RESEARCH ARTICLE

Federated Learning Privacy Protection via Training Randomness

Zhiyuan Wang[1], Ming Tian[1], Yutao Qiu[1], Zhixu Zhao[1], Yongpo Zhang[2] and Kun Liu[1,*]

[1] School of Automation, Beijing Institute of Technology, Beijing 100081, China
[2] Aviation University of Air Force, Jilin 130022, China

Abstract

Federated learning is a collaborative machine learning paradigm that trains models across multiple computing nodes while aiming to preserve the privacy of local data held by participants. However, because of the open network environment, federated learning faces severe privacy and security challenges. Studies have shown that attackers can reconstruct original training data by intercepting gradients transmitted across the network, thereby posing a serious threat to user privacy. One representative attack is the Deep Leakage from Gradients (DLG), which iteratively recovers training data by optimizing dummy inputs to match the observed gradients. To address this challenge, this paper proposes a novel privacy-preserving strategy that leverages the randomness inherent in model training. Specifically, during the training process, we introduce the dropout method to selectively disconnect neuronal connections, while adding Gaussian noise to mask the gradients of the disconnected neurons. This approach effectively disrupts gradient-leakage attacks and reduces overfitting, thereby enhancing model generalization. We further provide a theoretical analysis of privacy guarantees using differential privacy metrics. Extensive experiments under federated-learning attack-defense scenarios demonstrate the effectiveness of the proposed strategy. Compared with existing defenses, our method achieves strong privacy protection against gradient leakage while maintaining competitive model accuracy, offering new insights and techniques for federated learning privacy preservation.

Keywords: federated learning, privacy-preserving strategy, gradient leakage attack, dropout method, differential privacy.

1 Introduction

With continuous advances in machine learning, powerful algorithms have been widely applied to fields such as computer vision [1], speech recognition [2], and autonomous driving [3], achieving remarkable results. However, their widespread use in internet-based applications has raised increasing concerns regarding data privacy and security. Issues such as user privacy, business confidentiality, regulatory compliance, and national security often make direct data sharing among organizations infeasible, resulting in the emergence of numerous data silos. [4] Federated learning, a

Submitted: 22 September 2025
Accepted: 13 November 2025
Published: 28 November 2025

Vol. 2, **No.** 4, 2025.
10.62762/TSCC.2025.779613

***Corresponding author:**
✉ Kun Liu
kunliubit@bit.edu.cn

Citation
Wang, Z., Tian, M., Qiu, Y., Zhao, Z., Zhang, Y. & Liu, K. (2025). Federated Learning Privacy Protection via Training Randomness. *ICCK Transactions on Sensing, Communication, and Control*, 2(4), 226–237.

privacy-preserving distributed machine learning paradigm, has emerged as a promising solution for training models collaboratively across decentralized datasets without requiring the direct sharing of sensitive data [5].

Federated learning was initially proposed by Google in 2016 [6] to update models locally on Android devices. It enables collaborative training without sharing raw data by exchanging only gradients between clients and a central server. However, despite this limited data exchange, federated learning still faces privacy risks. For example, adversaries can analyze transmitted updates to infer sensitive attributes or indirectly recover user data through privacy inference techniques [7]. Therefore, developing effective privacy-preserving methods, including encryption, differential privacy, and secure multi-party computation, remains crucial for federated learning.

Considering the limitations in computational overhead and performance degradation associated with traditional privacy-preserving techniques, this paper proposes a novel federated learning approach leveraging randomness during model training to enhance privacy protection. The primary contributions of this paper are as follows:

(1) We propose a privacy-preserving federated learning algorithm based on dropout-induced randomness. By randomly disconnecting neuron connections and injecting Gaussian noise into their corresponding gradients, our method effectively defends against gradient leakage attacks.

(2) We provide a rigorous theoretical analysis demonstrating that our dropout-based federated learning strategy satisfies (ϵ, δ)-differential privacy.

(3) We conduct comprehensive experiments using image classification datasets and models to evaluate the effectiveness of our proposed method. The experimental results indicate that our method achieves robust defense against gradient leakage attacks while maintaining training efficiency and model accuracy compared to conventional approaches.

Notation: Define e as the base of the natural logarithm, $\mathbb{R}$ as the set of real numbers, and $\mathbb{R}^{m\times n}$ as the set of real matrices with m rows and n columns. For $x \in \mathbb{R}$, $|x|$ denotes its absolute value. For a matrix $X = \{x_{ij}\} \in \mathbb{R}^{m\times n}$, X^T denotes the transpose of X, $\|X\|_2 = \sqrt{X^T X}$ denotes the l_2 norm. Define $X \circ Y$ as the Hadamard product of matrices $X, Y \in \mathbb{R}^{m\times n}$, i.e., element-wise multiplication. Let I denote the identity matrix of appropriate dimensions. For a function $f : \mathbb{R}^{m\times n} \to \mathbb{R}$, the operator $\nabla_x f = \frac{\partial f}{\partial x}$ denotes the gradient of the function f with respect to the variable x, and $\arg\min f(x)$ denotes the value of x that minimizes the objective function $f(x)$. For a probability space, $\Pr[A]$ denotes the probability of event A occurring.

2 Related Work

2.1 Privacy and Security Challenges in Federated Learning

Federated learning improves privacy by keeping raw data local on devices; however, recent studies indicate that shared gradients can still leak sensitive information [7, 8]. Adversaries can intercept gradients transmitted from clients and reconstruct private training data through gradient inversion attacks. Zhu et al. [7] first demonstrated this by recovering images pixel-for-pixel from gradients, known as Deep Leakage from Gradients (DLG). Even when complete data recovery is challenging, attackers may infer sensitive attributes or class labels using model inversion and membership inference techniques [8, 9]. These threats demonstrate that federated learning alone cannot ensure robust privacy, highlighting the necessity for additional protective mechanisms.

2.2 Privacy-Preserving Strategies in Federated Learning

To mitigate privacy risks, researchers have explored several categories of privacy-preserving methods, each presenting limitations when compared with our approach:

Obfuscating Shared Updates: Techniques such as gradient compression and quantization reduce transmitted detail to enhance privacy and efficiency. Differential Privacy Stochastic Gradient Descent (DP-SGD), proposed by Abadi et al. [10], systematically adds noise to clipped gradients to ensure formal privacy guarantees. Although DP-SGD is principled, the added noise typically degrades model accuracy and convergence speed [11, 12]. Excessive noise can impair learning, while insufficient noise compromises privacy, reflecting an inherent accuracy-privacy trade-off [12].

Perturbing Local Data: Methods such as Mixup [13] and InstaHide [14] modify local data by mixing or obfuscating inputs before training. These approaches obscure raw data effectively, but typically lack rigorous theoretical privacy guarantees and may be vulnerable

to adaptive adversaries. Additionally, such data manipulation techniques can slightly degrade accuracy and complicate the training pipeline [15].

Cryptographic Methods: Secure Multi-Party Computation (SMC) and Homomorphic Encryption (HE) offer rigorous privacy by aggregating encrypted updates without revealing individual contributions [16]. However, these methods incur substantial computational and communication overheads, significantly increasing training latency and resource requirements. Fully homomorphic encryption, for instance, remains impractically slow and resource-intensive for real-world federated learning deployments [17].

