Preserving Differential Privacy in Deep Learning Based on Feature Relevance Region Segmentation

Fangwei Wang, Meiyun Xie, Zhiyuan Tan, *Member, IEEE,* Qingru Li, and Changguang Wang

Abstract—In the era of big data, deep learning techniques provide intelligent solutions for various problems in real-life scenarios. However, deep neural networks depend on large-scale datasets including sensitive data, which causes the potential risk of privacy leakage. In addition, various constantly evolving attack methods are also threatening the data security in deep learning models. Protecting data privacy effectively at a lower cost has become an urgent challenge. This paper proposes an Adaptive Feature Relevance Region Segmentation (AFRRS) mechanism to provide differential privacy preservation. The core idea is to divide the input features into different regions with different relevance according to the relevance between input features and the model output. Less noise is intentionally injected into the region with stronger relevance, and more noise is injected into the regions with weaker relevance. Furthermore, we perturb loss functions by injecting noise into the polynomial coefficients of the expansion of the objective function to protect the privacy of data labels. Theoretical analysis and experiments have shown that the proposed AFRRS mechanism can not only provide strong privacy preservation for the deep learning model, but also maintain the good utility of the model under a given moderate privacy budget compared with existing methods.

Index Terms—Deep learning, differential privacy, privacy leakage, feature relevance region segmentation

1 Introduction

Recent progress of deep learning techniques based on neural networks has brought new development opportunities to the field of artificial intelligence. Due to its powerful generalization ability and efficient information processing efficiency, deep learning has gradually developed into a key technology in many domains, including computer vision, natural language processing, image processing, speech recognition, autonomous vehicles, and other fields. It can fully mine the valuable information in the data by constructing a multi-layer neural network to meet the needs of different purposes.

However, the vigorous development of deep learning also brings unprecedented challenges for the security and privacy of data. Firstly, the training of the deep learning model is based on massive amounts of representative datasets, which usually contain a large amount of private information of individuals, such as medical information of patients, consumption record of users, personal voiceprint information, etc. Once they are leaked, irreversible losses will be caused [1]. Secondly, deep neural networks have a large number of hidden layers, which have the ability to encode some details of training data into model parameters [2], [3]. Existing studies have shown that attackers can extract individual private information in the training data from the neural network through member inference attacks [4], [5]. Transfer learning is regarded as the next driver to promote the development of machine learning. It is also widely used in deep learning. It can save the time of training the second model from the starting and improve its final performance. Transfer learning significantly promotes the pre-trained models to be shared, and a large number of pre-trained models have been publicly available, such as LeNet and ResNet. Therefore, the adversaries can undoubtedly surmise the confidential information of individuals in the training datasets by using the publicly available model parameters. In these cases, it is essential to implement deep learning-based privacy-preserving models to protect sensitive information from being obtained by adversaries.

How to ensure data privacy while using sensitive data has attracted people's attention. The commonly used privacy-preserving methods include anonymity, Secure Multi-Party Computing (SMPC), and Homomorphic Encryption (HE). However, anonymity cannot effectively protect attribute information and resist homogeneity attacks and background knowledge attacks. The computation and communication costs of SMPC are too high, and the implementation of Homomorphic Encryption is complex. Differential Privacy (DP) is applied to deep learning to realize privacy protection because of its advantages of simple implementation and quantifiable privacy protection level. It can be achieved simply by injecting noise into model parameters. The amount of noise injected is controlled by the

Manuscript received June 7, 2022; revised December 28, 2022. This work was supported by NSFC under Grants 61572170, Natural Science Foundation of Hebei Province under Grant F2021205004, Science and Technology Foundation Project of Hebei Normal University under Grant No. L2021K06, Science Foundation of Returned Overseas of Hebei Province Under Grant C2020342, Key Science Foundation of Hebei Education Department under Grant ZD2021062. (Corresponding authors: Zhiyuan Tan; Changguang Wang.)

- *F. Wang, M. Xie, Q. Li and C. Wang are with Key Laboratory of Network and Information Security of Hebei Province, College of Computer and Cyber Security, Hebei Normal University, Shijiazhuang, China (e-mail: fw_wang@hebtu.edu.cn; xmy-123@stu.hebtu.edu.cn; qingruli@hebtu.edu.cn; wangcg@hebtu.edu.cn).*
- *Z. Tan is with the School of Computing, Edinburgh Napier University, Edinburgh EH10 5DT, U.K. (e-mail: z.tan@napier.ac.uk).*

privacy budget ϵ. In addition, privacy budget can also reflect the level of privacy protection. However, the introduction of noise unavoidably decreases the accuracy of models. Thus, how to balance privacy and model utility becomes the key to differential privacy applications.

To maintain excellent model utility while preserving privacy, we propose an adaptive feature relevance region segmentation (AFRRS) mechanism for obtaining deep learning model with differential privacy. Firstly, input features are divided into regions with different levels of relevance based on relevance analysis. Less noise is injected into regions with stronger relevant features, and vice versa. Secondly, the loss function is transformed into a polynomial and different noise is injected into its coefficients.

Our contributions in the study are listed as follows:

(1) A novel differential privacy deep learning method called AFRRS mechanism is proposed. It implements noise injection according to the characteristics of the input features, which reduces the noise influence on the model's accuracy. Furthermore, we perturb the loss function to enhance the privacy protection effect.

(2) The process of injecting noise can be used as a preprocessing process, which is independent of model training. Therefore, the privacy budget is not accumulated in the model training phase. The AFRRS balances the privacy with practicability of the model.

(3) Strict mathematical analysis and the experimental results prove the effectiveness of our proposed method.

The rest of this study is structured as follows. A concise overview of privacy preservation in deep learning is given in Section 2. Section 3 introduces the theoretical basics about differential privacy. Section 4 gives our work in detail. Experimental results are analyzed and discussed in Section 5. Some conclusions are given in Section 6.

2 Related Work

The disclosure of private data and sensitive information might cause huge economic losses to individuals and enterprises, and even threaten national cybersecurity. Privacy preservation technology can protect data privacy without affecting the normal use of data. It mainly includes secure multi-party computing, homomorphic encryption, federated learning, differential privacy, etc. During the training process, deep leaning algorithms can be threaten by member inference attacks, and individual private information is extracted. Therefore, we focus only on differential privacy in deep learning.

Differential privacy has a set of precise mathematical theory. DP guarantees that it is almost impossible for adversaries to distinguish two adjacent datasets. Recently, DP has been successively used in deep learning to preserve data privacy. The differential privacy SGD algorithm is a general method. It injects the same amount of Gaussian noise into original gradient information in the training processing [6]. Still, the injection of the same amount of noise seriously affects the model's performance. Therefore, it is necessary to design the noise scheme carefully to reduce the influence of noise on the accuracy of the model. An improvement method is to gradually reduce the noise scale according to the iterations of the model. Yu *et al.* [7] propose a set of methods for privacy budget allocation, which change the noise scale in terms of the convergence of the model and effectively improve the model accuracy. Du *et al.* [8] introduce a sensitivity decay method to lessen the variance of noise injected into gradients in each iteration. Different from the method of reduction noise scale, Gong *et al.* [9] adaptively added noise to the gradient of neurons based on relevance analysis. In addition, gradient clipping is a necessary step to implement differential privacy. However, unreasonable clipping threshold reduces model accuracy. Therefore, some scholars try to replace the gradient clipping step with other schemes. Papernot *et al.* [10] replace ReLU activations with tempered sigmoids. Tempered sigmoids activations (TSA) control the gradient norm to avoid introducing too much noise. Stevens *et al.* [11] propose the backpropagation clipping (BC) algorithm to replace the gradient clipping operation, which limits the sensitivity of the gradient by clipping the gradient clips each trainable layer's inputs and its upstream gradients. To alleviate clipping bias, Liu *et al.* [12] propose a differentially private learning with grouped gradient clipping (DPL-GGC) method. The gradients are divided into different groups and each group is clipped separately. Xia *et al.* [13] propose Differentially Private Per-Sample Adaptive Clipping (DP-PSAC) algorithm. It coupled the clipping threshold with the learning rate to avoid tuning the clipping threshold. The sample gradients of different magnitudes remain in the same order as the original gradient size after being clipped. Bu *et al.* [14] propose the automatic clipping (AUTO clipping) strategy, which replaces the traditional gradient clipping with normalization. Wu *et al.* [15] propose an Adaptive Differentially Private Stochastic Gradient Descent (ADPSGD) algorithm, which adjusts the random noise added to the gradient by adaptive step size. Combining private learning with architectural search, Cheng *et al.* [16] propose the DPNASNet model, which achieves a state-of-the-art privacy/utility trade-off.

