Differentially Private Deep Learning with Importance-based Adaptive Gradient Processing

Ping Li* LIPING26@MAIL2.SYSU.EDU.CN
School of Computer Science, South China Normal University

Mingwei Liang LIANGMINGWEI@M.SCNU.EDU.CN
School of Artificial Intelligence, South China Normal University

Zhao Jiang JIANGZHAO@M.SCNU.EDU.CN
School of Computer Science, South China Normal University

Jun Zhang ZHANGJUNIRIS@SZU.EDU.CN
College of Education, Shenzhen University

Editors: Vu Nguyen and Hsuan-Tien Lin

Abstract

In recent years, with the rapid development of neural network technology, the application of deep learning in the field of artificial intelligence has made significant progress and improvement. However, during the training of neural network models, the utilization of datasets is involved, and these datasets may contain sensitive information from users. Attackers might exploit the well-trained models to gain access to this sensitive information, leading to privacy breaches. Considering this risk, some deep learning algorithms incorporate differential privacy technology to safeguard the privacy of the trained model. This protection comes at the cost of certain model performance, achieved by adding controllable random noise. In this paper, we propose a differential privacy deep learning algorithm based on the importance of each layer's gradients, called DP-AdamILG. DP-AdamILG further mitigates the impact of noise addition on model performance. It accomplishes this by combining the dynamic privacy budget allocation strategy with the formation of noise gradients based on the importance of each layer's gradients. And the algorithm's privacy is theoretically proven. Experimental results show that the DP-AdamILG algorithm can reach good performance of the neural network model and show strong robustness.

Keywords: deep learning, differential privacy, dynamic privacy budget, layer-wise gradient processing.

1. Introduction

Differential privacy (Dwork (2006)) provides a provable privacy guarantee for the protection of users' sensitive information. The core concept of differential privacy is to introduce controllable random noise in the data processing process to perturb the user's sensitive information in a certain way, so as to ensure that individual privacy will not be disclosed in data analysis or query. Based on the robust privacy protection performance of differential privacy and its uncomplicated implementation, differential privacy has been widely applied in various contexts, including the fields of deep learning (Zhu et al. (2022)), reinforcement learning (Liao et al. (2023)), and so on. In the context of differential privacy deep learning, it

*. Corresponding Author

primarily fall into two categories: data-based perturbation and gradient-based perturbation. Many research efforts in differential privacy for deep learning have focused on gradient-based perturbation. This method generates noise gradients by adding noise to the gradients during the back-propagation process in model training. Then these noise gradients are subsequently utilized to update the weight parameters of the neural network model. In our paper, we also use gradient-based perturbation method.

However, several of previous gradient-based works have some constraints: First, in work (Abadi et al. (2016); Xu et al. (2020)), *a fixed privacy budget* is allocated throughout the entire training process. Second, in works (Abadi et al. (2016); Yu et al. (2019); Lee and Kifer (2018)), *a uniform noise scale* is used for all gradients, resulting in the sampling and addition of noise with the same magnitude. These approaches does not consider both the inherent distinctions in gradients between layers of neural networks and the variability in privacy budget allocation, resulting in an imbalance between the utility of the model and the level of privacy protection. Third, a few studies (Xiang et al. (2019, 2023); Chen et al. (2023)) have taken both of the above limitations into account. Unfortunately, the algorithms designed in these studies suffer from *excessive complexity* in their frameworks. This complexity poses challenges in deploying them as a universal framework across various domains in deep learning, significantly increasing algorithmic runtime and imposing heightened hardware requirements on computers.

In this paper, taking into account the aforementioned limitations, we propose a differential privacy deep learning algorithm based on the importance of each layer's gradients, that is DP-AdamILG. Firstly, *for fixed privacy budget*, we employ a dynamic privacy budget allocation strategy. This strategy allows the noise scale to decay dynamically based on the number of training epochs. Secondly, *for uniform noise scale*, we propose a method of importance-based adaptive layer-wise gradient processing. This approach can calculate the L_2 sensitivity and noise scale of each layer's gradients according to their respective importance. This not only accomplishes the objective of applying distinct clipping thresholds to various network layers, but also performs noise sampling in accordance with the noise scale of each layer's gradient. Thirdly, *for algorithm generality*, we made simple modifications to the DP-SGD framework, ensuring a concise implementation without excessive computational overhead. Not only that, for a convenient and rigorous analysis of the cumulative privacy loss of DP-AdamILG, we use zero-mean centralized differential privacy (zCDP) (Bun and Steinke (2016), a variant of centralized differential privacy (CDP) proposed by Dwork and Rothblum (2016).

We summarize the main contributions as follows:

- *Dynamic Differential Privacy:* We introduce a strategy for dynamic privacy budget allocation, aiming to progressively reduce the noise scale throughout the training process, all while incorporating the minimum privacy condition.
- *Algorithm Design for Each Layer's Gradient:* We propose a novel gradient processing approach, which is based on the importance of each layer's gradients in the neural network. This approach enables layer-wise gradient clipping and noise addition.
- *Algorithm Generality:* We present a novel framework for differential privacy deep learning algorithm. This framework is designed to be simple, easily integrable, and suitable for various deep learning scenarios.

- *Numerical Results:* We evaluate the DP-AdamILG performance across five different datasets, achieving improved test accuracy. Numerical results demonstrate the effective performance and robustness of DP-AdamILG in neural network model training across various architectures and datasets.

The remainder of this paper is as follows. We cover the related work in Section 2, review necessary background in Section 3. Followed by Section 4, we propose the algorithm which combines dynamic privacy budget allocation with processing gradients based on the importance of each layer's gradients. We present the experimental evaluation in Section 5. Finally, we conclude the paper in Section 6.