In contrast to these existing methods, our proposed approach leverages inherent randomness during training—using dropout combined with gradient masking—to achieve strong privacy protection against gradient leakage attacks. This strategy not only maintains competitive accuracy and convergence rates but also circumvents the substantial computational and communication costs typical of cryptographic techniques, thereby providing a more practical and efficient privacy-preserving solution.

3 Preliminary

3.1 Federated Learning

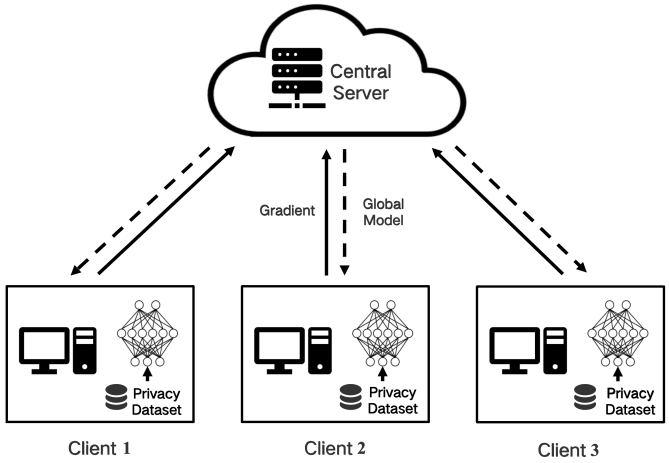


Figure 1. Federated Learning Framework.

Federated learning is a specialized distributed machine learning framework that enables collaborative model training in a decentralized setting by sharing gradients among participating nodes without exposing their local raw datasets.

As illustrated in Figure 1, a typical federated learning framework consists of the following training steps:

(1) The central server initializes a global learning model and sends the initial model parameters to each client;

(2) Each client trains the model locally using its private dataset and returns intermediate results (such as gradients or model weights) to the central server;

(3) The central server aggregates the received information according to a federated learning algorithm to update the global model, and then distributes the updated model back to all clients.

During the training process, the aggregation of models at the central server plays a critical role in federated learning. It determines how to combine the intermediate results from different clients to update the global model. One widely used aggregation algorithm is Federated Stochastic Gradient Descent (FedSGD) [6], as shown in Algorithm 1.

In a federated learning system using the FedSGD algorithm, during the t-th round of global iteration, each client i samples a training pair $(\mathbf{x}_{t,i}, \mathbf{y}_{t,i})$ from its local dataset after receiving the global model parameters, performs forward propagation using the local model, and computes the gradient of the loss function with respect to the model parameters via backpropagation:

$$\mathbf{g}_{t,i} = \nabla_{\mathbf{w}} \mathcal{L}(\mathbf{x}_{t,i}, \mathbf{y}_{t,i}) = \frac{\partial \mathcal{L}\left(F\left(\mathbf{x}_{t,i}, \mathbf{w}_t\right), \mathbf{y}_{t,i}\right)}{\partial \mathbf{w}_t}, \quad (1)$$

Then, each client sends its gradient to the central server for aggregation. The server computes the average of all received gradients and updates the global model using stochastic gradient descent with learning rate η:

$$\overline{\mathbf{g}_t} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{g}_{t,i}, \quad (2)$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \overline{\mathbf{g}_t}. \quad (3)$$

This iterative process continues until the global model converges or a predefined stopping condition is met.

3.2 Differential Privacy

Differential privacy is a widely adopted privacy standard that can be used to evaluate the effectiveness of privacy-preserving strategies in federated learning, as it provides a strong formal guarantee of privacy. The core idea of differential privacy is to obfuscate the influence of individual training samples on the model during the federated learning process, so that adversaries cannot accurately infer private information

Algorithm 1: Federated Stochastic Gradient Descent (FedSGD)

Data: Client set $\mathcal{C} = \{C_1, \ldots, C_n\}$, dataset $\mathcal{D} = \{D_1, \ldots, D_n\}$, model $F(\cdot)$, loss function $\mathcal{L}(\cdot)$
Result: Global model parameters $\mathbf{w}$
Initialize global model parameters $\mathbf{w}$ learning rate η and number of global iterations T;
for $t = 1$ **to** T **do**
  for $i = 1$ **to** n **do**
    Client C_i receives $\mathbf{w}$ from the server;
    Compute gradient: $\mathbf{g}_i = \nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}; \mathbf{x}_i, \mathbf{y}_i)$;
    Send $\mathbf{g}_i$ to the server;
  end
  Server receives $\{\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_n\}$ from clients;
  Aggregate gradient: $\overline{\mathbf{g}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{g}_i$;
  Update global model: $\mathbf{w} \leftarrow \mathbf{w} - \eta \overline{\mathbf{g}}$;
end
return $\mathbf{w}$

about any specific client. In practice, implementing differential privacy in federated learning typically involves adding random noise to the intermediate information exchanged during communication.

Definition 1. ***Adjacent Datasets****: Two datasets D and D' from the collection $\mathcal{D}$ are said to be adjacent, denoted as $adj(D, D')$, if they differ by only a single data point.*

Definition 2. (ϵ, δ)***-Differential Privacy****: A randomized algorithm $\mathcal{M}$ is said to satisfy (ϵ, δ)-differential privacy if, for any pair of adjacent datasets $D, D' \in \mathcal{D}$ and any subset $\mathcal{O} \subseteq \Theta$ of the output space, the following condition holds:*

$$\Pr[\mathcal{M}(D) \in \mathcal{O}] \leq e^{\epsilon} \Pr\left[\mathcal{M}\left(D'\right) \in \mathcal{O}\right] + \delta. \tag{4}$$

As stated in Equation (4) under Definition 2, for any subset $\mathcal{O}$ of the output space Θ, the probability that the output of $\mathcal{M}$ trained on adjacent datasets D and D' falls into $\mathcal{O}$ should differ by at most a multiplicative factor of e^{ϵ} and an additive term δ. Here, $\Pr[\mathcal{M}(D) \in \mathcal{O}]$ denotes the probability that the output of $\mathcal{M}$ trained on D lies in $\mathcal{O}$. The constants ϵ and δ are non-negative and characterize the strength of the differential privacy guarantee. A larger ϵ implies a weaker privacy guarantee, and in the limit $\epsilon \to \infty$, the learner $\mathcal{M}_{\infty}(D)$ behaves as if it provides no differential privacy. The parameter δ allows for a small probability of failure in the privacy guarantee, providing an additional degree of flexibility.

3.3 Privacy Attacks in Federated Learning

Threat Model Federated learning preserves privacy by design, as clients do not share raw data but instead gradients to collaboratively train a global model. However, these updates may reveal sensitive information if intercepted by adversaries during communication. Recent studies have shown that private data can be reconstructed from shared gradients [7].

There are two primary threats to data privacy in federated learning [18]:

(1) In each training round, clients upload their locally computed model updates to the central server. During transmission, malicious actors may compromise certain clients, intercept their updates, and exploit knowledge of the model architecture to infer private training data—a so-called white-box attack.

(2) The central server itself may be honest-but-curious: it follows the training protocol faithfully but may log and analyze client updates to infer sensitive information from specific participants.