The researches mentioned earlier focus on gradient perturbation to preserve privacy. These algorithms consume the total privacy budge with each iteration. To reduce unnecessary privacy budget loss, they need precise privacy budget metrics method, such as Moment Accountant (MA), Rényi DP (RDP) [17], Guassian DP (GDP) [18]. Therefore, some studies implement input perturbation or loss function perturbation, which avoids the accumulation of privacy loss in the training iterations. For example, the dPA method perturbs the loss function to enforce differential privacy, and minimizes the perturbed loss function to optimize the model [19]. The PrivR framework distinguishes strongly relevant features from weakly relevant features by a preset threshold. It then allocates more privacy budget to the coefficients of the objective function [20]. Furthermore, the AdLM algorithm adds Laplace noise for perturbing the input features and loss function [21]. The noise injection process of the algorithm is independent of the model training and can be considered as the pre-training step completed before model training.

Different from existing differential privacy methods, the proposed AFRRS allocates different noises according to the different relevance of the input features. Furthermore, noise is injected into the coefficients of the loss function. The key advantage of the AFRRS is a relatively smaller privacy bucket (PB) resulting in a better model accuracy.

3 Preliminaries

We firstly provide the definition of differential privacy (DP), then introduce its some important properties, and finally review the basic process of relevance decomposition.

3.1 Differential Privacy

Definition 1. *((ϵ, δ) - DP [22]). Let $\epsilon, \delta \geq 0$ and a randomized algorithm M is (ϵ, δ) - DP if for any two adjacent databases D and $D^{'}$ differing at most one element, and all $S \subseteq Range(M)$, it holds that*

$$Pr\left(M\left(D\right)=S\right) \leq e^{\epsilon} Pr\left(M\left(D^{'}\right)=S\right). \quad (1)$$

Property 1. *(Post-Processing [23]). A randomized algorithm $M: D \rightarrow R$ is (ϵ, δ) - DP. Let $A: R \rightarrow R^{'}$ be an arbitrary randomized mapping. Then $A \circ M: D \rightarrow R^{'}$ satisfies (ϵ, δ) - DP.*

Property 2. *(Composition Theorem [23]). For $i \in [k]$, the algorithm $M_i: D \rightarrow R_i$ is (ϵ_i, δ_i) - DP. Then if $M_{[k]}: D \rightarrow \Pi_{i=1}^{k} R_i$ is defined to be $M_{[k]}(x) = (M_1(x), M_2(x), ..., M_k(x))$, then $M_{[k]}$ is $\left(\Sigma_{i=1}^{k}\epsilon_i, \Sigma_{i=1}^{k}\delta_i\right)$ - DP.*

Definition 2. *(Sensitivity [23]). For two adjacent datasets D and $D^{'}$, the L_1 sensitivity of a query function $f: D \rightarrow R^d$ is defined as*

$$\Delta_f = \max_{D,D^{'}} \left\| f(D) - f\left(D^{'}\right)\right\|_1. \quad (2)$$

The Laplace mechanism is a fundament of preserving ϵ - DP of any function f by adding Laplace noise to the output. It is defined as follows.

Definition 3. *(Laplace Mechanism [18]). Given any function $f: D \rightarrow R^d$, the Laplace mechanism is defined as:*

$$M(D) = f(D) + Y, \quad (3)$$

where $Y \sim \mathrm{Lap}(\Delta_f/\epsilon)$.

3.2 Relevance Decomposition

Layer-wise Relevance Propagation (LRP) is usually adopted to understand or interpret the relevance of a single-pixel to the classification decision in image classifications [24]. It computes the contribution of each input feature by layer-wise pixel decomposition from model output to input feature. In the backward propagation of the relevance, the model output $F_{x_i}(\omega)$ is first decomposed as the total relevance. In this paper, we calculate the relevance of input features using a pre-trained model. The process mainly consists of two steps:

Step 1. Relevance decomposition. The relevance of upper-layer neurons is decomposed based on the following formula:

$$R_{q\leftarrow p}^{(l-1,l)}\left(x_i\right) = \begin{cases} \frac{z_{qp}}{z_p+\beta} R_p^{(l)}\left(x_i\right), & z_q \geq 0, \\ \frac{z_{qp}}{z_p-\beta} R_p^{(l)}\left(x_i\right), & z_q < 0. \end{cases} \quad (4)$$

The predefined parameter $\beta \geq 0$ is aimed to avoid boundlessness of the relevance $R_{q\leftarrow p}^{(l-1,l)}(x_i)$ when z_p is close to 0. The affine transformation z_p of the neuron p is

$$z_{qp} = a_q \omega_{qp}, \quad (5)$$

$$z_p = \sum_q z_{qp} + b_p. \quad (6)$$

Step 2. Relevance propagation. During the backward propagation, the relevance of a lower layer neuron is denoted as the sum of the relevance of upper layer neurons connected to it:

$$R_q^{(l-1)}\left(x_i\right) = \sum_{p\in h_l} R_{q\leftarrow p}^{(l-1,l)}\left(x_i\right). \quad (7)$$

Finally, the relevance between every hidden neuron and input feature is obtained. Therefore, Eq. (8) holds:

$$F_{x_i}(\omega) = \sum_{p\in h_l} R_p^{(l)}(x_i) = ... = \sum_{x_{ij}\in s_i} R_{x_{ij}}. \quad (8)$$

4 Method Description

The section presents the details of our approach. We propose an adaptive noise injection mechanism based on feature relevance region segmentation. The proposed AFRRS mechanism injects Laplace noise into the model from three position: input-output feature relevance, input features and the objective function. It is shown in Fig. 1.

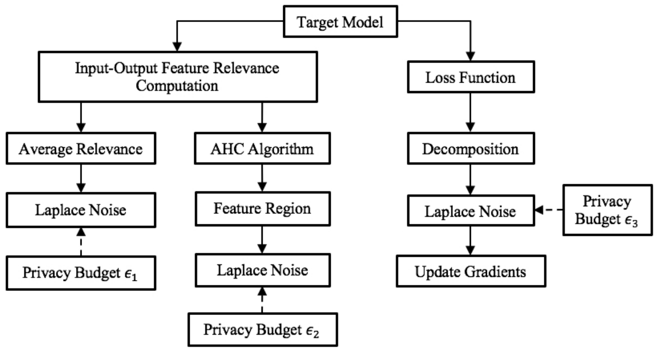


Fig. 1. Diagram of adding noise to the model.

4.1 Input-Output Feature Relevance Computation

Before adding noise, we first analyze the relevance of input features based on LRP algorithm. We can understand the role of each input feature in model decision-making by relevance analysis. Relevance analysis provides a basis for the subsequent feature relevance region segmentation.

Given an model output $F_{x_i}(\omega)$, the relevance of each input feature to the model output is calculated by LRP algorithm. It is worth noting that this step is completed in a pre-training model on the dataset D. Then, we can compute the average relevance of all the input features, which is shown as follows:

$$R_j(D) = \frac{1}{|D|} \sum_{x_i\in D} R_{x_{ij}}\left(x_i\right), \quad (9)$$

where $R_{x_{ij}}(x_i)$ is the relevance of the input feature x_{ij} to the model output $F_{x_i}(\omega)$. To ensure $R_j(D) \in [0,1]$, every $R_j(D)$ is standardized as $\frac{R_j(D)-\xi}{\tau-\xi}$, where τ and ξ are the maximum and minimum of $R_1(D), R_2(D), ..., R_d(D)$.