2. Related Work

Abadi et al. (2016) first introduced differential privacy to deep learning. They combined differential privacy mechanism with the stochastic gradient descent algorithm to form a new algorithm, called DP-SGD. Also, they presented the privacy analysis method "Moments Accountant", which enhances the rigor of privacy loss analysis. Subsequently, Yu et al. (2019) and Lee and Kifer (2018) enhanced the DP-SGD algorithm by presenting dynamic privacy budget allocation. Unlike the uniform allocation method, their approaches dynamically adjusted the privacy budget throughout the entire iterative process. In contrast to the two methods mentioned above, Xu et al. (2020) incorporated adaptive noise with varying magnitudes based on gradient L_2 sensitivity. They also employed an adaptive learning rate strategy when updating parameters. In addition to enhancing DP-SGD in terms of adaptive noise, Andrew et al. (2021) proposed the concept of adaptive clipping. They outlined a method for autonomously and adaptively adjusting the clipping threshold. At the same year, Hu et al. (2021) used clustering techniques to categorize gradients, and calculated the clipping threshold according to these groups for gradient clipping. Both of them transformed the fixed clipping threshold in the DP-SGD algorithm into an adaptive, dynamically changing threshold.

However, the improvement methods suggested for DP-SGD in these works either use a uniform noise scale or have a fixed privacy budget allocated throughout the entire training process. These limitations resulted in a suboptimal balance between model utility and privacy protection. Some works have taken into account these two shortcomings. Such as Xiang et al. (2019, 2023) improved the DP-SGD algorithm in terms of optimization. Chen et al. (2023) enhanced the DP-SGD algorithm with respect to gradient relevance. Although their methods can mitigate the aforementioned issues, but the implementations are overly complex and not suitable as a general framework.

3. Background

In this section, we will present the background of differential privacy, and introduce stochastic optimization techniques widely employed in deep learning.

3.1. Differential Privacy

Differential privacy (Dwork (2006)) provides a promising privacy protection mechanism for deep learning.

Definition 1 *(Neighbor Datasets) Two datasets D and D' are neighbor datasets, if their L_1 norm is at most 1.*

Definition 2 *(Differential privacy (Dwork (2006))) A randomized mechanism $\mathcal{M}$ is called (ϵ, δ)-differential privacy if for any two neighboring datasets D and D', for $\forall o \subseteq Range(\mathcal{M})$, it holds that*

$$Pr(\mathcal{M}(D) \in o) \leq e^{\epsilon} Pr(\mathcal{M}(D') \in o) + \delta, \tag{1}$$

where ϵ is the privacy budget and δ is the failure probability.

Especially, $\mathcal{M}$ implements approximate differential privacy if $\delta > 0$, and achieves a strictly stronger notion of pure differential privacy, named ϵ-differential privacy, if $\delta = 0$. In the remainder of this paper, we write (ϵ, δ)-DP and ϵ-DP for short.

One way to gain (ϵ, δ)-DP and ϵ-DP by using the *sensitivity* method, that is to adds noise sampled from Gaussian and Laplace distributions respectively, where the noise is proportional to the *sensitivity* of the query function.

Definition 3 *(Sensitivity (Dwork et al. (2006))) Given two neighboring data sets D and D', the L_2 sensitivity of a query function $q : \mathcal{D} \to \mathbb{R}^d$ is the maximum change in the output of q over all possible inputs:*

$$\Delta = \max_{D, D', ||D - D'||_1 = 1} ||q(D) - q(D')||_2. \tag{2}$$

Theorem 4 *(Gaussian Mechanism (Dwork et al. (2014))) Consider an arbitrary value $\epsilon \in (0, 1)$ and q be a query function with L_2 sensitivity of Δ. The Gaussian Mechanism with variance σ^2, which adds noise $N(0, \sigma^2 \Delta^2 \boldsymbol{I})$ to the output of $q(D)$, is (ϵ, δ)-DP if $\sigma \geq \frac{1}{\epsilon}\sqrt{2\log(1.25/\delta)}$ holds.*

To enhanced the trade-off between privacy and practicality of differential privacy protection, Bun and Steinke (2016) propose a variant of differential privacy, called zero-concentrated differential privacy (zCDP). It offers a flexible and adaptable framework for privacy protection, simplifying the analysis of privacy loss accumulation and ensuring robust privacy protection.

Definition 5 *(Zero-Concentrated Differential Privacy(zCDP) (Bun and Steinke (2016))) A randomized mechanism $\mathcal{M}$ is said to be ρ-zero concentrated differential privacy (ρ-zCDP), if for any two neighboring databases D and D', and for all $\alpha \in (1, \infty)$, the α-Rényi divergence between the distributions of $\mathcal{M}(D)$ and $\mathcal{M}(D')$, denoted as $D_\alpha(\mathcal{M}(D) \| \mathcal{M}(D'))$, it holds that:*

$$D_\alpha\left(\mathcal{M}(D) \| \mathcal{M}\left(D'\right)\right) \triangleq \frac{1}{\alpha - 1} \log\left(\mathbb{E}\left[e^{(\alpha-1)S(o)}\right]\right) \leq \rho\alpha. \tag{3}$$

Where $S(o)$ represents the privacy-loss random variable of mechanism $\mathcal{M}$ and can be presented by

$$S(o) = \log \frac{Pr[\mathcal{M}(\mathrm{D}) = o]}{Pr\left[\mathcal{M}\left(\mathrm{D}'\right) = o\right]}, \tag{4}$$

for any output $o \in range(\mathcal{M})$.

To restrict the moment generating function of the privacy loss $S(o)$, ρ-zCDP demands its concentration around zero. It needs to hold that:

$$\mathbb{E}\left[e^{(\alpha-1)S(o)}\right] \leq e^{(\alpha-1)\alpha\rho}, \forall\alpha \in (1,\infty). \tag{5}$$

In this paper, we utilize the following lemma for zCDP.

Lemma 6 *(Additive closure (Bun and Steinke (2016))) Given two mechanisms that satisfy ρ_1-zCDP and ρ_2-zCDP, their combination guarantees $(\rho_1+\rho_2)$-zCDP.*

Lemma 7 *(ρ-zCDP Gaussian Mechanism (Bun and Steinke (2016))) For a query function q with L_2 sensitivity of Δ, the Gaussian mechanism with variance σ^2, which outputs $q(D)+N(0,\sigma^2\Delta^2\boldsymbol{I})$, then it holds that $1/(2\sigma^2)$-zCDP.*

Lemma 8 *(ρ-zCDP to (ϵ,δ)-DP (Bun and Steinke (2016))) Suppose mechanism $\mathcal{M}$ satisfies ρ-zCDP, then $\mathcal{M}$ also satisfies (ϵ,δ)-DP for any $\delta>0$, where $\epsilon=\rho+2\sqrt{\rho\log(1/\delta)}$.*

3.2. Stochastic optimization techniques

Stochastic gradient descent (i.e., SGD) is a widely used optimization technique in deep learning model training. This algorithm aims to iteratively update the model parameters, with the goal of driving them towards a local minimum of the loss function. Typically, the training process of a neural network model is organized into epochs, and each epoch contains all the batches of the training dataset, meaning that within a single epoch, the entire dataset is processed once.