In summary, privacy risks in federated learning arise from either compromised clients during communication or untrusted servers analyzing shared updates.

Gradient Leakage Attack As shown in Algorithm 2, when the FedSGD algorithm is used in a federated learning network and gradients are transmitted as intermediate information, an adversary can perform the Deep Leakage from Gradients (DLG) attack [7] to iteratively update dummy samples so that the computed dummy gradients closely match the real ones, thereby reconstructing the original training samples. The overall workflow of the DLG algorithm is illustrated in Figure 2.

Algorithm 2: Deep Leakage from Gradients (DLG)

Data: Target model $F(\cdot)$, stolen gradient $\mathbf{g}^*$
Result: Recovered training sample $\mathbf{x}', \mathbf{y}'$
Initialize learning rate μ, number of iterations n and dummy inputs: $\mathbf{x}'_1 \sim \mathcal{N}(0, 1)$, $\mathbf{y}'_1 \sim \mathcal{N}(0, 1)$;
for $i = 1$ **to** n **do**
  Compute dummy gradient: $\mathbf{g}'_i = \nabla_{\mathbf{w}} \mathcal{L}(\mathbf{x}'_i, \mathbf{y}'_i)$;
  Compute gradient distance: $\mathbb{D}_{G_i} = \|\mathbf{g}'_i - \mathbf{g}^*\|_2$;
  Update dummy input: $\mathbf{x}'_{i+1} \leftarrow \mathbf{x}'_i - \mu \nabla_{\mathbf{x}'_i} \mathbb{D}_{G_i}$;
  Update dummy label: $\mathbf{y}'_{i+1} \leftarrow \mathbf{y}'_i - \mu \nabla_{\mathbf{y}'_i} \mathbb{D}_{G_i}$;
end
return $\mathbf{x}', \mathbf{y}'$

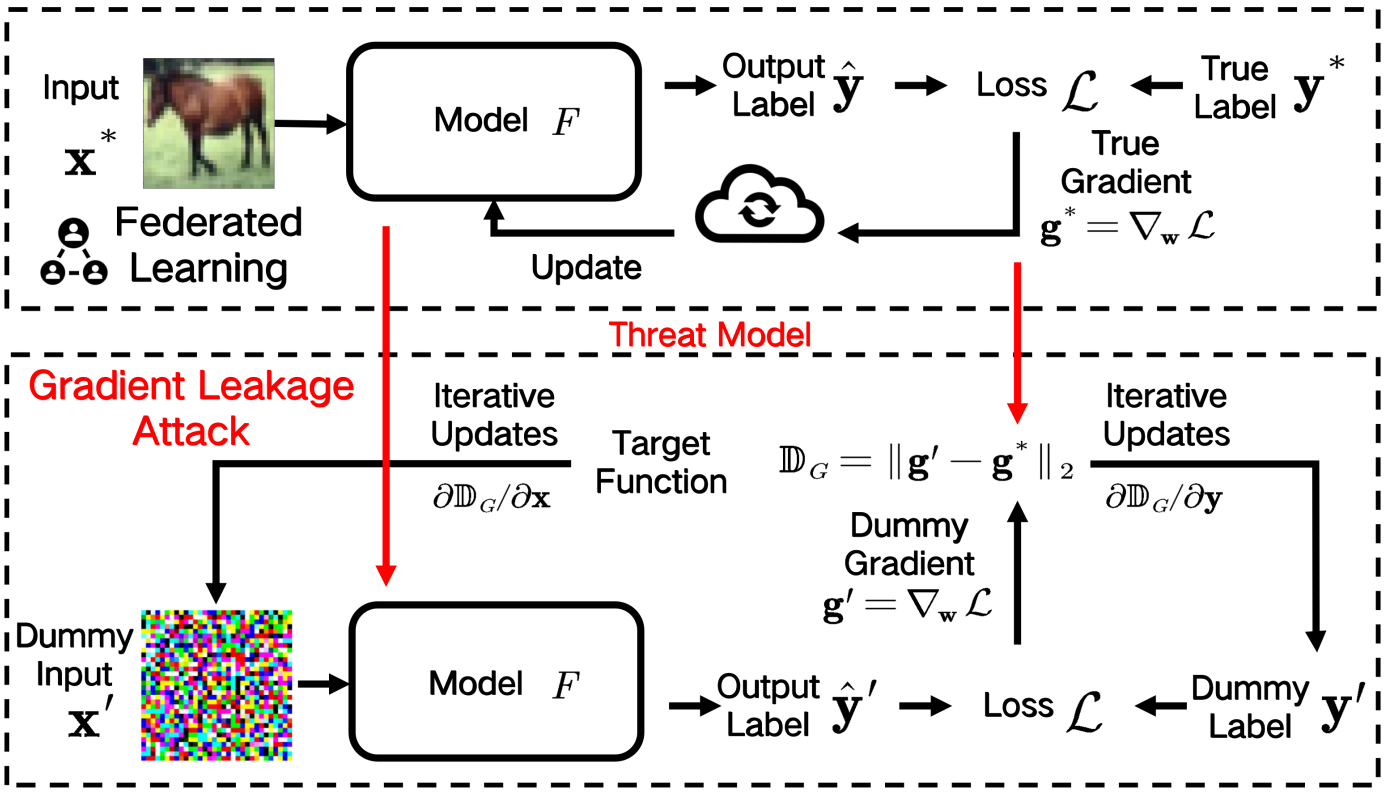


Figure 2. Workflow of the DLG algorithm.

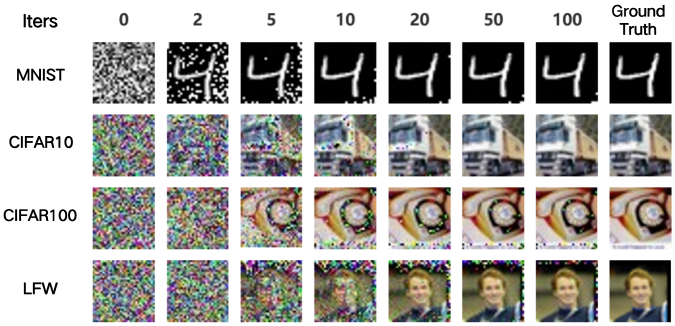


Figure 3. DLG reconstruction results on image datasets.

Given the threat model, an adversary can obtain the true gradients $\mathbf{g}^*$ transmitted by other clients, along with the target model $F(\cdot)$ and its parameter $\mathbf{w}$. The goal of the DLG algorithm is to recover the original training sample $\mathbf{x}^*, \mathbf{y}^*$ using only this information.