For $\forall j \in [1,d]$, we preserve differential privacy when computing the relevance $R_j(D)$ by injecting Laplace noise into it. $R_j^*(D)$ is defined as Equation (10):

$$R_j^*(D) = R_j(D) + \mathrm{Lap}\left(\Delta_R/\epsilon_1\right), \tag{10}$$

where ϵ_1 is the total budget injected into the relevance of all the input feature. The differential private relevance is ϵ_1 - differential privacy, whose correctness is based on Lemma 1 and Theorem 1, which are proven in Appendix A.1.

Lemma 1. *Given any two neighbouring databases D and D', the relevance of all input features on D and D' can be represented respectively as:*

$$\begin{aligned} &R(D) = \{R_j(D)\}_{j\in[1,d]}, \\ &\quad s.t. \quad R_j(D) = \tfrac{1}{|D|}\textstyle\sum_{x_i\in D} R_{x_{ij}}(x_i), \\ &R\left(D'\right) = \left\{R_j\left(D'\right)\right\}_{j\in[1,d]}, \\ &\quad s.t. \quad R_j\left(D'\right) = \tfrac{1}{|D'|}\textstyle\sum_{x'_i\in D'} R_{x'_{ij}}\left(x'_i\right). \end{aligned}$$

Then, Equation (11) holds:

$$\begin{aligned} \Delta_R =& \frac{1}{|D|}\sum_{j=1}^{d}\left\|\sum_{x_i\in D} R_{x_{ij}}(x_i) - \sum_{x'_i\in D'} R_{x'_{ij}}\left(x'_i\right)\right\|_1 \\ \le& \frac{2d}{|D|}. \end{aligned} \tag{11}$$

Theorem 1. *The computation of $R^*(D)$ satisfies ϵ_1 - differential privacy.*

4.2 Feature Relevance Region Segmentation

Through relevance analysis, we find that the input features are characterized by regional aggregation. That is, the relevance of adjacent input features is similar. The closer to the image center is, the stronger the input-output feature relevance is, and vice versa. Hence, we propose a region segmentation (RS) algorithm, which segments the input features into different relevance regions. In our method, we apply the Agglomerative Hierarchical Clustering (AHC) algorithm to segment the input features into regions with different relevance levels. Its detail process is shown in Algorithm 1.

In the RS algorithm, we need to construct an initial distance matrix. Each element in the matrix represents the distance between two clustering points, which is the absolute value of the relevance difference of two input features, that is

$$R[i,j] = |R_i - R_j|, \tag{12}$$

where $R[i,j]$ is the distance between input feature i and j.

Since the distance between the features i and j is the same with the distance between j and i, the distance matrix is symmetric, that is,

$$R[i,j] = R[j,i]. \tag{13}$$

Algorithm 1 Region Segmentation Algorithm

Input: Training dataset $D = \{x_1, ..., x_n\}$, minimum distance threshold γ, category distance calculation function $dis\_avg(C_i, C_j)$, the number of input features d.

Output: The number of divided regions s, the features contained in each region $C = \{C_1, C_2, ..., C_s\}$.

```
1: Calculate the differentially private relevance by LRP
   algorithm and Laplace mechanism, $\forall j \in [1,d]$:
   $R_j^*(D) = (1/|D|)\sum_{x_i\in D} R_{ij}(x_i) + \mathrm{Lap}(\Delta_R/\epsilon_1)$;
2: for $j \in [1,d]$ do
3:   Intitial every input feature as a category: $C_i = \{x_i\}$;
4: end for
5: for $i \in [1,d]$ do
6:   for $j \in [1,d]$ do
7:     Calculate the distance between two input features:
       $R[i,j] = |R_i - R_j|$;
8:     Construct a symmetric distance matrix:
       $R[i,j] = R[j,i]$;
9:   end for
10: end for
11: Set the current region number to $s = d$;
12: while $\min R < \gamma$ do
13:   Merge the two regions $C_i$ and $C_j$ with the smallest
      distance $C_i \leftarrow C_i \cup C_j$;
14:   Decrease the number of current regions by one $s \leftarrow$
      $s - 1$;
15:   Delete the row and column in the distance matrix;
16:   for $k \in [1,s]$ do
17:     Calculate the distance between the two categories
        $R[i,j] \leftarrow dis\_avg(C_i, C_j)$;
18:     $R[j,i] \leftarrow R[i,j]$.
19:   end for
20: end while
```

After the feature relevance region segmentation is completed, the input features with similar contributions are adaptively aggregated into the same region, and the region relevance can be calculated by averaging the relevance of all input features in a region.

4.3 Perturbation the input features

The relevance between each input feature and the model output is different. Therefore, adding the same amount of noise to all input features will affect the performance of the model. To address this problem, we propose an adaptive feature relevance region segmentation (AFRRS) mechanism, a dynamic privacy allocation method based on region segmentation. Its process is summarized in Algorithm 2. According to Algorithm 1, the input features are segmented into regions with different relevance. Then, we dynamically allocate privacy budgets for each region. The regional ratio is introduced as the basis for allocating privacy budgets for input features in each region, which is shown as follows:

$$\alpha_k = \frac{\left|\overline{R}_k\right|}{\sum_{k=1}^{s}\mu_k\left|\overline{R}_k\right|}, \tag{14}$$

where $\overline{R}_k$ is the differentially private relevance of the kth feature relevance region C_k, and $\left|\overline{R}_k\right|$ is the absolute value of $\overline{R}_k$. The proportion of the features contained in the kth feature relevance region C_k in all input features is defined as $\mu_k = |C_k|/d$, where $|C_k|$ represents the number of input features in the kth feature relevance region, and $\sum_{k=1}^{s} |C_k| = d$.

Based on the region relevance ratio, the privacy budget ϵ_k allocated to the input features in the kth feature relevance region is:

$$\epsilon_k = \alpha_k \times \epsilon_2. \tag{15}$$

The privacy budget ϵ_k is the overall privacy budget allocated to all input features. According to Eq (15), the stronger relevant the region is, the more the privacy budget is allocated, and the less noise is added. This result is significant. The features in the region with stronger relevance play a more critical role in the final decision. Therefore, less noise is added to reduce the impact on the final model output.

For $\forall x_i \in L$, the Laplace noise is adaptively added to the input feature belonging to a certain region according to the allocated privacy budget:

$$\overline{x}_{ij} = x_{ij} + (1/|L|) \operatorname{Lap}\left(\frac{\Delta_f}{\epsilon_k}\right), x_{ij} \in C_k. \tag{16}$$

The variance of the noise injected into the input feature is $\left(\frac{\Delta_f}{\epsilon_k}\right)$. More privacy budgets mean smaller noise scales, and vice versa. Thus, adaptive perturbation is injected into the input features according to the region relevance ratio. That is, less noise is injected into the input features with more vital relevance.

The first affine transformation layer h_0 of a model needs to access the original dataset D. To achieve the differentially private preservation of the original data information, we construct a DP layer $\overline{h}_0$, which is computed by adding Laplace noise to its affine transformation.

Given a training batch L, each neuron $p \in h_0$ is presented as:

$$f_{pL}^{(1)}(\omega^{(1)}) = \sum_{x_i \in L} (x_i \omega + b). \tag{17}$$

The perturbation to the bias b is as follows:

$$\overline{b} = b + \frac{1}{|L|} \operatorname{Lap}\left(\frac{\Delta_f}{\epsilon_2}\right), \tag{18}$$

where $|L|$ is batch size.

Given a training batch L, each input feature x_{ij} to the neuron $p \in h_0$ is perturbed based on the AFRRS mechanism. Therefore, the differentially private affine transformation layer $\overline{h}_0$ is presented as:

$$\overline{h}_{0L}(\omega^{(1)}) = \{\overline{f}_{pL}^{(1)}(\omega)\}_{p \in h_0},$$
$$s.t.\quad \overline{f}_{pL}^{(1)}(\omega) = \textstyle\sum_{x_i \in L} \left(\overline{x}_i \omega^T + \overline{b}\right).$$

The differentially private affine transformation layer satisfies ϵ_2 - differential privacy. Its correctness is based on Lemma 2 and Theorem 2, which are proven in Appendix A.1.

