples in the same class while discarding unique characteristics during training. As a result, the differences between prediction vectors within the same class are minimized. To avoid overfitting, the largest entry in a prediction vector is close to 1, and smaller entries are slightly above 0, leading to minor variations between samples. While this improves the model's generalization, it complicates model inversion because the unique details needed to distinguish samples are reduced or lost.

Given the observations above, it is clear that amplifying the information extracted from prediction outputs can significantly enhance the adversary's ability to recover hidden and sensitive details from the input data. Current attack methodologies [6, 8, 17] exhibit a limited exploitation of the rich information present within prediction vectors. These approaches primarily focus on category information, thereby neglecting the substantial hidden information embedded in the output. Yang et al. [15] addressed this challenge by employing a logarithmic function to rescale prediction vectors, while Zhang [18] introduced the Nonlinear Amplification Function to further enhance the retrieval of information from prediction outputs. However, these methods are constrained to centralized training models and fail to consider the complexities of federated learning systems. These studies typically assume a secure model within a black-box setting, where the attacker can only observe the model's output predictions. However, in the context of FL, the attacker operates within a white-box environment. This means that the attacker has access to the shared model parameters during the training process, providing a more comprehensive view of the model's internal state. The attack diagram is shown in Figure 2. In this scenario, the attacker could be a semi-honest aggregation server, a participating client, or an external adversary capable of intercepting and accessing the exchanged parameters between participants. The attacker can use these parameters to generate output predictions, which can then be leveraged to perform input data reconstruction attacks.

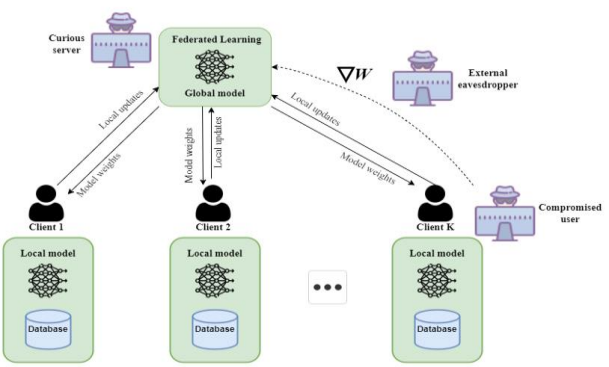

Figure 2. Federated Learning attacking simulation

In this paper, we extend this line of research by exploring the impact of a nonlinear amplification layer in conjunction with the gradient-sharing mechanisms inherent in federated learning. Our approach aims to amplify the subtle, often overlooked information present in the gradients exchanged during training, thereby increasing the adversary's ability to reconstruct sensitive data from participating clients. Using the obtained shared parameters, the attacker generates prediction outputs. However, these prediction outputs alone are not sufficiently effective in reconstructing the original data. Therefore, we apply an output information amplification technique to enhance data reconstruction capabilities. We rigorously

evaluate the effectiveness of this technique and demonstrate how it exposes federated learning systems to significant risks of privacy breaches. A detailed analysis of our attack methodology, including its implementation and impact, is presented in subsequent sections.

C. Amplifying Gradient Inversion Attacks to Reconstruct Distinctive Local Private Samples in Federated Learning

Reconstructing the fine-grained details of private samples requires more than just surface-level information. In this work, we aim to achieve such reconstruction by harnessing the hidden signals embedded within prediction vectors. Building on the methods proposed by Yang et al. [15] and Zhang et al. [18], we introduce a nonlinear amplification layer to amplify the subtle information captured by the Target Model's prediction layer before feeding it into the Attack Model.

The nonlinear amplification layer includes a nonlinear amplification function f. The function f is called a nonlinear amplification function if it satisfies all the following conditions:

- $x \leq f(x)$
- $f(0) = 0,\ f(1) = 1$
- If $x < y,$ then $f(x) < f(y)$, $f(x)$ is monotonically increasing
- The graph of $f(x)$ is concave down.