To recover the data from gradients, DLG first randomly initializes a dummy image input $\mathbf{x}'$ and dummy label $\mathbf{y}'$. These dummy samples are then fed into the model to compute the dummy gradient:

$$\mathbf{g}' = \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{x}', \mathbf{y}') = \frac{\partial\mathcal{L}\left(F\left(\mathbf{x}', \mathbf{w}\right), \mathbf{y}'\right)}{\partial\mathbf{w}}. \tag{5}$$

By iteratively optimizing the dummy samples so that the dummy gradient approaches the real gradient, the dummy inputs are gradually updated to approximate the true training data. Given the stolen gradient $\mathbf{g}^*$, the reconstruction objective can be formulated as minimizing the following loss:

$$\begin{aligned}(\mathbf{x}'^*, \mathbf{y}'^*) &= \arg\min_{\mathbf{x}', \mathbf{y}'} \mathbb{D}_G \\ &= \arg\min_{\mathbf{x}', \mathbf{y}'} \left\| \frac{\partial\mathcal{L}\left(F\left(\mathbf{x}', \mathbf{w}\right), \mathbf{y}'\right)}{\partial\mathbf{w}} - \mathbf{g}^* \right\|_2.\end{aligned} \tag{6}$$

By computing the gradient of the objective $\mathbb{D}_G$ with respect to the dummy input $\mathbf{x}'$ and label $\mathbf{y}'$, the dummy samples are updated iteratively via gradient descent with learning rate μ:

$$\begin{aligned}\mathbf{x}' &\leftarrow \mathbf{x}' - \mu\nabla_{\mathbf{x}'}\mathbb{D}_G(\mathbf{x}', \mathbf{y}'), \\ \mathbf{y}' &\leftarrow \mathbf{y}' - \mu\nabla_{\mathbf{y}'}\mathbb{D}_G(\mathbf{x}', \mathbf{y}').\end{aligned} \tag{7}$$

Because gradient-based iterative optimization involves second-order derivatives, the loss function in Equation (6) must be differentiable with respect to both the dummy input $\mathbf{x}'$ and dummy label $\mathbf{y}'$. Therefore, applying the DLG algorithm requires that the model $F(\cdot)$ is twice differentiable, which is generally satisfied in modern deep learning architectures, such as convolutional neural networks (CNNs) with sigmoid activation functions.

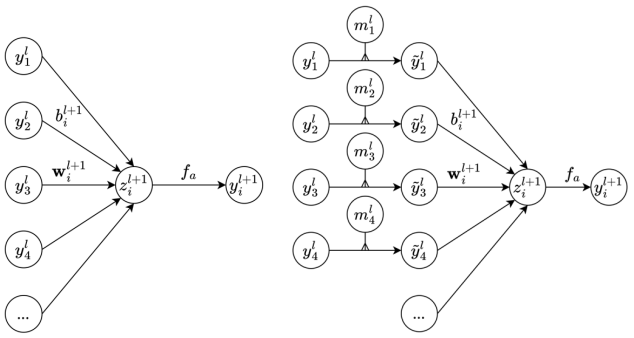
Figure 3 shows the recovered samples on MNIST [19], CIFAR10 [20], CIFAR100 [20], and LFW [21] datasets

after iterative updates using the DLG algorithm. These results demonstrate that DLG can successfully reconstruct data samples that are visually similar to the original training data, thereby posing a serious threat to privacy.

4 Methodology

Considering the shortcomings of existing privacy protection methods in terms of computational accuracy and communication efficiency, this paper proposes a novel idea for privacy protection. The proposed method introduces randomness during model training to fulfill differential privacy requirements, thereby effectively defending against privacy attacks such as gradient leakage attacks. The randomness during model training is introduced by applying dropout method with a specified probability [22].

4.1 Principle of the Privacy Protection Strategy Based on Dropout Method



(a) Standard Neuron Architectures. (b) Neuron Architecture with Dropout Method Applied.

Figure 4. Mathematical Model of Dropout Method in Neurons.

The classical dropout method is a common technique to prevent overfitting during neural network training. Its principle involves randomly selecting neurons in certain layers of the neural network during the forward propagation process, and disconnecting their connections with the inputs with probability p. This strategy enhances the generalization of the learning model, reducing reliance on specific local features in the training dataset.

As shown in Figure 4, the mathematical model of dropout method can be analyzed at the single-neuron level. For the standard neuron structure shown in Figure 4(a), the input-output relation satisfies:

$$\begin{aligned} z_i^{l+1} &= \mathbf{w}_i^{l+1}\mathbf{y}^l + b_i^{l+1}, \\ y_i^{l+1} &= f_a\left(z_i^{l+1}\right). \end{aligned} \tag{8}$$

For the neuron structure with dropout method applied, shown in Figure 4(b), the input-output relation satisfies:

$$\begin{aligned} r_j^l &\sim \mathcal{B}(p), \\ \widetilde{\mathbf{y}}^l &= \mathbf{y}^l - (\mathbf{m}^l \circ \mathbf{y}^l), \\ z_i^{l+1} &= \mathbf{w}_i^{l+1}\widetilde{\mathbf{y}}^l + b_i^{l+1}, \\ y_i^{l+1} &= f_a\left(z_i^{l+1}\right). \end{aligned} \tag{9}$$

The dropout method randomly disconnects neuron-input connections by subtracting the Hadamard product of the input vector $\mathbf{y}^l$ and a dropout indicator vector $\mathbf{m}^l$ (with elements drawn from a Bernoulli distribution). Specifically, if an input element y_n^l corresponds to $m_n^l = 1$, the connection is disabled, and the gradient corresponding to weight w_n^{l+1} is ignored during the training iteration. By applying dropout method, excessive neuron weights can be mitigated, preventing overfitting during neural network training.

Algorithm 3: Dropout-based Federated Stochastic Gradient Descent (Drop-FedSGD)

Data: Client set $\mathcal{C} = \{C_1, \ldots, C_n\}$, dataset $\mathcal{D} = \{D_1, \ldots, D_n\}$, model $F(\cdot)$, loss function $\mathcal{L}(\cdot)$, Gaussian noise variance σ^2, dropout probability p, gradient norm bound c

Result: Global model parameters $\mathbf{w}$

Initialize global model parameters $\mathbf{w}$, learning rate η, and number of global iterations T;

for $t = 1$ **to** T **do**

  for $i = 1$ **to** n **do**

    Client C_i receives $\mathbf{w}$ from the server;

    Generate dropout mask $M_{\text{drop}} \sim \mathcal{B}(p)$;

    Compute gradient with dropout: $\mathbf{g}_i = \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w} - (\mathbf{w} \circ M_{\text{drop}}); \mathbf{x}_i, \mathbf{y}_i)$;

    Clip gradient: $\breve{\mathbf{g}}_i = \mathbf{g}_i / \max\left(1, \frac{\|\mathbf{g}_i\|_2}{c}\right)$;

    Generate Gaussian noise $\xi_i \sim \mathcal{N}(0, \sigma^2 c^2 I)$;

    Send $\tilde{g}_i = \breve{g}_i + \xi_i \circ M_{\text{drop}}$ to the server;

  end

  Server receives $\{\tilde{\mathbf{g}}_1, \tilde{\mathbf{g}}_2, \ldots, \tilde{\mathbf{g}}_n\}$ from clients;

  Aggregate gradient: $\overline{\mathbf{g}} = \frac{1}{n}\sum_{i=1}^{n} \tilde{\mathbf{g}}_i$;

  Update global model: $\mathbf{w} \leftarrow \mathbf{w} - \eta\overline{\mathbf{g}}$;

end

return $\mathbf{w}$

Furthermore, to prevent attackers from easily identifying disabled neuron connections from gradients, Gaussian noise ξ_i is applied to corresponding weight gradients. The noise level is deliberately small to minimize its impact on training

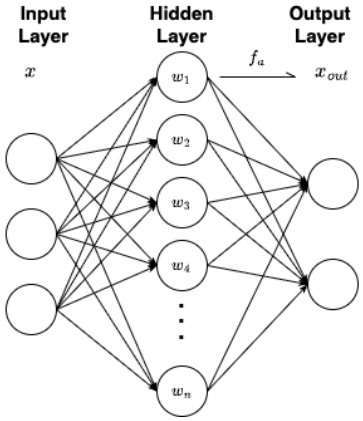


Figure 5. Neural network with weights.

accuracy. When attempting to reconstruct gradients trained with dropout method, attackers cannot precisely determine which weight gradients are inactive, making it difficult to identify an appropriate objective function for optimizing dummy samples $\mathbf{x}'$ and $\mathbf{y}'$. After introducing dropout into the FedSGD algorithm, we obtain Drop-FedSGD, as illustrated in Algorithm 3.