At present, several enhanced variants of SGD have been introduced, such as RMSprop, AdaGrad, Adadelta, Adam, etc., among which Adam proposed by Kingma and Ba (2014) stands out as one of the most widely used optimization algorithms in current practice. Adam combines the strengths of RMSprop and AdaGrad, handling sparse gradients and non-stationary targets. Additionally, it incorporates historical gradient information for more refined parameter updates. Our algorithm also utilizes Adam for gradient optimization.

4. Our Approach

DP-AdamILG algorithm adopts a method that combines dynamic privacy budget allocation with processing gradients based on the importance of each layer's gradients of neural network. A overview of our proposed framework is illustrated in Fig.1.

In this section, we will introduce the dynamic privacy budget allocation strategy. Additionally, we cover the implementation details of gradient processing based on the importance of each layer's gradients. Furthermore, we will provide the proof of privacy assurance for the DP-AdamILG algorithm. We also elaborate on the computation process for cumulative privacy loss during the training phase.

4.1. Dynamic privacy budget allocation

As training progresses, gradients converge and require a slower, more precise descent with minimal noise to avoid skipping local optima. Conversely, early iterations allow faster

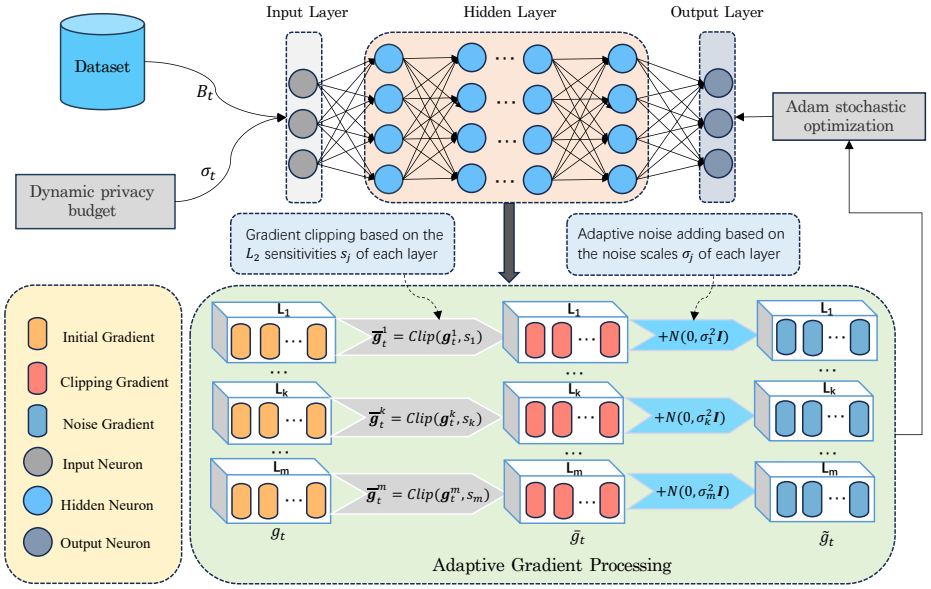


Figure 1: The Overview of DP-AdamILG

descent with higher noise tolerance, meaning additional noise has little impact on the final model's performance. However, maintaining a fixed privacy budget throughout training can add excessive noise when gradients are close to convergence. This surplus noise may alter the descent direction or cause overly rapid descent, bypassing local maxima, thus disrupting the balance between model utility and privacy protection. Therefore, to address the above problem, we use the method of dynamic privacy budget allocation.

The implementation of dynamic privacy budget is that:

Phase 1: **Parameters input phase**. We allocate the total privacy budget for model training in this phase.
Phase 2: **Training initialization phase**. A relatively substantial noise scale is employed for noise sampling within the model.
Phase 3: **Training convergence phase**. As epochs accumulate, the noise scale gradually decays, reducing the noise added to the gradients.

This strategy effectively reduces the impact of noise addition on model performance.

In our DP-AdamILG algorithm, we use the time-based decay function, through which the noise scale can decrease with the accumulation of epoch number. Its mathematical form is defined as:

$$\sigma_t = Decay(\sigma_0, k, t) = \sigma_0/(1 + k \cdot t), \tag{6}$$

where σ_0 is the initial input noise scale value, k is the decay rate. And t represents the cumulative change in training epochs. However, our observations indicate that using a time-based decay function for dynamic privacy budget allocation can cause issues. After a certain number of epochs, the noise scale may decay too close to zero, compromising privacy protection by reducing the added noise to nearly zero. Therefore, in contrast to previous

works (Yu et al. (2019)), we introduced an innovative improvement, which *involves incorporating a minimum privacy condition restriction* $\epsilon_{Lim} \in (0, 1)$ within the decay function, and calculates the noise scale threshold σ_* based on this condition. The noise scale threshold σ_* is calculated as:

$$\sigma_* = \sqrt{2\log(1.25/\delta)}/\epsilon_{Lim}. \tag{7}$$

This improvement enables the function to conduct a condition check before each noise scale decay, ensuring that the decayed noise scale meets the minimum privacy criteria. If it is met (i.e., $\sigma_t > \sigma_*$), the decay noise scale σ_t is used for the training model. Otherwise, the decay function inputs the noise scale threshold σ_*.

4.2. Adaptive Gradient Processing

An important component of our algorithm is adaptive gradient processing. We implement gradient clipping and noise addition based on the importance of each layer's gradients.