These four conditions play distinct roles in constraining the linear amplification function. The first condition, $x \leq f(x)$, ensures that the function amplifies all elements within the prediction vector, meaning that smaller values are boosted. The second condition requires retaining the values if they are 0 or 1, hence restricting the range of the nonlinear amplification function to $[0, 1]$. The third condition ensures that the relative relationships between elements in a vector remain unchanged. That is, if the value of an element x in the vector is smaller than the value of an element y in the same vector, then after amplification it must still be smaller than y after y is also amplified. This is critical for a successful inversion attack, as once the attack network learns this relationship, it can accurately reconstruct unseen private samples using their prediction vectors. Moreover, the requirement for monotonicity ensures that classes are not mixed after amplification, resulting in accurate reconstructions. The final condition ensures that the nonlinear amplification function will sufficiently boost smaller prediction elements in the vector to a large enough value, while only lightly amplifying already large prediction elements. If the fourth condition were not met, one could easily find a function that satisfies the first three conditions but only lightly amplifies smaller prediction elements, thus rendering the amplification function ineffective in this problem. As outlined earlier in this report, the function $y = x^{\alpha}$ with $0 < \alpha < 1$ satisfies all four conditions to qualify as a nonlinear amplification function.

The amplification is controlled by a nonlinear function x^{α} where $0 < \alpha < 1$, designed to boost small values in the prediction vectors, while simultaneously increasing the differences between vectors belonging to the same class. This enhancement allows the Attack Network to uncover and leverage previously obscured details within the prediction data.

During training, the Attack Network learns the distribution of these amplified prediction vectors and correlates them with the auxiliary dataset, establishing a robust mapping between the prediction vector space and the original data space. After training, the Attack Network is able to transform prediction vectors into meaningful reconstructions of private data samples. We refer to this technique as Amplifying Gradient Inversion Attacks (AGIA), with its architectural design illustrated in Figure 3. In this model, nonlinear amplification layer, which incorporates a nonlinear amplification function, is placed between the Target Network and the Attack Network. This amplification function enhances small entries in the prediction vectors, allowing for the reconstruction of distinctive samples.

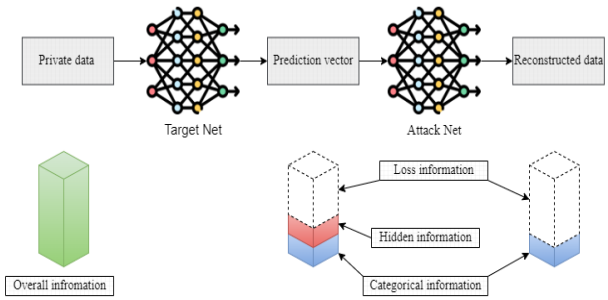

Figure 3. Architecture Overview of the AGIA

When the adversary obtains the local model W_i^t from participant P_i or the global model W^t after aggregation, they will use it to process the attack data x_{attack} through the model to generate prediction outputs $T_\theta(x_{attack})$. The Target Network T_θ processes an attack sample x_{attack} and outputs a prediction vector $\hat{y}_{attack} = T_\theta\ (x_{attack})$. This prediction vector is then passed through the nonlinear amplification function $Amplify()$ within the amplification layer, resulting in an amplified prediction vector $\hat{y}_{attack}^{amp} = Amplify(\hat{y}_{attack})$. The Attack Network A_θ then takes this amplified vector as input and generates a reconstructed sample x_{attack}^{rec}, which is compared to the original attack sample x_{attack} to compute the reconstruction loss. The Mean-Square Error (MSE) is used to quantify this loss. The overall training process of the Attack Network can be formalized as:

$$\widehat{A_\theta} = argmin_{A_\theta} MSE(A_\theta\ (Amplify(T_\theta(x_{attack}))), x_{attack}) \quad (3)$$

Once training is complete, the objective is for the Attack Network to reconstruct distinct private samples in each class. Given a prediction vector $\hat{y}_{train} = T_\theta\ (x_{train})$ of a private training sample, the Attack Network can reconstruct x_{train} by solving:

$$A_\theta(Amplify(\hat{y}_{train})) \approx x_{train} \quad (4)$$

III. Results

A. Experimental Setup

We evaluated the vulnerability of FL to AGIA attacks using the MNIST and Fashion-MNIST datasets, with a primary focus on assessing the effectiveness of the attack itself. While privacy-preserving techniques such as DP and Homomorphic Encryption (HE) were tested, they served as secondary considerations. The evaluation employed metrics including Mean Squared Error (MSE), Structural Similarity Index (SSIM), and Pixel Accuracy, with Pixel Accuracy proving the most reliable due to the use of a 5% luminance threshold for assessing image reconstruction fidelity.