4.2 Differential Privacy Analysis of the Dropout Method

Through theoretical analysis, we demonstrate that Algorithm 3, which introduces randomness into model training, satisfies differential privacy:

Theorem 1. *Consider Algorithm 3, where M_{drop} follows a Bernoulli distribution $\mathcal{B}(p)$ and Gaussian noise ξ_i follows $\mathcal{N}(0,\sigma^2)$. Given $p \in (0,1)$ and differential privacy parameters $\epsilon \geq 0$, $\delta \in (1-p^2,1)$, if we set $\sigma \geq \frac{\Delta c}{\epsilon}$, then Algorithm 3 achieves (ϵ,δ)-differential privacy, where Δ denotes the output range and*

$$c = \sqrt{2\ln\left(\frac{1.25(1-p)^2}{\delta - p^2 - 2p(1-p)}\right)}. \tag{10}$$

Proof. We focus on a particular hidden layer in a neural network, as shown in Figure 5. Suppose the weight matrix of this hidden layer is $\mathbf{w} \in \mathbb{R}^{h_1 \times h_2}$, where h_1 and h_2 represent the numbers of input and output neurons in this layer, respectively. For input x, the output of the hidden layer is:

$$x^{out} = f_a(\mathbf{w}^T x) = \begin{bmatrix} f_a(w_1^T x) \\ \cdots \\ f_a(w_n^T x) \end{bmatrix}, \tag{11}$$

where w_i is the weight vector of each neuron. Applying the chain rule, the gradient of the overall loss function with respect to w_i is:

$$\frac{\partial \mathcal{L}}{\partial w_i} = \frac{\partial \mathcal{L}}{\partial x_i^{out}} \cdot \frac{\partial x_i^{out}}{\partial w_i} = \frac{\partial \mathcal{L}}{\partial x_i^{out}} \cdot \partial_{w_i^T x} f_a(w_i^T x) \cdot x. \tag{12}$$

When using the dropout method in neural network training, the connections of neurons are randomly disabled with probability p. Thus, the output and gradient of the loss function with respect to w_i become:

$$\hat{x}^{out}(x,\mathbf{w}) = f_a((\mathbf{w}^T - (\mathbf{w} \circ M_{drop})^T)x), \tag{13}$$

$$\frac{\partial \hat{\mathcal{L}}}{\partial w_i} = \frac{\partial \hat{\mathcal{L}}}{\partial \hat{x}_i^{out}} \cdot \partial_{w_i^T x} f_a(w_i^T x \cdot (x \circ m_i)), \tag{14}$$

where $M_{drop} \in \mathbb{R}^{h_1 \times h_2}$ is the dropout indicator matrix, whose elements are independently sampled from the Bernoulli distribution $\mathcal{B}(p)$, and m_i denotes the i-th column of M_{drop}.

Considering that neural networks employing dropout tend to stabilize in loss function values across different random settings during training, we approximate Eq. (14) as:

$$\frac{\partial \hat{\mathcal{L}}}{\partial w_i} = \frac{\partial \mathcal{L}}{\partial x_i^{out}} \cdot \partial f_a(w_i^T x) \cdot (x \circ m_i). \tag{15}$$

To prevent attackers from easily identifying disabled neurons, we add Gaussian noise ξ to the gradients:

$$\begin{aligned} \mathbf{g}_i = \mathcal{M}(w_i) &= \frac{\partial \hat{\mathcal{L}}}{\partial w_i} + \xi \circ m_i \\ &= (\frac{\partial \mathcal{L}}{\partial x_i^{out}} \cdot \partial f_a(w_i^T x) \cdot x + \xi) \circ m_i, \end{aligned} \tag{16}$$

where the Gaussian noise $\xi \in \mathbb{R}^{h_1 \times h_2}$ is independently drawn from the normal distribution $\mathcal{N}(0,\sigma^2)$.

Correspondingly, the outputs obtained from training on adjacent datasets using the randomized algorithm are:

$$\mathcal{M}(w_i') = (\frac{\partial \mathcal{L}}{\partial x_i^{out}} \cdot \partial f_a(w_i'^T x) \cdot x + \xi') \circ m_i'. \tag{17}$$

For adjacent datasets w_i and w_i', differential privacy requires suitable parameters ϵ, δ that satisfy:

$$\Pr[\mathcal{M}(w_i) \in \mathcal{O}] \leq e^{\epsilon} \Pr[\mathcal{M}(w_i') \in \mathcal{O}] + \delta. \tag{18}$$

For simplicity, we consider w_i as a one-dimensional case (higher-dimensional cases can be generalized).

Given that the random variables m_i and m_i' in the outputs $\mathcal{M}(w_i)$ and $\mathcal{M}(w_i')$ follow independent Bernoulli distributions $\mathcal{B}(p)$, we have:

$$\begin{aligned} \Pr[m_i = 1] &= p, \\ \Pr[m_i = 0] &= 1 - p. \end{aligned} \tag{19}$$

The subsets of outputs $\mathcal{O}$ from adjacent datasets can be discussed as follows:

(1) The probability of both adjacent datasets having non-zero Bernoulli outputs simultaneously is:

$$\Pr[\mathcal{O}_1] = \Pr[m_i = 1 \text{ or } m_i' = 1] = p^2 + 2p(1-p). \tag{20}$$

In this case, subset $\mathcal{O}_1 = \varnothing$.

(2) The probability of both adjacent datasets having zero Bernoulli outputs simultaneously is:

$$\Pr[\mathcal{O}'] = \Pr[m_i = 0 \text{ and } m_i' = 0] = (1-p)^2. \tag{21}$$

Here, we have $\mathcal{M}(w_i) = \xi$, $\mathcal{M}(w_i') = \xi'$.

For given $\epsilon \in [0, 1]$ and $\delta > 1 - p^2 - 2p(1-p)$, we set the variance condition as:

$$\sigma^2 \geq \left(\frac{\Delta c}{\epsilon}\right)^2, \tag{22}$$

where Δ represents the output range of the algorithm, and the clipping boundary c is defined by Eq. (10).