A neural network contains multiple hidden layers. we construct other hidden layers $h_1, h_2, ..., h_l$ on top of h_0. Since the hidden layers $h_1, h_2, ..., h_l$ are computed based on the DP layer h_0, they do not access the original data information, thus achieving differential privacy protection for the original data.

Algorithm 2 AFRRS Mechanism

```
Input: Training dataset $D = \{x_1, ..., x_n\}$, privacy budget $\epsilon_1$, $\epsilon_2$, $\epsilon_3$, learning rate $\eta$, the number of batches $T$, loss function $F(\omega)$, minimum distance threshold $\gamma$, category distance calculation function $dis\_avg(C_i, Cj)$, the batch size $|L|$, the number of input features $d$.
Output: Optimal parameter $\omega_T$.
    Segment input features by AHC algorithm:
    $C = \{C_1, C_2, ..., C_s\}$;
 2: for $k \in [1, s]$ do
        Calculate the average relevance of the $k$th feature region $C_k$:
        $\overline{R}_k = (1/|C_k|) \sum_{j \in C_k} R_j^*$;
 4:     Calculate the proportion of the features contained in the $k$th feature relevance region $C_k$ to the overall features:
        $\alpha_k = |\overline{R}_k| / \sum_{k=1}^{s} \mu_k |\overline{R}_k|$;
        Calculate the privacy budget allocated for the $k$th feature relevance region $C_k : \epsilon_k = \alpha_k \times \epsilon_2$;
 6: end for
    for $x_i \in D, j \in [1, d]$ do
 8:     $\overline{x}_{ij} = x_{ij} + (1/|L|) \operatorname{Lap}(\Delta_f/\epsilon_k)$,
        $\overline{b} = b + (1/|L|) \operatorname{Lap}(\Delta_f/\epsilon_2)$;
    end for
10: Construct hidden layers: $\{h_1, h_2, ..., h_l\}$;
    Calculate the perturbed loss function: $\overline{F}_L(\omega_t)$;
12: for each $k \in [T]$ do
        Take a stochastic batch $L$;
14:     Calculate the perturbed loss function with the privacy budget $\epsilon_3$: $\overline{F}_L(\omega_k)$;
        Calculate the gradients: $g_k(x_i) \leftarrow \nabla \overline{F}_L(\omega_k)$;
16:     Update gradients: $\omega_{k+1} = \omega_k - \eta_t (1/|L|) g_k(x_i)$.
    end for
```

Lemma 2. *Let D and $D^{'}$ denote two neighbouring datasets, and the model is trained in divided batches. Suppose L and $L^{'}$ are any two adjacent batches. Given the weight of the first affine transformation of the model $\omega^{(1)}$, let $h_{0L}(\omega^{(1)})$ and $h_{0L'}(\omega^{(1)})$ represent the output of the first affine transformation of the model on L and $L^{'}$, respectively, expressed as:*

$$h_{0L}(\omega^{(1)}) = \{f_{pL}^{(1)}(\omega)\}_{p \in h_0},$$
$$s.t.\quad f_{pL}^{(1)}(\omega) = \textstyle\sum_{x_i \in L} \left(x_i \omega^T + b\right),$$
$$h_{0L'}(\omega^{(1)}) = \{f_{pL'}^{(1)}(\omega)\}_{p \in h_0},$$
$$s.t.\quad f_{pL'}^{(1)}(\omega) = \textstyle\sum_{x_i^{'} \in L'} \left(x_i^{'} \omega^T + b\right).$$

Then, the following inequality holds:

$$\Delta_f = \sum_{p \in h_0} \sum_{k=1}^{s} \sum_{x_{ij}, x_{ij}^{'} \in C_k} \left\| \sum_{x_i \in L} x_{ij} - \sum_{x_i^{'} \in L'} x_{ij}^{'} \right\|_1 \leq 2 \sum_{p \in h_0} d.$$

Theorem 2. *The differentially private affine transformation of the model is ϵ_2 - differential privacy.*

4.4 Perturbation to The Loss Function

In the training process, we can obtain a deep learning model by minimizing the loss function. In previous steps, we realize adequate protection of sensitive information in the training dataset by adaptively injecting Laplacian noise into the input features. Furthermore, since the calculation of the loss function requires access to the data labels, it is essential to protect the given labels, which can be achieved by perturbing the loss function. Among loss functions, the cross-entropy error is widely used, which can be shown as

$$\begin{aligned} F_L(\omega) &= -\sum_{c=1}^{m}\sum_{x_i\in L}\left(y_{ic}\log y_{ic}^{'} + (1-y_{ic})\log\left(1-y_{ic}^{'}\right)\right) \\ &= -\sum_{c=1}^{m}\sum_{x_i\in L}\left(y_{ic}\log\left(1+e^{-h_{lc}(x_i)(W_{lc})^T}\right)\right) \\ &\quad -\sum_{c=1}^{m}\sum_{x_i\in L}\left((1-y_{ic})\log\left(1-e^{h_{lc}(x_i)(W_{lc})^T}\right)\right). \end{aligned} \tag{19}$$

Based on the literature [19], the loss function is transformed into a polynomial form

$$F_L^{'}(\omega) = \sum_{c=1}^{m}\sum_{x_i\in L}\sum_{R=0}^{2}\frac{f_c^{(R)}(0)}{R!}\left(h_{lc}\left(x_i\right)\left(W_{lc}\right)^T\right)^R, \tag{20}$$

where m denotes the number of the categories.

For $\forall c \in [1,m]$, we have $f_c(z) = y_{ic}\log\left(1+e^{-z}\right) + (1-y_{ic})\log\left(1+e^{z}\right)$. It is worth noting that when calculating the cross-entropy loss, the output range of the last hidden layer of the model is $[0,1]$.

The loss function is expanded into a polynomial by applying the function mechanism [25]. The ϵ_3 - DP is realized by adding Laplace noise to the polynomial coefficients. Let $\lambda_c^{(R)}\left(x_i\right)$ stand for $f_c^{(R)}(0)/R!$, where $R \in [0,2]$; then we have $\{\lambda_c^{(0)}\left(x_i\right) = \log 2, \lambda_c^{(1)}\left(x_i\right) = 1/2 - y_{ic}, \lambda_c^{(2)}\left(x_i\right) = 1/8\}$. $\phi_c\left(x_i\right) = \{\phi_c^{(0)}\left(x_i\right), \phi_c^{(1)}\left(x_i\right), \phi_c^{(2)}\left(x_i\right)\}$ stands for the coefficients, which are composed of combinations between $\lambda_c^{(R)}\left(x_i\right)$ and the Rth power of $h_{lc}\left(x_i\right)$. The perturbed coefficients are as follows:

$$\bar{\phi}_c^{(R)}\left(x_i\right) = \phi_c^{(R)}\left(x_i\right) + \frac{1}{|L|}\operatorname{Lap}\left(\Delta_F/\epsilon_3\right). \tag{21}$$

The perturbed loss function satisfies ϵ_3 - differential privacy. Its correctness is based on Lemma 3 and Theorem 3, which are proven in Appendix A.1.

Lemma 3. *Let L and $L^{'}$ represent the neighbouring batches. $F_L^{'}(\omega)$ and $F_{L^{'}}^{'}(\omega)$ are the polynomial approximation of $F_L(\omega)$ corresponding to L and $L^{'}$, then we have*

$$\begin{aligned} \Delta_F &= \sum_{c=1}^{m}\sum_{R=0}^{2}\left\|\sum_{x_i\in L}\phi_c^{(R)}\left(x_i\right) - \sum_{x_i^{'}\in L^{'}}\phi_c^{(R)}\left(x_i^{'}\right)\right\|_1 \\ &\le m\left(|h_l| + \frac{1}{4}|h_l|^2\right). \end{aligned} \tag{22}$$

Theorem 3. *The perturbed loss function $\overline{F}_L(\omega)$ is ϵ_3 - DP.*

Based on the previously mentioned Lemmas and Theorems, we can obtain Theorem 4.