Table 1. Target Net Architecture

Target Net Architecture
(conv1): Conv2d(1, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
(conv2): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
(conv3): Conv2d(256, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
(bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
(bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
(bn3): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
(mp1): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
(mp2): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
(mp3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
(relu1): ReLU()
(relu2): ReLU()
(relu3): ReLU()
(fc1): Linear(in_features=8192, out_features=50, bias=True)
(dropout): Dropout(p=0.5, inplace=False)
(fc2): Linear(in_features=50, out_features=10, bias=True)

The TargetNet, as shown in Table 1, consists of three convolutional layers, starting with a Conv2d layer that takes an input of 1 channel and outputs 128 feature maps using a 3 × 3 kernel with a stride of 1 and padding of 1. This is followed by a second Conv2d layer with 256 feature maps and a third Conv2d layer with 512 feature maps, both using the same kernel, stride, and padding configurations. Each convolutional layer is paired with a BatchNorm2d layer to normalize the feature maps, helping stabilize training. The network also includes three

MaxPool2d layers with a 2×2 kernel and stride of 2 to reduce the spatial dimensions. After each convolutional and pooling operation, ReLU activations are applied to introduce non-linearity. The output from the convolutional stack is flattened and passed through a fully connected layer with 8192 input features and 50 output features, followed by a dropout layer with a dropout probability of 0.5 to prevent overfitting. Finally, another fully connected layer maps the 50 features to 10 output classes.

TABLE 2. ATTACK NET ARCHITECTURE

Attack Net Architecture
(deconv1): ConvTranspose2d(10, 512, kernel_size=(4, 4), stride=(1, 1))
(deconv2): ConvTranspose2d(512, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1,1))
(deconv3): ConvTranspose2d(256, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
(deconv4): ConvTranspose2d(128, 1, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
(bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
(bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
(bn3): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
(relu1): ReLU()
(relu2): ReLU()
(relu3): ReLU()
(sigmod): Sigmoid()

The AttackNet, as shown in Table 2, mirrors the architecture of TargetNet but reverses the process, utilizing transposed convolutional layers to reconstruct images from prediction vectors. Starting from 10 input channels, the network gradually upsamples the data through a series of deconvolutional layers with output channels of 512, 256, 128, and finally 1, which represents a grayscale image output. Each deconvolution layer is followed by a BatchNorm2d layer, enhancing stability during training by normalizing activations and aiding in smoother gradient flow. ReLU activations are applied after each deconvolution to introduce non-linearity, allowing the network to capture complex patterns from the input data. Finally, a Sigmoid activation function in the output layer constrains the pixel values to between 0 and 1, ensuring the reconstructed images remain within a normalized range suitable for comparison with the original data.

The experiments were conducted using PyTorch on an NVIDIA Tesla P40 GPU with 24GB of GDDR5 memory. Both the Target and Attack Networks were configured with the following hyperparameters: a batch size of 32, 100 training epochs, and the Adam optimizer with a learning rate of 0.001, betas set to (0.5, 0.999), and a power factor of the non-linear amplifying function $\alpha = 1/10$. The source code for the experiments is available at: https://github.com/lab-secureai/Model-Inversion-Attacks.git.

Training time (seconds) of the model in the cases described in Table 3 and Table 4.

TABLE 3. COMPARISON OF MODELS BASED ON EPOCH AND TIME

Testcase	**Models**					
	Global	**Local IID**	**Local Non-IID**	**DP (min)**	**DP (max)**	**SMC**
Epoch	**Time**	**Time**	**Time**	**Time**	**Time**	**Time**
1	40.97	40.26	38.37	41.88	42.92	41.04
10	405.60	398.57	379.86	414.61	424.91	406.29
50	2137.72	2443.10	2101.79	2245.50	3664.66	2155.68
100	4241.16	4847.04	4169.88	4495.50	7270.56	4315.68