A key issue in gradient processing is establishing criteria for assessing importance. We observe variations in the L_2 norms of gradients across different layers. A larger L_2 norm indicates a higher overall gradient value and more information, highlighting the layer's importance. Conversely, lower L_2 norms indicate lesser significance. Therefore, in the DP-AdamILG algorithm, we use the L_2 norm to assess the importance of each layer's gradients.

The implementation of adaptive gradient processing is that: firstly, we calculate the L_2 sensitivities s_j of every layers based on the L_2 norm of each layer's gradients.

$$s_j = ||\boldsymbol{g}_t^j||_2 \cdot \alpha/\sqrt{n_j}, \tag{8}$$

where $1 \leq j \leq J$ denotes j^{th} layer of the neural network model with a total of J layers, $||\boldsymbol{g}_t^j||_2$ represents the L_2 norm of j^{th} layer gradient of a batch in the t iteration, α is the clipping factor and n_j signifies the dimension of j^{th} layer gradient.

Secondly, we employ these L_2 sensitivities s_j as the clipping thresholds to clip the gradients of each layer respectively.

$$\overline{\boldsymbol{g}}_t^j = Clip(\boldsymbol{g}_t^j, s_j) = min\{max\{g_t^{ji}, -s_j\}, s_j\}, \tag{9}$$

where g_t^{ji} represents i^{th} dimension of j^{th} layer's gradients and $\forall i = 1, 2, \cdots, n_j$. Unlike per-example clipping in DP-SGD, our approach accomplishes *layer-wise gradient clipping*.

Finally, according to each layer's L_2 sensitivity s_j, each layer's noise scale σ_j is calculated, followed by adaptive Gaussian noise sampling. Subsequently, these noises are incorporated into the gradients to create the noisy gradients. To satisfy the conditions of Lemma 9 (given later), the noise scale σ_j of each layer is assessed as:

$$\sigma_j = s_j \cdot \sqrt{n_j} \cdot \sigma_t, \tag{10}$$

where σ_t is the decayed noise scale input for the current epoch.

Compared to previous differential privacy deep learning algorithms, such as ADADP ((Xu et al. (2020))), which estimates the current gradient sensitivities by recording historical gradient values, we introduce the concept of gradient importance of each layer, and use the L_2 norm as an index to calculate each layer's L_2 sensitivity and noise scale. On one

hand, our algorithm scientifically implements layer-wise gradient clipping by considering variations in L_2 sensitivity across layers. Layers with higher L_2 norms require more noise due to greater sensitivity. On the other hand, as gradients converge and their magnitudes diminish, noise sampling based on each layer's L_2 norm ensures that noise magnitude decays appropriately. This approach adapts the noise distribution throughout the gradient descent process according to the importance of each layer's gradients.

We outlines each step of our algorithm at Algorithm 1. It builds on the basic DP-SGD algorithm with a few key computations, making it easy to reproduce and apply.

Algorithm 1 DP-AdamILG

Input: Training dataset D, learning-rate η, loss function $L(\boldsymbol{\omega}, d)$, initial noise scale σ_0, batch size B, clipping factor α, decay rate k, privacy condition restriction ϵ_{Lim}, total privacy budget ρ_{tot}

Output: $\boldsymbol{\omega}_T$

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

  Allocate privacy budget: $\sigma_t \leftarrow Decay(\sigma_0, k, t)$;

  if $\sigma_t < \sigma_*$(calculated by Eq.7), $\sigma_t \leftarrow \sigma_*$;

  Calculate the remaining privacy budget: $\rho_{tot} \leftarrow \rho_{tot} - 1/(2 \cdot \sigma_t^2)$;

  if $\rho_{tot} < 0$, break;

  Take a random sample B_t with size B from D;

  Compute the gradient of B_t: For each $i \in B_t$, $\boldsymbol{g}_t(d_i) \leftarrow \nabla L(\boldsymbol{\omega}_t, d_i)$;

  $\boldsymbol{g}_t \leftarrow \frac{1}{B} \sum_i \boldsymbol{g}_t(d_i)$;

  for *the* j^{th} *layer of the neural network* **do**

    Compute the sensitivity: $s_j \leftarrow ||\boldsymbol{g}_t^j||_2 \cdot \alpha / \sqrt{n_j}$;

    Gradient clipping: $\overline{\boldsymbol{g}}_t^j \leftarrow Clip(\boldsymbol{g}_t^j, s_j)$;

    Compute the noise scale: $\sigma_j \leftarrow s_j \cdot \sqrt{n_j} \cdot \sigma_t$;

    Add the adaptive noise:$\widetilde{\boldsymbol{g}}_t^j \leftarrow \overline{\boldsymbol{g}}_t^j + N(0, \sigma_j^2 \boldsymbol{I})$;

  end

  Use Adam update the parameter;

end

4.3. Privacy Analytics

Lemma 9 *(Xu et al. (2020)) Consider a mechanism $\mathcal{M}(D) = f(D) + Z$ is (ϵ, δ)-DP, where D is the input dataset, $f(\cdot)$ is an m-dimension function with L_2 sensitivity $\Delta \leq 1$, and noise $Z \sim N(0, \sigma_t^2)$. Now, another mechanism $\mathcal{M}'(D) = f'(D) + Z'$ is also (ϵ, δ)-DP if $\sum_{i=1}^{m} \frac{s_i^2}{\sigma_i^2} \leq \frac{1}{\sigma_t^2}$, where $f'(\cdot)$ is also an m-dimension function, $Z' = (z_1', \cdots, z_m')^T$ with $z_i' \sim N(0, \sigma_i^2)$, s_i is the L_2 sensitivity of the i^{th} dimension of $f'(\cdot)$, for $i = 1, 2, \cdots, m$.*

Based on Lemma 9 and Theorem 4, along with rigorous privacy analysis and experimental comparisons, our proposed algorithm leads to the following corollaries:

Corollary 10 *Algorithm 1 satisfies (ϵ, δ)-DP.*

Proof Consider the gradients returned by the current batch B_t are that $f(\boldsymbol{\omega}_t, B_t) + N(0, \sigma_j^2 \boldsymbol{I}) = f(\boldsymbol{\omega}_t, B_t) + N(0, \sigma_t^2 \cdot \Delta^2 \boldsymbol{I})$, where $f(\boldsymbol{\omega}_t, B_t)$ denotes the clipped gradient values for the subset B_t with parameters $\boldsymbol{\omega}_t$, and $\Delta^2 = s_j^2 \cdot n_j$. Followed by Theorem 4, the overall gradients returned by this iteration satisfies(ϵ, δ)-DP, and $\epsilon = \sqrt{2\log(1.25/\delta)}/\sigma_t$. At the same time, to meet the condition $\epsilon \in (0, 1)$, the noise scale of the initial input is limited by $\sigma_t \geq \sqrt{2\log(1.25/\delta)}$.