Theorem 4. *Algorithm 2 satisfies $\epsilon_1 + \epsilon_2 + \epsilon_3$ - DP.*

Proof. First of all, it can be known from Theorem 1 that the differentially private relevance of the input features satisfies ϵ_1 - DP. From Theorem 2, it can be seen that the differentially private affine transformation layer of the model is ϵ_2 - DP. Because DP is not affected by post-processing, the computation of $h_1, h_2, ..., h_l$ is ϵ_2 - DP. According to Theorem 3, the loss function satisfies ϵ_3 - DP. In the processing of gradient calculation, access to the original data is not required, and gradient descent still satisfies ϵ_3 - DP based on Property 1. Furthermore, the procedures mentioned aforementioned access to the same training dataset at each training epoch. Therefore, based on Property 2, Algorithm 1 satisfies $\epsilon_1 + \epsilon_2 + \epsilon_3$ - DP. □

5 Experiments

5.1 Experimental Setting

1) *Dataset*. We chose two datasets to validate the proposed AFRRS in this paper.

- **MNIST** [26]. It contains 70,000 handwritten digital images of 10 categories, including 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. There are 60000 images for training, 10000 images for testing. The size of each image is 28×28.
- **CIFAR10** [27]. It contains 60000 color images of 10 categories, including airplanes, cars, birds, cats, etc. There are 50000 images for training, 10000 images for testing. Every image has 32×32 pixels and three channels (RGB).

2) *Experimental settings*. Experiments are performed with PyTorch framework with NVIDIA GeForce RTX 2080 Ti. The network architectures of MNIST dataset and CIFAR10 dataset are shown in Table 1 and Table 2 respectively. All experiments run 100 epochs. The experimental results for our method are averaged over 10 runs.

TABLE 1
The model architecture for MNIST.

Layers	Parameters
Convolution	16 filters of 8×8, strides 2
Max-Pooling	2×2
Convolution	32 filters of 4×4, strides 2
Max-Pooling	2×2
Fully connected	32 units
Softmax	10 units

TABLE 2
The model architecture for MNIST.

Layers	Parameters
Convolution	32 filters of 3×3, strides 1
Avg-Pooling	2×2, strides 1
Convolution	64 filters of 3×3, strides 1
Avg-Pooling	2×2, strides 1
Convolution	128 filters of 4×4, strides 1
Avg-Pooling	2×2, strides 1
Convolution	256 filters of 4×4, strides 1
Fully connected	32 units
Softmax	10 units

5.2 Results of Feature Relevance Region Segmentation

The differentially private relevance of each input feature is computed based on the LPR algorithm and Laplace mechanism. On the basis of the differentially private relevance, the RS algorithm is applied to segment the input features. The heatmaps of MNIST and CIFAR10 after segmentation are shown in Fig. 2 and Fig. 3 respectively.

The coordinates of the heat map indicate the image size. For the mnist dataset, the pre-training model is LeNet-5 when calculating the relevance. Therefore, the image size is adjusted to 32×32. For simplicity, we only use the central 28×28 part of the correlation matrix when subsequently adding noise to the input features.

Relevance analysis requires access to the raw data information. To prevent data privacy leakage, we compute the differential privacy relevance and then perform region segmentation of the input features based on it, as well as add noise. Fig. 2 and Fig. 3 show the region segmentation results. They are drawn based on the differential privacy relevance of input features. The results illustrate that the relevance of features has the characteristic of regional aggregation. The input features of weak contribution to the model output are mainly concentrated at the margin of an image, and the input features of strong contribution to the model are primarily concentrated in the center of an image. This is because the main content of an image is focused on the central region, and the edge part is mainly the background of the image, which plays a relatively minor role in the final decision.Input features with the same color belong to the same feature relevance region. The relevance of the region with the darker color is stronger, which has a more significant influence on the final decision of the model. Therefore, in the subsequent noise addition operation, less noise to the feature regions is injected into the regions with stronger relevance to reduce the influence of noise on the final decision. More noise is injected into the regions with weaker relevance to provide a privacy guarantee.

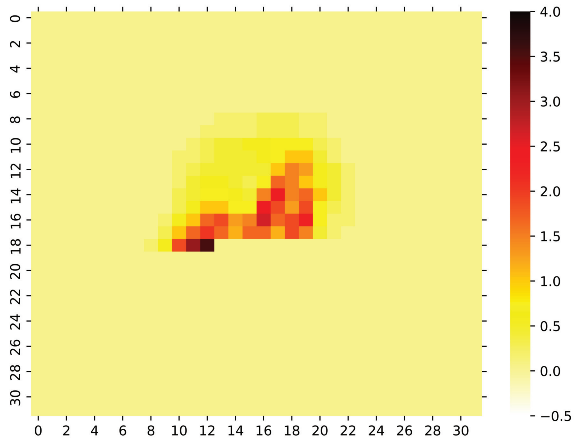


Fig. 2. The heatmap of differentially private relevance of the input features on MNIST.

5.3 Effectiveness Evaluation of Different Algorithms

To verify the effectiveness of the AFRRS mechanism, the experiments were designed to compare TSA [10], BC

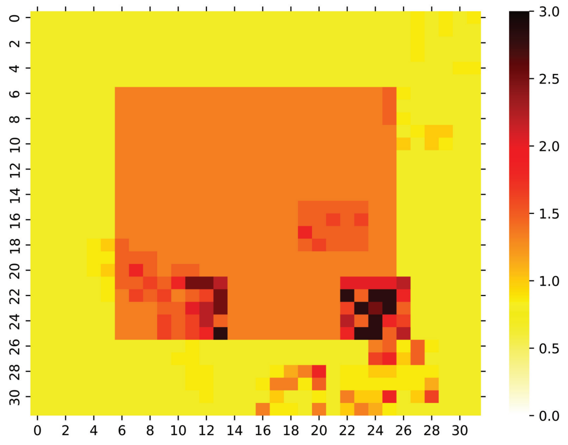


Fig. 3. The heatmap of differentially private relevance of the input features on CIFAR10.

[11], DPL-GGC [12], DP-PSAC [13], AUTO clipping [14], ADPSGD [15], DPNASNet [16], and the deep learning model without differential privacy protection (No_DP). The results are shown in Table 3 and Table 4. The experiments show that the algorithm proposed in this paper can improve the model accuracy with smaller privacy budget. In addition, the compared algorithms add noise to the gradient, which cause the noise to gradually accumulates during the model iteration. Model accuracy is affected by the privacy budget metric method. If the privacy budget used in each iteration is not calculated accurately, unnecessary privacy budget will be consumed. However, as a pre-processing step, the noise addition step of the AFRRS mechanism can be completed before the model is trained. It can achieve an accurate measure of the consumed privacy budget, and the noise is not accumulated during the iterations. Therefore, higher model accuracy can be obtained by running more epochs. The AFRRS mechanism smooths the gap with the non-privacy models in the privacy-preserving deep learning model.

TABLE 3
Comparison result between the AFRRS mechanism and other algorithm on MNIST.

Algorithm	Accuracy (%)	PB (ϵ)
No_DP	99.00	-
TSA [10]	98.10	2.93
DPL-GGC [12]	98.23	2.85
DP-PSAC [13]	98.24	3.00
AUTO clipping [14]	98.15	3.00
DPNASNet [16]	98.57	3.00
AFRRS	98.80	0.60

We evaluated the model accuracy of the AFRRS mechanism at different privacy levels. The experimental results are shown in Fig. 4 and Fig. 5. The results show that for these two datasets, the model accuracy of the AFRRS mechanism improves as the privacy budget increases. The cause lies in the fact the more privacy budget means the weaker privacy guarantee, and less noise is injected into the model.

TABLE 4
Comparison result between the AFRRS mechanism and other algorithm on CIFAR10.