TABLE 4. COMPARISON OF MODELS BASED ON MSE, EPOCHS, AND TIME

MSE	**Global**		**Local IID**		**Local Non-IID**	
	Epoch	**Time**	**Epoch**	**Time**	**Epoch**	**Time**
0.027	-	-	1	40.26	1	38.37
0.028	1	40.97	-	-	-	-
0.031	-	-	10	398.57	10	379.86
0.032	10	405.60	-	-	-	-
0.033	-	-	-	-	50	2443.10
0.034	-	-	-	-	100	4847.04
0.035	50	2137.72	-	-	-	-
0.036	100	4241.16	-	-	-	-
0.037	-	-	-	-	-	-

MSE	DP (min)		DP (max)		SMC	
	Epoch	Time	Epoch	Time	Epoch	Time
0.027	1	41.88	-	-	1	41.04
0.031	10	414.61	-	-	10	406.29
0.032	-	-	-	-	-	-
0.033	50	2245.50	-	-	50	2155.68
0.034	100	4495.50	-	-	100	4315.68
0.035	-	-	50	3664.66	-	-
0.036	-	-	100	7270.56	-	-

B. Reconstruction Results

We explore the performace of AGIA on two types of models: one trained using FL and the other through centralized training. For the FL model, we utilize 5 participants, with data distributed in an IID (Independent and Identically Distributed) fashion. Each participant performs training for a single local epoch before sending updated parameters to the server for aggregation. Both the FL and centralized models are trained over 50 global epochs. Figures 4 presents the reconstruction results of AGIA on the MNIST and Fashion-MNIST datasets, respectively. The reconstructed images are produced at epoch 100 of the attack network, when it has just reached a fit with the auxiliary dataset.

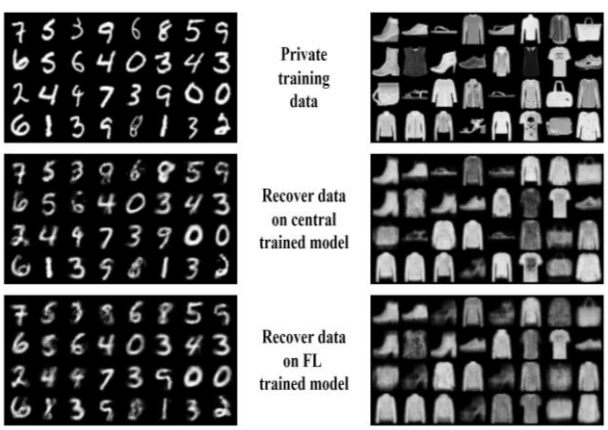

Figure 4. Reconstruction results of AGIA on MNIST and Fashion-MNIST dataset

Our findings reveal that AGIA is able to attack the FL-trained model with results nearly on par with those from attacking the centrally trained model. This similarity is reflected through key metrics such as SSIM, Pixel Loss, and MSE, as illustrated in Table 5.

TABLE 5. EVALUATION METRICS FOR MNIST AND FASHION-MNIST DATASETS

Dataset	SSIM Loss		MSE Loss		Pixel Loss	
	Central	FL	Central	FL	Central	FL
MNIST	0.4576	0.5156	0.0335	0.0373	0.2685	0.2908
Fashion-MNIST	0.4768	0.5470	0.0276	0.0349	0.4521	0.5133

While AGIA's attack on the FL model shows slightly lower accuracy compared to the centralized model, the difference is negligible. This underscores AGIA's potential to inflict substantial data leakage from models trained under the FL paradigm, revealing a critical security flaw within FL. Although FL may mitigate the risk of inversion attacks, it cannot entirely eliminate the threat posed by AGIA.

C. Analysis of Amplification Layer Effects on Attack Performance

The parameter α, with $\alpha = \frac{1}{beta}$, is crucial in adjusting the amplification function's nonlinearity, directly affecting the amplification of prediction vectors. Figures 5 and 6 show the impact of α on AGIA attack metrics (MSE Loss, SSIM Loss, Pixel Loss) for FL-trained models. As α decreases, these metrics initially improve, reaching an optimal value. Beyond this point, further decreases cause performance to decline as excessive amplification blurs the distinction between amplified and original values, diminishing reconstruction accuracy.