Let ω_i^j represent i^{th} weight parameter of j^{th} layer. Each layer of the neural network is composed of multiple parameters $\boldsymbol{\omega}^j = (\omega_1^j, \omega_2^j, \cdots, \omega_n^j)$, and the noise added to the gradient of each layer should also be n-dimension.

DP-AdamILG algorithm first calculates the L_2 norm of each layer gradients: $||\boldsymbol{g}_t^j||_2$, according to which the L_2 sensitivity s_j of each layer is obtained (calculated by Eq.8). Then the noise scale σ_j of each layer is gained (calculated by Eq.10). For the gradient of j^{th} layer we can obtain : $\sum_{i=1}^{n_j} \frac{s_j^2}{\sigma_j^2} = \sum_{i=1}^{n_j} \frac{s_j^2}{\sigma_t^2 \cdot s_j^2 \cdot n_j} = \frac{1}{\sigma_t^2}$, satisfying the inequality : $\sum_{i=1}^{n_j} \frac{s_j^2}{\sigma_j^2} \leq \frac{1}{\sigma_t^2}$. According to Lemma 9, the noise gradients of each layer returned by the Alg 1 are satisfied (ϵ, δ)-DP. ■

Theorem 11 *(Parallel Composability (Yu et al. (2019))) Let $\mathcal{M}$ be a composite mechanism consisting of a sequence of k adaptive mechanisms, denoted as $\mathcal{M}_1, \ldots, \mathcal{M}_k$. Each $\mathcal{M}_i : \prod_{j=1}^{i-1} \mathbb{R}_j \times \mathcal{D} \to \mathbb{R}_i$ and satisfies ρ_i-zCDP $(1 \leq i \leq k)$. Consider $D_1, D_2, \ldots, D_k$ resulting from a randomized partitioning of the input dataset D. The composite mechanism $\mathcal{M}(D) = (\mathcal{M}_1(D \cap D_1), \ldots, \mathcal{M}_k(D \cap D_k))$ adheres to $\max_i \rho_i$-zCDP.*

Suppose there is $\rho_i = \rho$ for all $i \in (1, \cdots, k)$, then in this case, the mechanism $\mathcal{M}$ satisfies ρ-zCDP according to Theorem 11.

The neural network training process involves reshuffling the dataset into batches for each iteration. An epoch consists of training on disjoint subsets $D = (D_1, \ldots, D_k)$, where each subset D_i is distinct. If the input noise scale is σ_t, the noise scale for any subset B_i is also σ_t. According to Theorem 11, the privacy cost for the current epoch ρ_t is given by Lemma 7: $\rho_t = \frac{1}{2\sigma_t^2}$. For N epochs, followed by Lemma 6, the total privacy cost is $\rho_{tot} = \sum_{t=1}^{N} \rho_t$, resulting in ρ_{tot}-zCDP. Based on Lemma 8, the total training process satisfies $(\rho_{tot} + 2\sqrt{\rho_{tot}\log(1/\delta)}, \delta)$-DP, and each epoch satisfies $(\rho_t + 2\sqrt{\rho_t \log(1/\delta)}, \delta)$-DP.

5. Experiment Evaluation

In this section, we answer the following three question:
- **Q1**: Can DP-AdamILG achieve good convergence speed and generalization performance?
- **Q2**: Is DP-AdamILG exhibit a certain stability with respect to hyperparameter tuning?
- **Q3**: Does the DP-AdamILG exhibit good robustness?

5.1. Experimental settings

Our primary evaluation relies on five datasets: MNIST (LeCun et al. (1998)) , Fashion-MNIST(FMNIST) (Xiao et al. (2017)) , CIFAR-10 (Krizhevsky et al. (2009)) , CIFAR-100 (Krizhevsky et al. (2009)) and SVHN (Netzer et al. (2011)).

For the entire experiments, we fixed the failure probability $\delta = 10^{-5}$. And for clarity, we present the other parameter settings in Table 1. Unless specified otherwise, the experimental parameters follow the settings in Table 1. Specially, the selection of ρ_{tot} is guided by Yu et al. (2019), with the aim of facilitating experimental comparisons and ensuring that the training process maintains a certain number of epochs. All experiments are conducted on an Intel 8 core i7-10700 U CPU@2.90 GHz machine with an NVIDIA A10 GPU.

Table 1: Summary of parameter setting

	CIFAR-10, CIFAR-100 and SVHN	MNIST and FMNIST
σ_0	6	10
ρ_{tot}	1.5625	0.78125
k	0.001	0.05
B	200	600
α	1.2	1.2
Neural Network	ResNet-18	I.P.R.O.[1]

[1] I.P.R.O. represents the neural network structure consisting of the input layer, PCA layer, ReLU layer, and output layer.