Then, by defining event sets:

$$\begin{aligned} \mathcal{O}_2 &= \{\mathcal{M}(w_i') : |\xi'| > c\Delta/\epsilon\}, \\ \mathcal{O}_3 &= \mathcal{O}' - \mathcal{O}_2, \end{aligned} \tag{23}$$

it can be easily demonstrated based on the literature [23] that:

$$\begin{aligned} \Pr[\mathcal{M}(w_i) \in \mathcal{O}] &= \Pr[\mathcal{M}(w_i') \in \mathcal{O}_1] \\ &\quad + \Pr[\mathcal{M}(w_i) \in \mathcal{O}_2] + \Pr[\mathcal{M}(w_i) \in \mathcal{O}_3] \\ &\leq \Pr[\mathcal{M}(w_i) \in \mathcal{O}_3] + \delta \\ &\leq e^{\epsilon} \Pr[\mathcal{M}(w_i) \in \mathcal{O}_3] + \delta. \end{aligned} \tag{24}$$

Thus, the proposed method achieves (ϵ, δ)-differential privacy. This completes the proof. □

5 Experiments

5.1 Experimental Setup and Implementation of Federated Learning Network

In the privacy attack-defense simulation experiments, the hardware setup includes: Intel(R) Xeon(R) Silver 4210R CPU @ 2.40GHz, 128GB RAM, NVIDIA GeForce RTX 3090 GPU with 24GB of memory. The software environment consists of Ubuntu 20.04.5 OS, Python 3.9.1, and PyTorch 2.1.0.

The experiments simulate a horizontal federated learning network with one central server and four clients using multi-processing with multi-core CPUs on a single machine. The federated learning network uses MNIST [19] and CIFAR10 [20] datasets, and the learning models employed are LeNet and ResNet, respectively, with Sigmoid activation functions. The aggregation algorithm on the central server is FedSGD.

In the federated learning network constructed via multiprocessing, the central server runs in the main process and aggregates gradients from clients to generate the global model. The main process runs in parallel with three subprocesses, each representing a client. These clients receive commands and global model weights from the server, train on their local datasets, and share gradients to the server. Each client includes the following two major functions:

(1) **Configuration Function**: Copies main configuration settings (model selection, local epochs, learning rate, etc.) to each client, builds the local model based on the configuration, and trains it using the local dataset. In horizontal FL training, the MNIST and CIFAR10 datasets are partitioned by client ID using `torchvision.datasets`, ensuring non-overlapping subsets among clients.

(2) **Collective Communication Function**: Implements communication operations such as broadcast, reduce, scatter, and gather using message-passing interfaces, enabling inter-process gradient transmission.

The training process of the federated learning network is shown in Figure 6 and includes the following steps:

(1) Load configuration parameters (including model, learning rate, batch size, epochs, etc.);

(2) Create multiple threads to simulate the central server and each client, initializing local models;

(3) Perform global iterations: each client sends gradients to the server, which aggregates and sends updated global model parameters back to the clients;

(4) Terminate the process group after iterations.

In the federated learning network using the FedSGD algorithm, client privacy data can be attacked using the DLG algorithm. An adversary may steal gradients sent from clients to the server and use them to iteratively

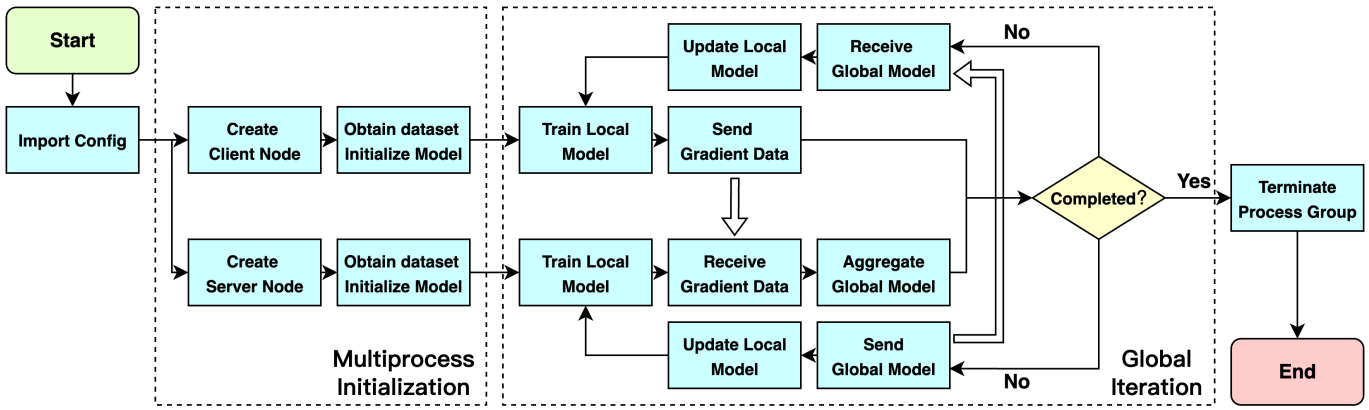


Figure 6. Training Workflow of Federated Learning Network with Multiprocessing.

optimize dummy samples in the DLG attack. As shown in Figure 3, the adversary can recover private client data.

To quantify the defense effectiveness against gradient leakage attacks, Mean Square Error (MSE) is used to measure the difference between the reconstructed and true images. The formula is:

$$\mathrm{MSE}(I,K)=\frac{1}{M\cdot N}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\big[I(i,j)-K(i,j)\big]^2, \tag{25}$$

where M and N denote the image height and width, respectively. A lower MSE indicates higher similarity. In addition, we report the Success Rate (SR) to evaluate the proportion of successful gradient-leakage attacks. An attack is counted as successful if the DLG optimization converges such that the gradient distance in Eq. (6) falls below 0.1; we cap the maximum number of DLG iterations at 20 and use a learning rate of 0.01. A higher SR indicates greater vulnerability to privacy leakage.