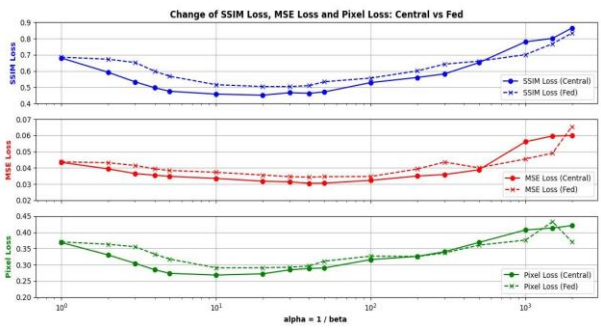

Figure 5. The impact of the amplification layer's alpha parameter on the evaluation metrics

Conversely, a high α results in insufficient amplification, making distinguishing information from incorrect labels too subtle and reducing attack performance. This underscores the importance of balancing α - it must be small enough to enhance attack effectiveness but not so

small as to erase critical data variations. For optimal results, we selected $\alpha = \frac{1}{30}$ in subsequent evaluations.

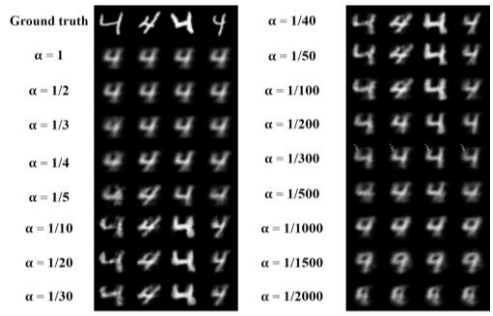

Figure 6. Comparison of inversion outcomes with different α values

D. Impact of Attacks on the Number of FL Rounds

In this experiment, we assessed the vulnerability of individual clients to AGIA. The results of attacking each client (from 1 to 5) at different global epoch rounds are illustrated in Figure 7. Specific evaluation metrics are detailed in Tables 6 and 7.

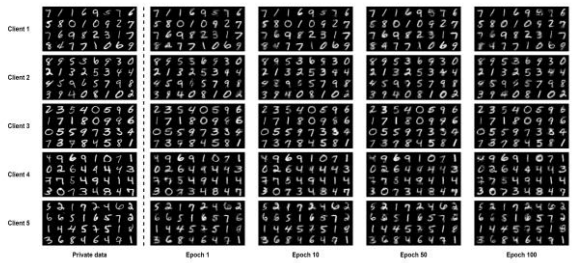

Figure 7. Inversion outcomes from client at different global epoch

Notably, we introduced two new metrics to quantify the effect of inversion attacks on FL models: AvgSSIM, which measures the average SSIM Loss across clients, and AvgMSE, which captures the mean squared error of recovered images from the attack on the model.

TABLE 6. SSIM LOSS FOR DIFFERENT CLIENTS AND EPOCHS

SSIM Loss	Client 1	Client 2	Client 3	Client 4	Client 5	AvgSSIM
Epoch 1	0.4021	0.3930	0.4001	0.4123	0.4133	0.4042
Epoch 10	0.4344	0.4369	0.4402	0.4385	0.4380	0.4376
Epoch 50	0.4710	0.4804	0.4704	0.4791	0.4671	0.4736
Epoch 100	0.4935	0.4925	0.4921	0.4954	0.4793	0.4906

Our findings indicate that as the global epoch increases, the ability to attack a specific client's model diminishes. This reduction occurs because the data, after undergoing several rounds of FL training, becomes more entangled with the aggregated dataset, thereby reducing the likelihood of extracting private information from any individual participant.

TABLE 7. MSE LOSS FOR DIFFERENT CLIENTS AND EPOCHS

MSE Loss	Client 1	Client 2	Client 3	Client 4	Client 5	AvgMSE
Epoch 1	0.0274	0.0273	0.0271	0.0279	0.0284	0.0276
Epoch 10	0.0310	0.0312	0.0315	0.0308	0.0313	0.0312
Epoch 50	0.0342	0.0346	0.0343	0.0339	0.0342	0.0342
Epoch 100	0.0361	0.0353	0.0360	0.0360	0.0351	0.0357

E. Impact of Attacks on Non-IID Data Distributions

Figure 8 illustrates the image reconstruction results for each client in the case of non-IID data distribution. In the case of non-IID data, we divided the training dataset into 20 shards, with each client receiving 4 shards, and these 4 shards correspond to 4 different labels. It is evident that AGIA can successfully reconstruct the images belonging to each client involved in the training process. As shown in Figure 8, even the unique data features present only at specific clients can be recovered during an attack. Despite a significant drop in accuracy, this still underscores the potential risk of data leakage posed by AGIA attacks in real-world scenarios.