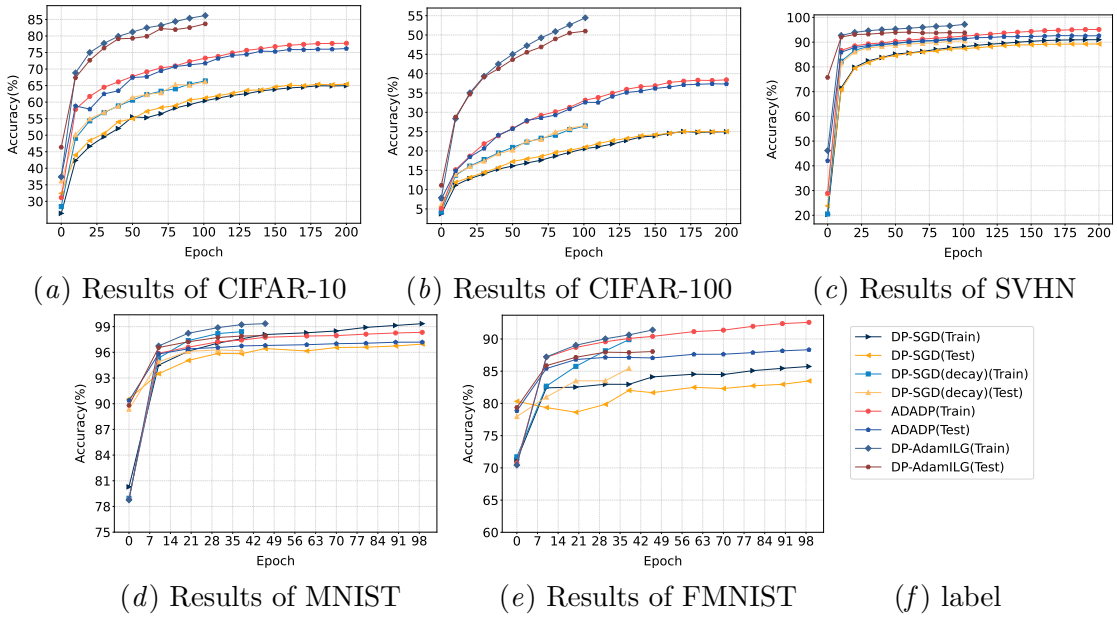
5.2. Experimental Result

5.2.1. Comparison of Different Algorithms Performance and Privacy Cost

To answer **Q1**, we conduct comparative analysis of our algorithm against DP-SGD (Abadi et al. (2016)), DP-SGD (decay) (Yu et al. (2019)) and ADADP (Xu et al. (2020)) to evaluate its performance. We selected these three algorithms for comparison for the following reasons: Firstly, DP-SGD incorporates differential privacy by adding noise to gradients at each iteration, serving as a universal framework in differential privacy deep learning. Secondly, DP-SGD (decay) extends DP-SGD with dynamic privacy budget allocation, inspiring the adaptive privacy strategy in our proposed algorithm. Lastly, ADADP algorithm leverages RMSprop to integrate adaptive learning rates and noise into DP-SGD. It stands out as a relatively new and effective algorithm in the realm of differential privacy deep learning.

Experimental Results and Analysis: We compared the convergence speed and generalization performance of the aforementioned algorithm on five datasets. It can be observed from Fig.2 that, for the training tasks on the CIFAR-10 and CIFAR-100 datasets, our proposed algorithm outperforms DP-SGD, DP-SGD(decay) and ADADP. For MNIST and SVHN, the accuracy trends of the four algorithms are quite close, but our algorithm still exhibits better performance. While ADADP achieves slightly higher final training and testing accuracy compared to DP-AdamILG on the FMNIST dataset, our approach maintains an advantage in terms of convergence speed. This reveal that the proposed algorithm, incorporating dynamic privacy budget and gradients processing based on the importance of each layer's gradients, *outperforms ADADP, DP-SGD and DP-SGD(decay) in both convergence speed and generalization performance.* At the same time, the gap between the training and testing accuracy for DP-AdamILG is 1% to 4%, which is an acceptable range compared with other algorithms. Therefore, there are no indications of over-fitting.

We also measured the privacy costs of DP-AdamILG, ADADP, and DP-SGD (decay) in achieving predefined accuracy levels on five datasets. The experimental results in Table 2



(*a*) Results of CIFAR-10 (*b*) Results of CIFAR-100 (*c*) Results of SVHN

(*d*) Results of MNIST (*e*) Results of FMNIST (*f*) label

Figure 2: Comparative Analysis of Model Training Using Different Algorithms

Table 2: Experimental Results on the privacy cost of different Algorithm

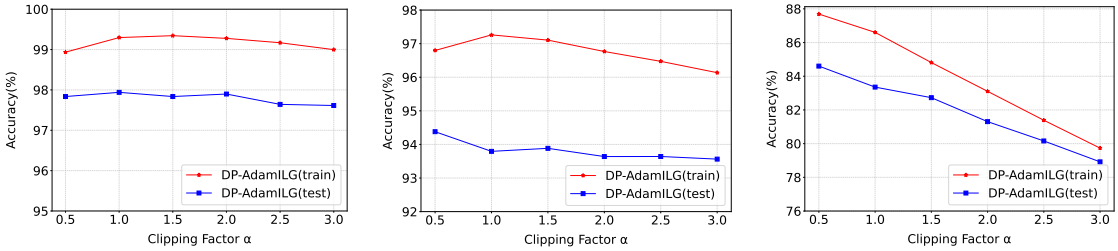
Dataset	Accuracy(%)	δ	DP-AdamILG	ADADP (Xu et al. (2020))	DP-SGD(decay) (Yu et al. (2019))	
MNIST	96.5	10^{-5}	**2.238** **12**	3.703 33	6.885 39	ϵ Epoch
FMNIST	85.5	10^{-5}	**1.948** **10**	2.172 12	7.091 40	ϵ Epoch
SVHN	90.2	10^{-5}	**1.862** **5**	5.299 64	9.227 89	ϵ Epoch
CIFAR-10	65.2	10^{-5}	**2.220** **7**	4.051 39	8.705 81	ϵ Epoch
CIFAR-100	23.5	10^{-5}	**2.382** **8**	3.938 37	8.305 75	ϵ Epoch

display the privacy cost(ϵ) required by each algorithm to achieve the desired accuracy, alongside the corresponding number of epochs. It can be seen that DP-AdamILG significantly outperforms both ADADP and DP-SGD (decay) in terms of training epochs and privacy cost on the five datasets. Compared to ADADP and DP-SGD(decay), DP-AdamILG reduces the privacy cost by an average of 39.6% and 67.5% on MNIST, 10.3% and 72.5% on FMNIST, 64.7% and 79.8% on SVHN, 45.2% and 74.5% on CIFAR-10, 39.5% and 71.3% on CIFAR-100. This demonstrates that the proposed algorithm achieves a *good balance* between accuracy and privacy cost. Furthermore, DP-AdamILG requires fewer training epochs to reach the predefined accuracy level, indicating a *faster convergence speed*.