Algorithm	Accuracy (%)	PB (ϵ)
No_DP	76.90	-
TSA [10]	66.20	7.53
BC [11]	74.00	3.64
DPL-GGC [12]	67.11	3.19
ADPSGD [15]	69.63	6.40
DPNASNet [16]	68.33	3.00
AFRRS	76.00	2.00

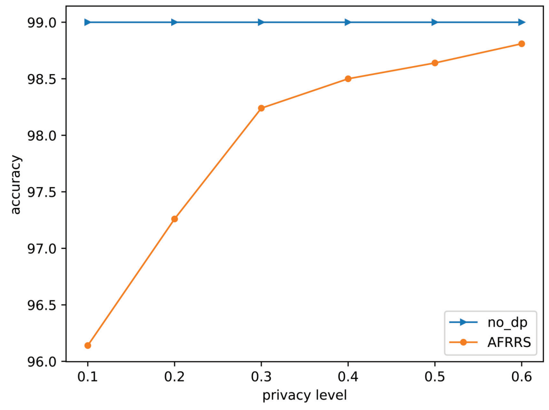


Fig. 4. The accuracy of the AFRRS mechanism for different privacy levels for MNIST.

6 Conclusions

In this paper, an adaptive noise addition mechanism based on feature relevance region segmentation is proposed. The method first obtains the differential privacy relevance. Based on the relevance analysis, the input features are segmented into regions with different relevance levels. Less noise is injected into the input features in the regions with stronger relevance, and vice versa. It retains the local characteristics of the input features and reduces the impact on the important feature region. Furthermore, the loss function is transformed into the form of a polynomial, and then the loss function is perturbed by injecting noise into its coefficients. In our method, the process of injecting noise is used as a preprocessing process, which can be completed before the model training, and the noise does not accumulate with the training of the model. The privacy loss can also be precisely calculated. The effectiveness of our proposed algorithm is proved theoretically. Experimental results demonstrate that the proposed method heightens the model's accuracy under a reasonable privacy budget. And our method can provide a privacy guarantee for different neural networks.

The limitation of this study is that the performance of AFRRS mechanism in various attack scenarios has not been evaluated. In the future, we will explore the defense performance of AFRRS mechanism in real attack scenarios such as member inference and attribute inference.

Appendix A

A.1 Proof of Lemma 1

Proof. Assume that the D and $D^{'}$ are the neighbouring databases differing only on the last element. Let x_n and $x_n^{'}$ denote the final element in D and $D^{'}$, respectively. We derive

$$\begin{aligned}\Delta_R =&\frac{1}{|D|}\sum_{j=1}^{d}\left\|\sum_{x_i\in D}R_{x_{ij}}(x_i)-\sum_{x_i^{'}\in D^{'}}R_{x_{ij}^{'}}\left(x_i^{'}\right)\right\|_1\\ =&\frac{1}{|D|}\sum_{j=1}^{d}\left\|R_{x_{nj}}(x_n)-R_{x_{nj}^{'}}\left(x_n^{'}\right)\right\|_1\\ \le&\frac{2}{|D|}max_{x_i\in D}\sum_{j=1}^{d}\left\|R_{x_{ij}}(x_i)\right\|_1.\end{aligned}$$

Since $\forall x_i\in D$, $j\in[1,d]$, $R_{x_{ij}}\in[0,1]$, the following inequality holds $\Delta_R\le\frac{2d}{|D|}$. □

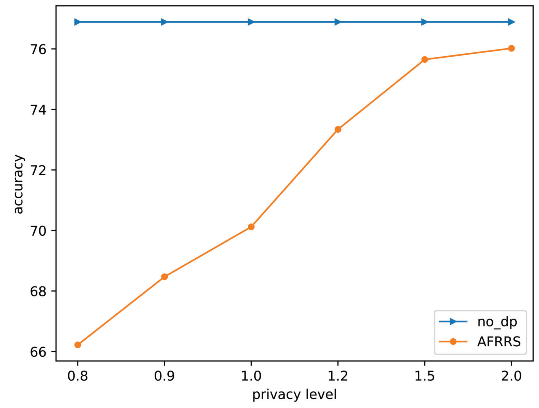


Fig. 5. The accuracy of the AFRRS mechanism for different privacy levels for CIFAR10.

A.2 Proof of Theorem 1

Proof. Let $R_j^*(x_i)$ be perturbed relevance, that is

$$R_j^*(x_i)=R_j(D)+\text{Lap}\left(\Delta_R/\epsilon_1\right).$$

We have that

$$\begin{aligned}\frac{Pr(R^*(D))}{Pr(R^*(D'))}=&\frac{\Pi_{j=1}^{d}\exp\left(\frac{\epsilon_1}{\Delta_R}\left\|R_j(D)-R_j^*)\right\|_1\right)}{\Pi_{j=1}^{d}\exp\left(\frac{\epsilon_1}{\Delta_R}\left\|R_j(D')-R_j^*)\right\|_1\right)}\\ \le&\Pi_{j=1}^{d}\exp\left(\frac{\epsilon_1}{\Delta_R}\left\|R_j(D)-R_j(D^{'}))\right\|_1\right)\\ \le&\Pi_{j=1}^{d}\exp\left(\frac{\epsilon_1}{|D|\Delta_R}\left\|R_j(x_n)-R_j(x_n^{'})\right\|_1\right)\\ \le&\Pi_{j=1}^{d}\exp\left(\frac{\epsilon_1}{|D|\Delta_R}2\max_{x_n\in D}\left\|R_j(x_n)\right\|_1\right)\\ \le&\exp\left(\frac{2\epsilon_1}{|D|\Delta_R}\sum_{j=1}^{d}\max_{x_n\in D}\left\|R_j(x_n)\right\|_1\right)\\ =&\exp(\epsilon_1).\end{aligned}$$

Therefore, the computation of $R^*(D)$ is ϵ_1 - differential privacy. This completes the proof. □

A.3 Proof of Lemma 2

Proof. Suppose that L and $L^{'}$ are the adjacent batches differing only on the last element. x_n and $x_n^{'}$ denote the final element in L and $L^{'}$, respectively, we obtain

$$\begin{aligned}\Delta_f =&\sum_{p\in h_0}\sum_{k=1}^{s}\sum_{x_{ij},x_{ij}^{'}\in C_k}\left\|\sum_{x_i\in L}x_{ij}-\sum_{x_i^{'}\in L^{'}}x_{ij}^{'}\right\|_1\\ =&\sum_{p\in h_0}\sum_{k=1}^{s}\sum_{x_{nj},x_{nj}^{'}\in C_k}\left\|x_{nj}-x_{nj}^{'}\right\|_1\\ \le&\sum_{p\in h_0}\sum_{k=1}^{s}\sum_{x_{nj},x_{nj}^{'}\in C_k}\left(\left\|x_{nj}\right\|_1+\left\|x_{nj}^{'}\right\|_1\right)\\ \le&2\sum_{p\in h_0}\sum_{k=1}^{s}\sum_{x_{nj}\in C_k}\left(\max_{x_n\in L}\left\|x_{nj}\right\|_1\right).\end{aligned}$$

Without loss of generality, we suppose $\sqrt{\sum_{j=1}^{d}x_{ij}^2}$ is supposed less than or equal to 1, where $x_{ij}\ge 0$. It can be achieved by normalization techniques. Since for $\forall x_{ij}: x_{ij}\in[0,1]$, we have

$$\Delta_f\le 2\sum_{p\in h_0}\sum_{k=1}^{s}|C_k|\le 2\sum_{p\in h_0}d.$$

This completes the proof. □

A.4 Proof of Theorem 2

Proof. $\forall p\in h_0$, the perturbed affine transformation is expressed as:

$$\begin{aligned}\overline{f}_L^{(1)}(\omega)=&\sum_{k=1}^{s}\sum_{x_{ij}\in C_k}\left[\sum_{x_i\in L}\left(x_{ij}+\frac{1}{|L|}\operatorname{Lap}\left(\frac{\Delta_f}{\epsilon_k}\right)\right)\omega^T\right]\\ &+\sum_{x_i\in L}\left(b+\frac{1}{|L|}\operatorname{Lap}\left(\frac{\Delta_f}{\epsilon_2}\right)\right).\end{aligned}$$