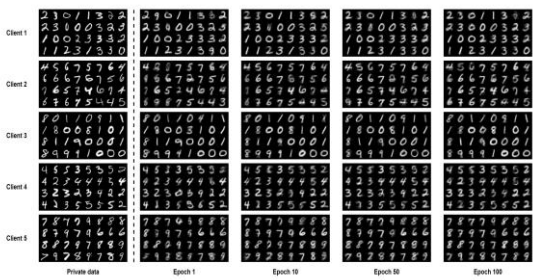

Figure 8. Reconstructed Images from Each Client during the FL Training Process

F. Attacks on Differential Privacy Mechanisms

Tables 8 and 9 present the evaluation metrics, specifically SSIM Loss and average SSIM Loss (AvgSSIM), for images reconstructed from AGIA attacks on various clients under the protection of DP.

TABLE 8. SSIM LOSS FOR DIFFERENT CLIENTS AND EPOCHS WITH SMALL NOISE

	Client 1	Client 2	Client 3	Client 4	Client 5	AvgSSIM
Epoch 1	0.4227	0.4039	0.4301	0.4089	0.4268	0.4185
Epoch 10	0.4724	0.4661	0.4554	0.4772	0.4764	0.4695
Epoch 50	0.4939	0.4959	0.4917	0.4998	0.5114	0.4985
Epoch 100	0.5224	0.5176	0.5059	0.5059	0.5093	0.5122

Table 8 shows results with a lower level of noise ($\sigma = 0.001$) added to the gradients, while Table 9 illustrates outcomes with a higher level of noise ($\sigma = 0.01$). It can be observed that increasing the noise level diminishes the attack's effectiveness in terms of the model's shared information. However, the reconstructed images still exhibit high SSIM and low MSE values, indicating that the quality of reconstructed images remains relatively high despite the presence of noise. This suggests that the current implementation of DP may not provide adequate protection against AGIA attacks, as the added noise does not sufficiently obscure the sensitive data within the model.

TABLE 9. SSIM LOSS FOR DIFFERENT CLIENTS AND EPOCHS WITH LARGE NOISE

	Client 1	Client 2	Client 3	Client 4	Client 5	AvgSSIM
Epoch 1	0.4217	0.4100	0.4108	0.4044	0.4192	0.4132
Epoch 10	0.4366	0.4556	0.4295	0.4275	0.4293	0.4357
Epoch 50	0.5481	0.5159	0.5483	0.5254	0.5271	0.5330
Epoch 100	0.5978	0.6118	0.6033	0.5760	0.5917	0.5961

G. Analysis of Attacks Targeting Cryptography Methods

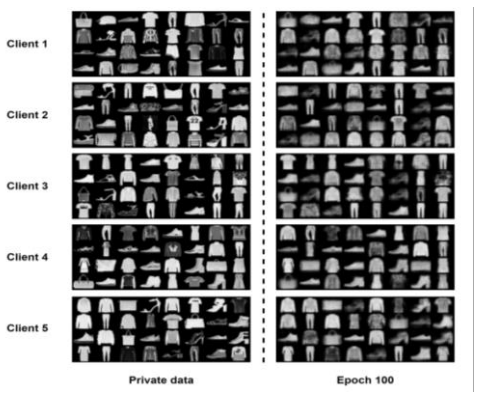

Figure 9. Reconstructed Images of Attacks Targeting Cryptography Methods

The Figure 9 above shows the recovered image using the encryption method. For cryptographic techniques [3], model parameters are encrypted before being shared, ensuring that even the central server cannot access the outputs or parameters during the training process. This provides robust data protection as long as there is no collusion among participating parties. However, techniques like Bonawitz et al. 's secret sharing [5] or secure multi-party computation (SMPC) approaches such as [14] still expose the aggregated model at each training round. Consequently, an attacker could potentially obtain the aggregated model at each epoch. Although this makes the attack more challenging and the obtained results reflect the combined data from multiple parties, it does not directly compromise the data of individual clients.