5.2.2. Hyper-parameter Effects

The DP-AdamILG algorithm involves two key hyperparameters, namely the clipping factor α and the privacy budget decay rate k. The different values of these two hyperparameters

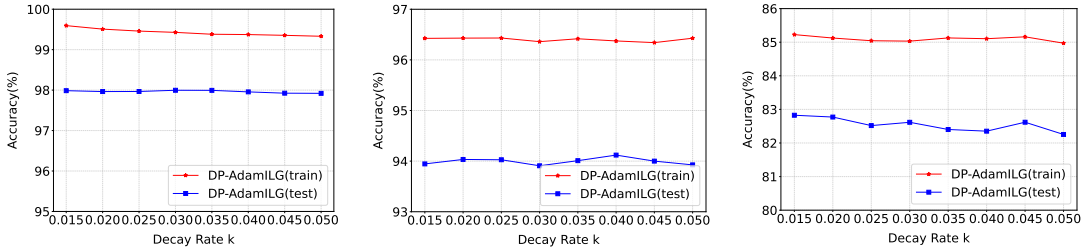
may affect the training of the model. So in this section we investigate the impact of these two hyperparameters to answer **Q2**. We will use the datasets MNIST, CIFAR-10, and SVHN for this experiment. This experiment will employ the control variable method. Unless otherwise specified, all parameters will remain at their default values as Table.1, except for the hyperparameters under investigation.



(*a*) MNIST (*b*) SVHN (*c*) CIFAR-10

Figure 3: The Impact of Various Clipping Factors on Model Performance

Analysis of Experimental Results with Different Clipping Factors: The clipping factor α controls the scales of each layer's gradients of neural network. A smaller α removes more gradient information, whereas a larger one introduces more noise. Fig.3 shows that, for the MNIST and SVHN dataset, the impact of the clipping factor α is relatively stable, with no significant fluctuations. Additionally, the algorithm achieves its highest performance when the α is around 1.0. However, training on CIFAR-10 exhibits a distinct behavior. As α increases, model performance gradually decreases, particularly for CIFAR-10, which is more sensitive to noise. The increased noise significantly impacts CIFAR-10's performance, leading to a continuous decline. Hence, to preserve meaningful gradient information and incorporate noise of an appropriate magnitude, a reasonable setting for α is to take a value slightly greater than 1.0.



(*a*) MNIST (*b*) SVHN (*c*) CIFAR-10

Figure 4: The Impact of Various Decay Rates on Model Performance

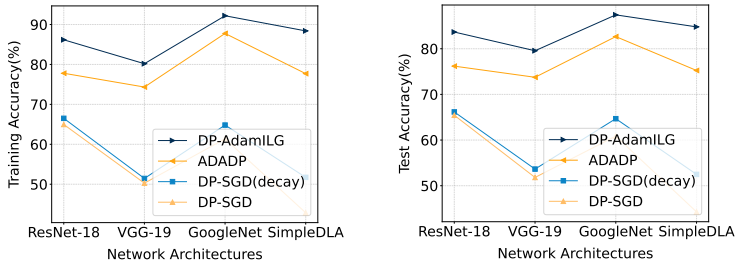
Analysis of Experimental Results with Different Decay Rates: The decay rate k determines the speed at which the noise scale decreases. Given the total privacy budget, a higher decay rate consumes the privacy budget more quickly, resulting in fewer total epochs. Fig.4 shows that the decay rate k has minimal impact on the training of the proposed algorithm, remaining stable across different k values. The results show that the performance gap between the highest and lowest model performance with varying decay rates does not exceed

0.5%. This indicates that DP-AdamILG achieves faster gradient convergence and maintains stability during training, ensuring consistent final model performance despite changes in the decay rate. The findings also suggest that the rapid convergence characteristic of the DP-AdamILG algorithm is applicable across different datasets. To achieve comparable results with ADADP and DP-SGD(decay) effectively, the default values for the decay rate parameter k in experiments are set as Table 1.

The experimental results indicate that DP-AdamILG's hyperparameters have *minimal impact* on overall training outcomes, despite variations in clipping factors and decay rates. However, the clipping factor α significantly affects the CIFAR-10 dataset.

5.2.3. The Impact of Different Network Architectures

To answer **Q3**, we evaluate DP-AdamILG's robustness by training ResNet-18 (He et al. (2016)), VGG-19 (Simonyan and Zisserman (2014)), GoogLeNet (Szegedy et al. (2015)), and SimpleDLA (Yu et al. (2018)) on the CIFAR-10 dataset. This evaluation compares DP-AdamILG with DP-SGD, DP-SGD (decay), and ADADP.



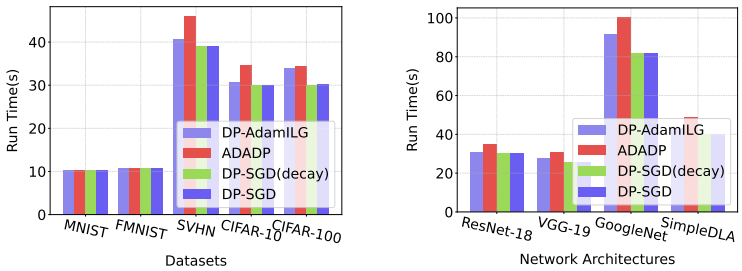
(*a*) Training Accuracy (*b*) Test Accuracy

Figure 5: Different Algorithms Performance Across Different Network Architectures

Analysis of Experimental Results: Different algorithms exhibit varying training effects on different neural networks. Fig.5 shows that, with the same parameters, DP-AdamILG and ADADP achieve the best performance with GoogLeNet, while DP-SGD and DP-SGD (decay) perform best with ResNet-18. Nonetheless, when training on the CIFAR-10 dataset using four classical network architectures, the DP-AdamILG algorithm consistently shows superior performance in both training and test accuracy. This indicates that DP-AdamILG achieves rapid convergence and strong generalization across various network architectures. Thus, demonstrating *robustness and adaptability* in model training across diverse models.