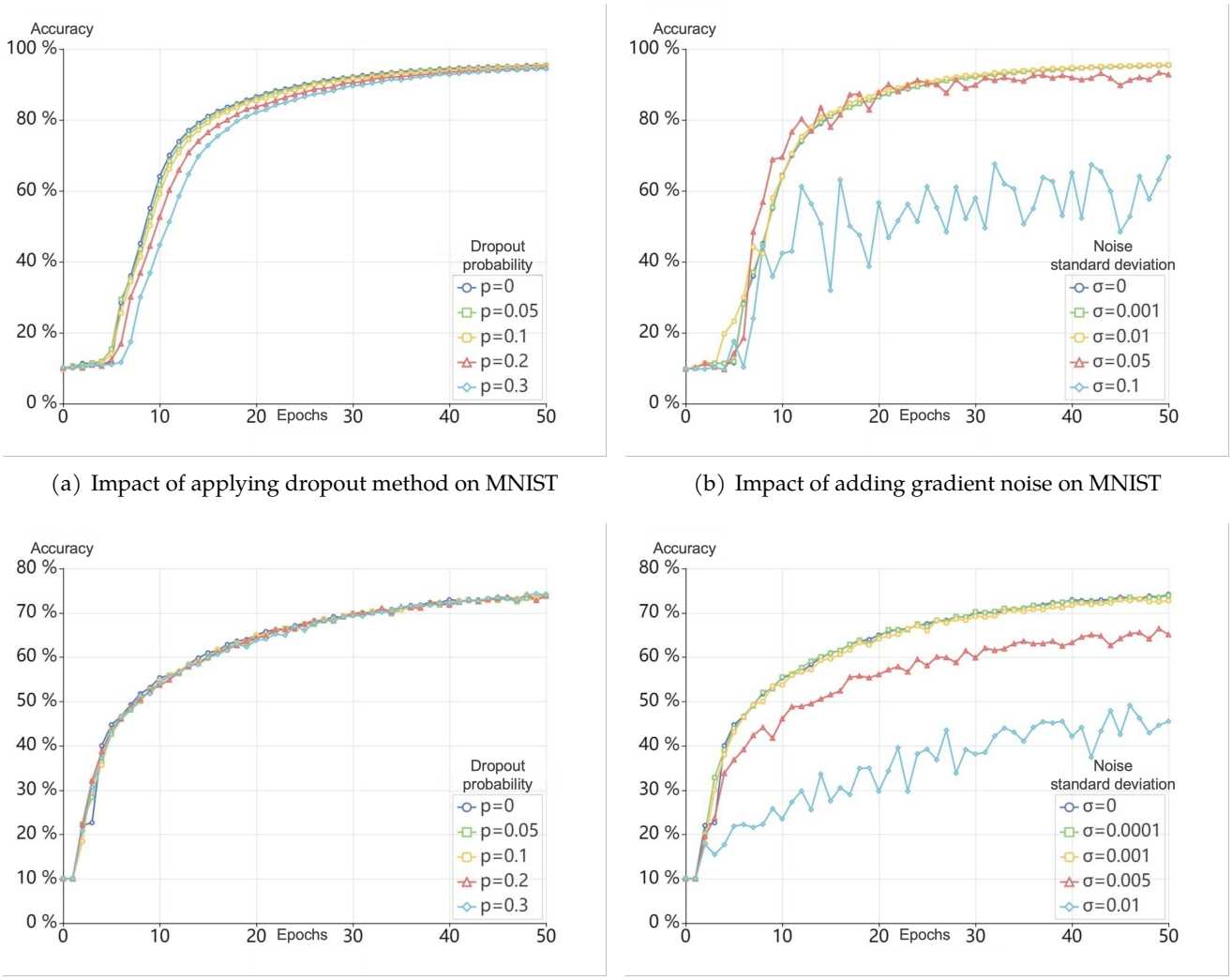
We use the DLG algorithm to reconstruct individual samples from MNIST and CIFAR10 using neural networks trained with various defense strategies, and evaluate: (i) the attack Success Rate (SR); (ii) the average image-space error via MSE; and (iii) perceptual fidelity via SSIM (higher is better, $[0,1]$) and LPIPS (lower is better, typically $[0,1+]$). Results are summarized in Table 1. It is evident that when the dropout probability $p \geq 0.05$ or the noise standard deviation $\sigma \geq 0.05$, the attack SR drops below 10%, effectively mitigating gradient leakage; meanwhile, SSIM decreases and LPIPS increases as defense strength grows, which indicates poorer reconstructions—consistent with rising MSE and falling SR.

5.2 Impact of Privacy Protection on Federated Learning Training Performance

Table 1. DLG reconstruction success rate and MSE under different privacy protection strategies.

Dataset	Metric	Parameter			
Dropout p		**0**	**0.01**	**0.05**	**0.1**
MNIST	SR	100%	47%	2%	1%
	MSE	0.05	0.38	1.12	4.26
	SSIM	0.371	0.280	0.120	0.050
	LPIPS	0.201	0.320	0.550	0.720
CIFAR10	SR	99%	41%	2%	1%
	MSE	0.03	0.33	1.37	3.17
	SSIM	0.124	0.100	0.050	0.020
	LPIPS	0.532	0.600	0.750	0.850
Noise $\mathcal{N}(0,\sigma^2)$		**0**	**0.005**	**0.01**	**0.05**
MNIST	SR	100%	61%	12%	6%
	MSE	0.05	0.93	1.78	3.36
	SSIM	0.371	0.250	0.150	0.080
	LPIPS	0.201	0.350	0.500	0.650
CIFAR10	SR	99%	53%	10%	5%
	MSE	0.32	0.81	1.33	2.21
	SSIM	0.124	0.090	0.060	0.030
	LPIPS	0.532	0.620	0.720	0.820

We train the proposed federated learning algorithm using various dropout probabilities and compare it with traditional noise-based differential privacy defenses. Figure 7(a) and Figure 7(c) illustrate how Drop-FedSGD affects classification accuracy on MNIST and CIFAR10 under different p values. Results show that dropout has little to no negative impact on accuracy; appropriate p values may even improve performance over the baseline without privacy protection. Moreover, $p = 0.05$ or higher



(a) Impact of applying dropout method on MNIST

(b) Impact of adding gradient noise on MNIST

(c) Impact of applying dropout method on CIFAR10

(d) Impact of adding gradient noise on CIFAR10

Figure 7. Impact of different privacy protection strategies on federated learning model performance

Table 2. Training time of federated learning under different privacy protection strategies.

Defense	Fwd/Bwd	Comm.	Priv. Acc.	Total
None	2500	1205	0	3705
Dropout	2400	1205	41	3646
Noise	2500	1205	827	4532
HE	4000	5000	3073	12073

reduces the data recovery probability to below 2%.

Figure 7(b) and Figure 7(d) demonstrate that increasing the noise standard deviation σ leads to decreased model accuracy. For example, with $\sigma = 0.01$ on MNIST using LeNet, accuracy remains stable, yet DLG still succeeds 12% of the time. Larger σ values improve privacy but degrade accuracy. Similarly, for ResNet on CIFAR10, $\sigma = 0.005$ yields a 61% success rate for DLG, rendering it ineffective. Larger noise reduces attack success but also harms model accuracy.

We also compare the time required to train the FL network for 50 epochs on MNIST using different strategies, as shown in Table 2. To provide finer insight, we profile runtime into three components: forward/backward computation (Fwd/Bwd), communication (Comm.), and privacy accounting (Priv. Acc.; e.g., noise generation, clipping, encryption). For dropout, random masks are generated and applied locally on each client without upload, incurring negligible Priv. Acc. overhead and no extra Comm. cost. Noise-based defense adds moderate Priv. Acc. overhead for gradient perturbation, whereas homomorphic encryption (via PySyft [24]) substantially inflates all components due to cryptographic computation and transmission. Noise and homomorphic encryption increase total training time by 22% and 226%, respectively, versus

plain FedSGD, whereas dropout slightly reduces Fwd/Bwd time due to neuron deactivation while maintaining similar Comm. cost.

In summary, compared to other defense methods, dropout-based privacy strategy exhibits strong resistance to gradient leakage attack while minimally impacting training performance. This makes it a promising privacy-preserving strategy for federated and distributed learning systems.

6 Conclusion

This paper tackles the privacy challenges in federated learning by introducing a novel defense strategy that capitalizes on the inherent randomness of model training. Departing from conventional gradient noise-addition methods, our approach integrates stochasticity into local training through randomized neuron connection deactivation, augmented with Gaussian noise injection to obscure potential vulnerabilities against adversarial analysis. Comprehensive simulations on privacy attacks and defenses demonstrate that this randomness-based strategy not only effectively counters gradient leakage threats but also maintains the computational efficiency and model accuracy essential for practical federated learning applications, thereby offering a robust and viable solution for privacy-preserving collaborative learning systems.