IV. CONCLUSION

Federated Learning offers a powerful solution for privacy-preserving collaborative model training, but our research highlights the growing concern over its vulnerability to gradient inversion attacks. By employing a nonlinear amplification strategy, we have shown that the risk of data leakage is substantially heightened in federated environments, raising critical questions about the robustness of FL in sensitive domains. Our evaluation of privacy-preserving techniques, including DP and HE, reveals that while these methods can provide some level of defense, they are not entirely effective against enhanced inversion attacks. Overall, this work underscores the importance of continued research into more resilient privacy-preserving techniques to safeguard against increasingly sophisticated threats in FL systems.

ACKNOWNLEDGMENT

The article was completed with support from the Academy of Cryptography Techniques. Anh Tu Tran is partly support by a Japan Advanced Institute of Science and Technology (JAIST) PhD Fellowship in Knowledge Science.

REFERENCES

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang, "Deep learning with differential privacy", *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 308–318, 2016.

[2] Alham Fikri Aji and Kenneth Heafield, "Sparse communication for distributed gradient descent", *arXiv preprint arXiv:1704.05021*, 2017.

[3] Yoshinori Aono, Takuya Hayashi, Lihua Wang, Shiho Moriai, et al, "Privacy-preserving deep learning via additively homomorphic encryption", *IEEE transactions on information forensics and security*, 13(5):1333–1345, 2017.

[4] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar, "Compressed optimisation for non-convex problems", *International Conference on Machine Learning*, pp. 560–569. PMLR, 2018.

[5] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth, "Practical secure aggregation for federated learning on user-held data", *arXiv preprint arXiv:1611.04482*, 2016.

[6] Si Chen, Mostafa Kahla, Ruoxi Jia, and GuoJun Qi, "Knowledge-enriched distributional model inversion attacks", *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 16178–16187, 2021.

[7] Jinhao Duan, Fei Kong, Shiqi Wang, Xiaoshuang Shi, and Kaidi Xu, "Are diffusion models vulnerable to membership inference attacks?", *International Conference on Machine Learning*, pp. 8717–8730. PMLR, 2023.

[8] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures", *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pp. 1322– 1333, 2015.

[9] Jonas Geiping, Hartmut Bauermeister, Hannah Droge, and Michael Moeller, "Inverting gradientshow easy is it to break privacy in federated learning?", *Advances in neural information processing systems*, pp. 33:16937-16947, 2020.

[10] Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz, "Deep models under the gan: information leakage from collaborative deep learning", *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pp. 603-618, 2017.

[11] Jinwoo Jeon, Kangwook Lee, Sewoong Oh, Jungseul Ok, et al, "Gradient inversion with generative image prior", *Advances in neural information processing systems*, pp. 34:29898–29908, 2021.

[12] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas, "Communication-efficient learning of deep networks from decentralized data", *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, 2017.

[13] Ahmed Salem, Apratim Bhattacharya, Michael Backes, Mario Fritz, and Yang Zhang, {UpdatesLeak}: "Data set inference and reconstruction attacks in online learning", *29th USENIX security symposium (USENIX Security 20)*, pp. 1291–1308, 2020.

[14] T. A. Tu, L. T. Dung, and P. X. Sang, "A novel privacy-preserving federated learning model based on secure multi-party computation", *International Symposium on Integrated Uncertainty in Knowledge Modelling and Decision Making*, pp. 321–333. Springer, 2023.

[15] Ziqi Yang, Jiyi Zhang, Ee-Chien Chang, and Zhenkai Liang, "Neural network inversion in adversarial setting via background knowledge alignment", *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pp. 225–240, 2019.

[16] Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M Alvarez, Jan Kautz, and Pavlo Molchanov, "See through gradients: Image batch recovery via gradinversion", *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16337–16346, 2021.

[17] Yuheng Zhang, Ruoxi Jia, Hengzhi Pei, Wenxiao Wang, Bo Li, and Dawn Song, "The secret revealer: Generative model-inversion attacks against deep neural networks", *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 253-261, 2020.

[18] Zeping Zhang, Xiaowen Wang, Jie Huang, and Shuaishuai Zhang, "Analysis and utilization of hidden information in model inversion attacks", *IEEE Transactions on Information Forensics and Security*, vol. 18, pp.4449-4462, 2023.

[19] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen, "idlg: Improved deep leakage from gradients", *arXiv preprint arXiv:2001.02610*, 2020.