We also evaluated the operational efficiency of DP-SGD, DP-SGD (decay), ADADP, and DP-AdamILG by measuring their runtime per epoch. Fig.6 indicates that our algorithm, being an improvement on accuracy and framework upon DP-SGD and DP-SGD(decay), exhibits slightly longer runtime compared to both. In contrast, our algorithm demonstrates significant improvement in operational efficiency compared to ADADP, suggesting its structural simplicity and ease of implementation.

The three aforementioned experiments validate DP-AdamILG from three key perspectives: algorithm performance, hyperparameter stability, and algorithm robustness. The Table 3 summarizes the training results of DP-SGD, DP-SGD (decay), ADADP, and DP-AdamILG across different datasets and network architectures. For each algorithm, the first



(*a*) Different Datasets (*b*) Different Network

Figure 6: Running Time of Different Datasets and Network Architectures

Table 3: Experimental results on various datasets and network architectures

	Datasets					**Network Architectures**			
	MNIST	FMNIST	SVNH	CIFAR-10	CIFAR-100	ResNet-18	VGG-19	GoogleNet	SimpleDLA
DP-SGD (Abadi et al. (2016))	99.347	85.735	90.993	64.964	24.972	64.964	50.266	61.370	42.854
	96.960	83.490	89.259	65.430	25.050	65.430	51.810	60.860	44.270
	10.291	10.688	**39.041**	**29.920**	30.154	**29.920**	**25.415**	**81.566**	**39.929**
DP-SGD(decay) (Yu et al. (2019))	98.497	89.897	91.713	66.406	26.468	66.506	51.450	64.816	51.718
	96.460	85.430	90.500	66.160	26.580	66.160	53.640	64.680	52.480
	10.303	**10.682**	39.045	29.929	**30.033**	29.929	25.456	81.782	39.940
ADADP (Xu et al. (2020))	98.335	**92.580**	95.127	77.195	38.400	77.794	74.338	87.764	77.694
	97.190	**88.330**	92.605	76.190	37.330	76.190	73.750	82.660	75.240
	10.432	10.744	45.915	34.664	34.495	34.664	30.619	100.160	48.642
Our	**99.353**	91.398	**97.181**	**86.174**	**54.446**	**86.174**	**80.180**	**92.190**	**88.400**
	97.950	88.050	**93.800**	**83.600**	**51.000**	**83.660**	**79.560**	**87.420**	**84.790**
	10.208	10.684	40.678	30.754	33.855	30.754	27.565	91.614	43.101

row represents training accuracy(%), the second row indicates test accuracy(%) and the thirh row denotes per epoch running time(s).

6. Conclusion

In this paper, we propose a novel differential privacy deep learning algorithm framework, called DP-AdamILG. Firstly, we introduce an enhanced strategy for dynamically allocating privacy budget. This adaptation ensures that the privacy budget decays gradually, better meeting practical requirements. Secondly, we implement gradient clipping and noise addition based on the importance of each layer's gradients. Additionally, we elaborate on the privacy protection in DP-AdamILG and present a cumulative method for privacy loss calculation. Experimental results demonstrate the good performance and great robustness of our DP-AdamILG algorithm. Future work will involve comparing our algorithm with more differential privacy deep learning methods.

Acknowledgments

This work was supported by the Special Project for Research and Development in Key areas of Guangdong Province, China (Grant No. 2020B0101090003) and the Natural Science Foundation of Guangdong Province, China (Grant No. 2021A1515011607). The opinions

in this paper are those of the authors and do not necessarily reflect the opinions of any funding sponsor or the China Government.

References

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

Galen Andrew, Om Thakkar, Brendan McMahan, and Swaroop Ramaswamy. Differentially private learning with adaptive clipping. *Advances in Neural Information Processing Systems*, 34:17455–17466, 2021.

Mark Bun and Thomas Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography Conference*, pages 635–658. Springer, 2016.

Lin Chen, Danyang Yue, Xiaofeng Ding, Zuan Wang, Kim-Kwang Raymond Choo, and Hai Jin. Differentially private deep learning with dynamic privacy budget allocation and adaptive optimization. *IEEE Transactions on Information Forensics and Security*, 2023.

Cynthia Dwork. Differential privacy. In *International colloquium on automata, languages, and programming*, pages 1–12. Springer, 2006.

Cynthia Dwork and Guy N Rothblum. Concentrated differential privacy. *arXiv preprint arXiv:1603.01887*, 2016.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*, pages 265–284. Springer, 2006.

Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Yuhang Hu, Zhou Tan, Xianxian Li, Jinyan Wang, et al. Adaptive clipping bound of deep learning with differential privacy. In *2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 428–435. IEEE, 2021.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Jaewoo Lee and Daniel Kifer. Concentrated differentially private gradient descent with adaptive per-iteration privacy budget. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1656–1665, 2018.

Chonghua Liao, Jiafan He, and Quanquan Gu. Locally differentially private reinforcement learning for linear mixture markov decision processes. In *Asian Conference on Machine Learning*, pages 627–642. PMLR, 2023.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

Liyao Xiang, Jingbo Yang, and Baochun Li. Differentially-private deep learning from an optimization perspective. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pages 559–567. IEEE, 2019.

Liyao Xiang, Weiting Li, Jungang Yang, Xinbing Wang, and Baochun Li. Differentially-private deep learning with directional noise. *IEEE Transactions on Mobile Computing*, 22(5):2599–2612, 2023.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Zhiying Xu, Shuyu Shi, Alex X Liu, Jun Zhao, and Lin Chen. An adaptive and fast convergent approach to differentially private deep learning. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 1867–1876. IEEE, 2020.

Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2403–2412, 2018.

Lei Yu, Ling Liu, Calton Pu, Mehmet Emre Gursoy, and Stacey Truex. Differentially private model publishing for deep learning. In *2019 IEEE symposium on security and privacy (SP)*, pages 332–349. IEEE, 2019.

Tianqing Zhu, Dayong Ye, Wei Wang, Wanlei Zhou, and S Yu Philip. More than privacy: Applying differential privacy in key areas of artificial intelligence. *IEEE Transactions on Knowledge & Data Engineering*, 34(06):2824–2843, 2022.