Let all b be the input features in the 0th feature relevance region and ω_b be the weight parameter of b. The weight parameter of all input features is $\omega=\omega_b\cup\omega$. We have

$$\begin{aligned}\overline{f}_L^{(1)}(\omega)=&\sum_{k=0}^{s}\sum_{x_{ij}\in C_k}\left[\sum_{x_i\in L}\left(x_{ij}+\frac{1}{|L|}\operatorname{Lap}\left(\frac{\Delta_f}{\epsilon_k}\right)\right)\omega^T\right]\\ =&\sum_{k=0}^{s}\sum_{x_{ij}\in C_k}\varphi_j\omega^T,\end{aligned}$$

where

$$\varphi_j=\sum_{x_i\in L}\left(x_{ij}+\frac{1}{|L|}\operatorname{Lap}\left(\frac{\Delta_f}{\epsilon_k}\right)\right).$$

Since all neurons of the first affine transformation layer have the perturbation as mentioned above, the following formula holds:

$$\begin{aligned}&\frac{Pr\left(\overline{h}_{0L}\omega^{(1)}\right)}{Pr\left(\overline{h}_{0L^{'}}\omega^{(1)}\right)}\\ =&\frac{\Pi_{p\in h_0}\Pi_{k=0}^{s}\Pi_{x_{ij}\in C_k}\exp\left(\frac{\epsilon_k}{\Delta_f}\left\|\sum_{x_i\in L}x_{ij}-\varphi_j\right\|_1\right)}{\Pi_{p\in h_0}\Pi_{k=0}^{s}\Pi_{x_{ij}^{'}\in C_k}\exp\left(\frac{\epsilon_k}{\Delta_f}\left\|\sum_{x_i^{'}\in L}x_{ij}^{'}-\varphi_j\right\|_1\right)}\\ \le&\Pi_{p\in h_0}\Pi_{k=0}^{s}\Pi_{x_{ij},x_{ij}^{'}\in C_k}\exp\left(\frac{\epsilon_k}{\Delta_f}\left\|x_{nj}-x_{nj}^{'}\right\|_1\right)\\ \le&\Pi_{p\in h_0}\Pi_{k=0}^{s}\Pi_{x_{ij},x_{ij}^{'}\in C_k}\exp\left(\frac{\epsilon_k}{\Delta_f}2\max_{x_n\in L}\left\|x_{nj}\right\|_1\right)\\ \le&\Pi_{p\in h_0}\Pi_{k=0}^{s}\exp\left(\frac{2\epsilon_k|C_k|}{\Delta_f}\right)\\ \le&\Pi_{p\in h_0}\Pi_{k=0}^{s}\exp\left(\frac{2d}{\Delta_f}\frac{\mu_k\left|\overline{R}_k\right|}{\sum_{k=1}^{s}\mu_k\left|\overline{R}_k\right|}\epsilon_2\right)\\ =&\Pi_{p\in h_0}\exp\left(\frac{2d}{\Delta_f}\epsilon_2\right)\\ =&\exp\left(\frac{2\sum_{p\in h_0}d}{\Delta_f}\epsilon_2\right)\\ =&\exp\left(\epsilon_2\right).\end{aligned}$$

□

A.5 Proof of Lemma 3

Proof. Suppose that L and $L^{'}$ differ in only on the last element. The two elements are represented as x_n and $x_n^{'}$, respectively. We obtain

$$\begin{aligned}\Delta_F=&\sum_{c=1}^{m}\sum_{R=0}^{2}\left\|\sum_{x_i\in L}\phi_c^{(R)}(x_i)-\sum_{x_i^{'}\in L^{'}}\phi_c^{(R)}(x_i^{'})\right\|_1\\ =&\sum_{c=1}^{m}\sum_{R=0}^{2}\left\|\phi_c^{(R)}(x_n)-\phi_c^{(R)}(x_n^{'})\right\|_1.\end{aligned}$$

We know $\phi_c^{(0)}(x_n)=\lambda_c^{(0)}(x_n)=\log 2$, and $\phi_c^{(0)}(x_n^{'})=\lambda_c^{(0)}(x_n^{'})=\log 2$, thus $\phi_c^{(0)}(x_n)=\phi_c^{(0)}(x_n^{'})$. Therefore,

$$\begin{aligned}\Delta_F\le&\sum_{c=1}^{m}\sum_{R=1}^{2}\left(\left\|\phi_c^{(R)}(x_n)\right\|_1+\left\|\phi_c^{(R)}(x_n^{'})\right\|_1\right)\\ \le&2\max\sum_{c=1}^{m}\sum_{R=1}^{2}\left\|\phi_c^{(R)}(x_n)\right\|_1\\ \le&2\max\sum_{c=1}^{m}\left[\left(\frac{1}{2}-y_{nc}\right)\sum_{p\in h_l}h_l^p(x_n)\right]\\ &+2\max\sum_{c=1}^{m}\left[\frac{1}{8}\sum_{p,q\in h_l}h_l^p(x_n)h_l^q(x_n)\right]\\ \le&2\sum_{c=1}^{m}\left[\max\left(\left(\frac{1}{2}-y_{nc}\right)\sum_{p\in h_l}h_l^p(x_n)\right)\right]\\ &+2\sum_{c=1}^{m}\left[\max\left(\frac{1}{8}\sum_{p,q\in h_l}h_l^p(x_n)h_l^q(x_n)\right)\right]\\ \le&2\sum_{c=1}^{m}\left(\frac{1}{2}|h_l|+\frac{1}{8}|h_l|^2\right)\\ =&m\left(|h_l|+\frac{1}{4}|h_l|^2\right),\end{aligned}$$

where $h_l^p(x_i)$ is the input of pth neuron in h_l. □

A.6 Proof of Theorem 3

Proof. Suppose that L and $L^{'}$ differ in only on the last element. The two elements are represented as x_n and $x^{'}_n$, respectively. During back propagation, the perturbed loss function $\overline{F}_L(\omega_t)$ needs to be minimized, and the perturbation to the coefficient $\overline{\phi}_c^{(R)}$ denotes as

$$\overline{\phi}_c^{(R)} = \sum_{x_i \in L} \left(\phi_c^{(R)}(x_i) + \frac{1}{|L|} \operatorname{Lap}\left(\frac{\Delta_F}{\epsilon_3} \right) \right).$$

We have that

$$\frac{Pr\left(\overline{F}_L(\omega_t)\right)}{Pr\left(\overline{F}_{L'}(\omega_t)\right)}$$

$$= \frac{\Pi_{c=1}^{m}\Pi_{R=0}^{2} \exp\left(\frac{\epsilon_3}{\Delta_F} \left\| \sum_{x_i \in L} \phi_c^{(R)}(x_i) - \overline{\phi}_c^{(R)} \right\|_1 \right)}{\Pi_{c=1}^{m}\Pi_{R=0}^{2} \exp\left(\frac{\epsilon_3}{\Delta_F} \left\| \sum_{x_i' \in L'} \phi_c^{(R)}(x_i) - \overline{\phi}_c^{(R)} \right\|_1 \right)}$$

$$\leq \Pi_{c=1}^{m}\Pi_{R=0}^{2} \exp\left(\frac{\epsilon_3}{\Delta_F} \left\| \sum_{x_i \in L} \phi_c^{(R)}(x_i) - \sum_{{x_i}' \in L'} \phi_c^{(R)}(x_i^{'}) \right\|_1 \right)$$

$$\leq \Pi_{c=1}^{m}\Pi_{R=1}^{2} \exp\left(\frac{\epsilon_3}{\Delta_F} 2 \max_{x_n \in L} \left\| {\phi_c}^{(R)}(x_n) \right\|_1 \right)$$

$$\leq \exp\left(\frac{\epsilon_3}{\Delta_F} 2 \max_{x_n \in L} \sum_{c=1}^{m} \sum_{R=1}^{2} \left\| {\phi_c}^{(R)}(x_n) \right\|_1 \right)$$

$$\leq \exp\left(\epsilon_3 \frac{m\left(|h_l| + \frac{1}{4} |h_l|^2 \right)}{\Delta_F} \right)$$

$$= \exp(\epsilon_3).$$

Consequently, the perturbed loss function $\overline{F}_L(\omega)$ preserves ϵ_3 - differential privacy. □

References

[1] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, "Privacy risk in machine learning: Analyzing the connection to overfitting," *IEEE 31st computer security foundations symposium (CSF)*, pp. 268-282, 2018.