[20] Ligeng Zhu, Zhijian Liu, and Song Han, "Deep leakage from gradients", *Advances in neural information processing systems*, no. 1323, pp. 14774 - 14784, 2019.

ABOUT THE AUTHOR

Tran Anh Tu

Workplace: Academy of Cryptography Techniques
Email: tutran@actvn.edu.vn.
Education: Anh-Tu Tran received his B.Eng degree (2013) in Applied Mathematics and Informatics and MSc degree (2016) in Applied Mathematics both from the Hanoi University of Science and Technology, Hanoi, Viet Nam. He is currently a Ph.D. candidate of Computer Science at Institue of Information Technology and Telecommunication, Graduate University of Science and Technology, Vietnam Academy of Science and Technology and Visiting PhD. Student at Japan Advanced Institute of Science and Technology (JAIST).

Recent research direction: His research interests include privacy preserving machine learning, deep learning, cyber security and blockchain technology.

Tên tác giả: **Trần Anh Tú**
Cơ quan công tác: Học viện Kỹ thuật mật mã
Email: tutran@actvn.edu.vn

Quá trình đào tạo: Nhận bằng Cử nhân kỹ thuật (2013) chuyên ngành Toán ứng dụng và Tin học và bằng Thạc sĩ (2016) chuyên ngành Toán ứng dụng tại Đại học Bách khoa Hà Nội, Việt Nam. Hiện đang là ứng viên Tiến sĩ Khoa học Máy tính tại Viện Công nghệ Thông tin và Viễn thông, Viện Sau đại học Khoa học và Công nghệ, Viện Hàn lâm Khoa học và Công nghệ Việt Nam và là Nghiên cứu sinh Tiến sĩ thỉnh giảng tại Viện Khoa học và Công nghệ Tiên tiến Nhật Bản (JAIST).

Hướng nghiên cứu hiện nay: học máy bảo vệ quyền riêng tư, học sâu, an ninh mạng và công nghệ chuỗi khối.

Dinh Cong Thanh

Workplace: Academy of Cryptography Techniques
Email: at170745@actvn.edu.vn
Education: He is currently a final year student at the Academy of Cryptography Techniques, Hanoi, Viet Nam, majoring in Information Security.

Recent research direction: His research interests include privacy preserving machine learning, deep learning.

Tên tác giả: **Đinh Công Thành**
Cơ quan công tác: Học viện Kỹ thuật mật mã
Email: at170745@actvn.edu.vn.
Quá trình đào tạo: Hiện đang là sinh viên năm cuối tại Học viện Kỹ thuật mật mã, Hà Nội, Việt Nam, chuyên ngành An ninh thông tin.
Hướng nghiên cứu hiện nay: học máy bảo vệ quyền riêng tư, học sâu.

Tran Duc Su

Workplace: Posts and Telecommunications Institute of Technology (PTIT)

Email: sutd@ptit.edu.vn

Education: Graduated from a 5-year regular university majoring in information technology from Hanoi University of Science and Technology in 1988, defended and was awarded a doctorate in engineering specializing in electronics - informatics at the Institute of Electronics - Informatics - Automation. in 2004. Received the title of associate professor in 2015.

Recent research direction: Cryptography in information security, block cipher, detecting attacks that ensure privacy on distributed network data sources, data mining, association rule mining that ensures privacy on distributed data doc, network security monitoring.

Tên tác giả: **Trần Đức Sự**

Cơ quan công tác: Học viện Công nghệ Bưu chính Viễn thông

Email: sutd@ptit.edu.vn

Quá trình đào tạo: Tốt nghiệp Đại học chính quy hệ 5 năm chuyên ngành công nghệ thông tin tại trường Đại học Bách khoa Hà Nội năm 1988, bảo vệ và được cấp bằng Tiến sỹ Kỹ thuật chuyên ngành Điện tử - Tin học tại Viện Điện tử - Tin học - Tự động hóa năm 2004. Được phong hàm Phó Giáo sư năm 2015.

Hướng nghiên cứu hiện nay: Mật mã trong an toàn thông tin, mã khối, phát hiện các cuộc tấn công đảm bảo quyền riêng tư trên các nguồn dữ liệu mạng phân tán, khai thác dữ liệu, khai thác luật kết hợp đảm bảo quyền riêng tư trên dữ liệu phân tán, giám sát an ninh mạng.