Data Availability Statement

Data will be made available on request.

Funding

This work was supported in part by the National Natural Science Foundation of China under Grant U24A20264 and Grant 62273041; in part by the Open Projects of the Institute of Systems Science, Beiiing Wuzi University under Grant BWUISS11.

Conflicts of Interest

The authors declare no conflicts of interest.

Ethical Approval and Consent to Participate

Not applicable.

References

[1] He, K., Zhang, X., Ren, S., & Sun, J. (2016, June). Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770-778). IEEE. [CrossRef]

[2] Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., & Sainath, T. N. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine, 29*(6), 82–97. [CrossRef]

[3] Grigorescu, S., Trasnea, B., Cocias, T., & Macesanu, G. (2020). A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics, 37*(3), 362–386. [CrossRef]

[4] Li, Q., Diao, Y., Chen, Q., & He, B. (2022, May). Federated learning on non-iid data silos: An experimental study. In *2022 IEEE 38th international conference on data engineering (ICDE)* (pp. 965-978). IEEE. [CrossRef]

[5] Zhang, C., Xie, Y., Bai, H., Yu, B., Li, W., & Gao, Y. (2021). A survey on federated learning. *Knowledge-Based Systems, 216*, 106775. [CrossRef]

[6] McMahan, B., Moore, E., Ramage, D., Hampson, S., & y Arcas, B. A. (2017, April). Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics* (pp. 1273-1282). PMLR.

[7] Zhu, L., Liu, Z., & Han, S. (2019). Deep leakage from gradients. *Advances in neural information processing systems, 32*. [CrossRef]

[8] Fredrikson, M., Lantz, E., Jha, S., Lin, S., Page, D., & Ristenpart, T. (2014). Privacy in pharmacogenetics: An {End-to-End} case study of personalized warfarin dosing. In *23rd USENIX security symposium (USENIX Security 14)* (pp. 17-32). [CrossRef]

[9] Nasr, M., Shokri, R., & Houmansadr, A. (2019, May). Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE symposium on security and privacy (SP)* (pp. 739-753). IEEE. [CrossRef]

[10] Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., & Zhang, L. (2016). Deep learning with differential privacy. *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pp. 308–318. [CrossRef]

[11] Tramer, F., & Boneh, D. (2020). Differentially private learning needs better features (or much more data). *arXiv preprint arXiv:2011.11660*.

[12] Yu, D., Zhang, H., Chen, W., Yin, J., & Liu, T. Y. (2021, July). Large scale private learning via low-rank reparametrization. In *International Conference on Machine Learning* (pp. 12208-12218). PMLR.

[13] Zhang, H., Cisse, M., Dauphin, Y. N., & Lopez-Paz, D. (2017). mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.

[14] Huang, Y., Song, Z., Li, K., & Arora, S. (2020,

November). Instahide: Instance-hiding schemes for private distributed learning. In *International conference on machine learning* (pp. 4507-4518). PMLR.

[15] Carlini, N., Deng, S., Garg, S., Jha, S., Mahloujifar, S., Mahmoody, M., ... & Tramèr, F. (2021, May). Is private learning possible with instance encoding?. In *2021 IEEE Symposium on Security and Privacy (SP)* (pp. 410-427). IEEE. [CrossRef]

[16] Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., ... & Seth, K. (2017, October). Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (pp. 1175-1191). [CrossRef]

[17] Li, T., Sahu, A. K., Talwalkar, A., & Smith, V. (2020). Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine, 37*(3), 50-60. [CrossRef]

[18] Enthoven, D., & Al-Ars, Z. (2021). An overview of federated deep learning privacy attacks and defensive strategies. *Federated Learning Systems: Towards Next-Generation AI*, 173-196. [CrossRef]

[19] Deng, L. (2012). The MNIST database of handwritten digit images for machine learning research [Best of the Web]. *IEEE Signal Processing Magazine, 29*(6), 141–142. [CrossRef]

[20] Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images.

[21] Huang, G. B., Mattar, M., Berg, T., & Learned-Miller, E. (2008). Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. In *Workshop on faces in'Real-Life'Images: detection, alignment, and recognition*.

[22] Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

[23] Dwork, C., & Roth, A. (2014). The algorithmic foundations of differential privacy. Foundations and Trends in Theoretical Computer Science, 9(3–4), 211–407. [CrossRef]

[24] Ryffel, T., Trask, A., Dahl, M., Wagner, B., Mancuso, J., Rueckert, D., & Passerat-Palmbach, J. (2018). A generic framework for privacy preserving deep learning. *arXiv preprint arXiv:1811.04017*.

Zhiyuan Wang received the B.S. degree in electrical engineering and automation from Beijing Institute of Technology, Beijing, China, in 2023. He is currently working toward the M.S. degree in control science and engineering at Beijing Institute of Technology, Beijing, China. His current research interests include privacy security and machine learning. (Email: zhiyuanwang@bit.edu.cn)

Ming Tian received the B.S. degree in automation from Beijing University of Chemical Technology, Beijing, China, in 2023. He is currently working toward the M.S. degree in control science and engineering at Beijing Institute of Technology, Beijing, China. His current research interests include privacy security and large language models. (Email: mingtian@bit.edu.cn)

Yutao Qiu received the B.S. degree in automation from Beijing Institute of Technology, Beijing, China, in 2024. He is currently working toward the M.S. degree in control science and engineering at Beijing Institute of Technology, Beijing, China. His current research interests include embodied intelligence and safe reinforcement learning. (Email: 3120240873@bit.edu.cn)

Zhixu Zhao received the B.S. degree in electrical engineering and automation from China University of Geosciences, Beijing, China,in 2024. He is currently working toward the M.S. degree in control science and engineering at Beijing Institute of Technology, Beijing, China. His current research interests include dversarial attack and machine learning. (Email: zhixuzhaobit@163.com)

Yongpo Zhang received the M.S.degree in Applied Mathematics from the University of Science and Technology Beijing in 2007, and joined the Aviation Basic College of the Air Force Aviation University in 2007. His current research interests include machine learning and biomathematics. (Email: haiquan1234@126.com)

Kun Liu received the Ph.D. degree in electrical engineering and systems from Tel Aviv University, Tel Aviv-Yafo, Israel, in 2012. From 2013 to 2015, he was a Postdoctoral Researcher with the ACCESS Linnaeus Centre, KTH Royal Institute of Technology, Stockholm, Sweden. In 2015, he held Researcher, Visiting, and Research Associate positions with, respectively, the KTH Royal Institute of Technology; CNRS, Laboratory for Analysis and Architecture of Systems, Toulouse, France; and the University of Hong Kong, Hong Kong. In 2018, he was a Visiting Scholar with INRIA, Lille, France. In 2015, he joined the School of Automation, Beijing Institute of Technology, Beijing, China. His current research interests include networked control, game-theoretic control, and security and privacy of cyber-physical systems, with applications in autonomous systems. Dr. Liu currently serves as an Associate Editor for the IMA Journal of Mathematical Control and Information and the Journal of Beijing Institute of Technology. He is a Conference Editorial Board Member of the IEEE Control Systems Society. (Email: kunliubit@bit.edu.cn)