[2] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," *Communications of the ACM*, vol. 64, no. 3, pp. 107–115, 2017.

[3] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, and U. Erlingsson, "Extracting training data from large language models," *30th USENIX Security Symposium (USENIX Security 21)*, pp. 2633-2650, 2021.

[4] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proceedings of the IEEE Symposium on Security and Privacy*, 2017, pp. 3–18.

[5] B. Jayaraman, Ling. Wang, K. Knipmeyer, Q. Gu, and D. Evans, "Deep learning with differential privacy," *arXiv preprint*, arXiv: 2005.10881, 2020.

[6] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 23rd ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 308–318.

[7] L. Yu, L. Liu, C. Pu, M. E. Gursoy, and S. Truex, "Differentially private model publishing for deep learning," in *Proceedings of the 40th IEEE Symposium on Security and Privacy*, 2019, pp. 332–349.

[8] J. Du, S. Li, F. Mo, and S. Chen, "Dynamic differential-privacy preserving SGD," *arXiv preprint*, arXiv:2111.00173, 2021.

[9] M. Gong, K. Pan, Y. Xie, A. Qin, and Z. Tang, "Preserving differential privacy in deep neural networks with relevance-based adaptive noise imposition," *Neural Networks*, vol. 125, no. 5, pp. 131–141, 2020.

[10] N. Papernot, A. Thakurta1, S. Song, S. Chien, and Ú. Erlingsson, "Tempered sigmoid activations for deep learning with differential privacy," in *Proceeding of the AAAI Conference on Artificial Intelligence*, 2021, pp. 9312–9321.

[11] T. Stevens, I. C. Ngong, D. Darais, C. Hirsch, D. Slater, and J. P. Near, "Backpropagation clipping for deep learning with differential privacy," *arXiv preprint*, arXiv: 2202.05089, 2022.

[12] H. Liu, C. Li, B. Liu, P. Wang, S .Ge, and W .Wang, "Differentially Private Learning with Grouped Gradient Clipping,*ACM Multimedia Asia*, pp. 1-7, 2021.

[13] T. Xia, S. Shen, S. Yao, X. Fu, K. Xu, X. Xu, X. Fu and W. Wang, "Differentially private learning with per-sample adaptive clipping," *arXiv preprint*, arXiv: 2212.00328, 2022.

[14] Z. Bu, Y. Wang, S. Zha, and G. Karypis, "Automatic clipping: differentially private deep learning made easier and stronger," *arXiv preprint*, arXiv: 2206.07136, 2022.

[15] X. Wu, L. Wang, and I. Cristali, "Adaptive differentially private empirical risk minimization," *arXiv preprint*, arXiv:2110.07435, 2021.

[16] A. Cheng, J. Wang, X. Zhang, Q. Chen, and P. Wang, "Dpnas: Neural architecture search for deep learning with differential privacy," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022, pp. 6358-6366.

[17] I. Mironov, "Rényi differential privacy," *IEEE 30th computer security foundations symposium (CSF)*, 2017, pp. 263-275.

[18] J. Dong, A. Roth, J W Su, "Gaussian differential privacy," *arXiv preprint*, arXiv: 1905.02383, 2019.

[19] N. Phan, Y. Wang, X. Wu, and D. Dou, "Differential privacy preservation for deep auto-encoders: An application of human behavior prediction," in *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, 2016, pp. 1309–1316.

[20] M. Gong, K. Pan, and Y. Xie, "Differential privacy preservation in regression analysis based on relevance," *Knowledge-Based Systems*, vol. 173, no. 6, pp. 140–149, 2019.

[21] N. Phan, X. Wu, H. Hu, and D. Dou, "Adaptive Laplace mechanism: Differential privacy preservation in deep learning," in *Proceedings of the IEEE International Conference on Data Mining*, 2017, pp. 385–394.

[22] C. Dwork, "Differential privacy," in *Proceedings of the 33rd International Colloquium on Automata, Languages, and Programming*, 2006, pp. 1–12.

[23] C. Dwork, and A. Roth, "The algorithmic foundations of differential privacy," *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 3-4, pp. 211–407, 2014.

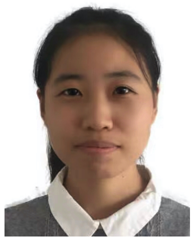
[24] S. Bach, A. Binder, G. Montavon, F. Klauschen, K. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifer decisions by layer-wise relance propagation," *PLoS One*, vol. 10, no. 7, pp. 1–46, 2015.

[25] J. Zhang, Z. Zhang X. Xiao, Y. Yang, and M. Winslett, "Functional mechanism: regression analysis under differential privacy," in *Proceedings of the VLDB Endowment*, 2012, pp. 1364–1375.

[26] The mnist dataset. *URL http://yann.lecun.com/exdb/mnist/*.

[27] The cifar10 dataset. *URL http://www.cs.toronto.edu/ kriz/cifar.html*.

Fangwei Wang received the B.S. degree from Hebei Normal University, Shijiazhuang, China, in 2000, the M.S. degree from Hebei University of Technology, Tianjin, China, in 2003, and Ph.D degree from College of Computer at Xidian University, Xi'an, China, in 2009. Currently, he is a professor in the College of Computers at Hebei Normal University, China. He has published more than 30 research papers and has presented more than 10 papers at conferences. He serves as a reviewer of Wireless Communications and Mobile Computing, Security and Communication Networks. His research area includes network security, privacy preservation, distributed computing.



Meiyun Xie received her B.S. degree in 2020 from College of mathematics, Hebei GEO University, Shijiazhuang, China. Currently she is currently pursuing the Master's degree in the college of Computer and Cyber Security of Hebei Normal University. Her research interests include privacy preservation, cloud security.

Zhiyuan Tan (Member, IEEE) received the B. Eng. degree in computer science and technology from Northeastern University, Shenyang, China, in 2005, the M. Eng. degree in software engineering from the Beijing University of Technology, Beijing, China, in 2008, and the Ph. D. degree in computer systems from the University of Technology Sydney, Ultimo, NSW, Australia, 2014. He was a Postdoctoral Researcher in cybersecurity with the University of Twente (UT), The Netherlands, from 2014 to 2016. He is currently a Lecturer with the School of Computing, Edinburgh Napier University (ENU), U.K. His current research interests include cybersecurity, machine learning, data analytics, virtualization, and cyber-physical systems. Dr. Tan is also an Academic Editor of the Security and Communication Networks and a Section Editor of the Ad Hoc and Sensor Wireless Networks Journal.

Qingru Li received the B.S. degree in department of physics from Hebei Normal University, Shiijiazhuang, China, in 1995, the M.S. degree in department of radio engineering from Lanzhou University, Lanzhou, China, 1998, and the Ph.D degree at Xidian university, Xi'an, China, in 2016. Currently, she is an associate in the College of Computers at Hebei Normal University, China. He has published more than 20 research papers. Her area of research is information security, privacy preservation.

Changguang Wang received the B.S. degree in department of physics from Hebei Normal University, Shiijiazhuang, China, in 1993, the M.S. degree in department of radio engineering from Sichuan University, Chengdu, China, 1996, and the Ph.D degree at Xidian university, Xi'an, China, in 2009. Currently, he is a professor in the College of Computers at Hebei Normal University, China. He has published more than 20 research papers. His research interests include wireless network security, IoV.