Private Deep Neural Network Models Publishing for Machine Learning as a Service

Yunlong Mao, Boyu Zhu, Wenbo Hong, Zhifei Zhu, Yuan Zhang, Sheng Zhong
State Key Laboratory for Novel Software Technology, Nanjing University, China

***Abstract*—Machine learning as a service has emerged recently to relieve tensions between heavy deep learning tasks and increasing application demands. A deep learning service provider could help its clients to benefit from deep learning techniques at an affordable price instead of huge resource consumption. However, the service provider may have serious concerns about model privacy when a deep neural network model is published. Previous model publishing solutions mainly depend on additional artificial noise. By adding elaborated noises to parameters or gradients during the training phase, strong privacy guarantees like differential privacy could be achieved. However, this kind of approach cannot give guarantees on some other aspects, such as the quality of the disturbingly trained model and the convergence of the modified learning algorithm. In this paper, we propose an alternative private deep neural network model publishing solution, which caused no interference in the original training phase. We provide privacy, convergence and quality guarantees for the published model at the same time. Furthermore, our solution can achieve a smaller privacy budget when compared with artificial noise based training solutions proposed in previous works. Specifically, our solution gives an acceptable test accuracy with privacy budget $\epsilon = 1$. Meanwhile, membership inference attack accuracy will be deceased from nearly 90% to around 60% across all classes.**

***Index Terms*—Quality of MLaaS, Deep Neural Network, Differential Privacy, Model Publishing**

I. Introduction

Deep Neural Networks (DNNs) have significantly improved the user experience of many applications in recent years, such as personal advertisement recommendation, facial recognition based authentication and voice controlled smart devices. By learning the hidden relationship between input and output on a large training dataset, a DNN model is supposed to be sophisticated enough to approximate any function under some assumptions according to the universal approximation theorem in [1]. However, training such a DNN model usually takes extremely high computing resources and huge data bulk. Even with powerful graphic processing units (GPUs), DNN training is still not easy [2].

To ease the imbalance between intensive computing load and rising demands for DNN applications, many designs of machine learning as a service (MLaaS) have emerged, such as Google AI Platform, AWS Deep Learning, Azure Machine Learning Service. Among all kinds of MLaaS, sharing well-trained DNN models by publishing model parameters (or weights, interchangeably) directly may be the most efficient way [3]–[5]. It is widely agreed that publishing model parameters directly could make the most use of well-trained DNN models for MLaaS clients. However, DNN models trained on private datasets are intellectual properties of companies or research agencies. It is highly risky to publish privately trained DNN models, especially these models trained with sensitive data (e.g. financial data or diagnostic history). Although the publisher could sign an end-user license agreement (EULA) with MLaaS clients, EULA cannot prevent published models from being abused. Because it is hard to prove a DNN model is some company's intellectual property or not [6].

Recent research works have found that publishing a well-trained DNN model without any privacy protection could cause serious privacy leakage, including membership inference [7], [8], task property [9], [10] and representative data reconstruction [11], [12]. Data privacy issues in DNN model publishing are so serious that many research works have been focusing on it in recent years. Generally, recently proposed solutions can be categorized into three types, i.e. adversarial training [8], secure computing [13] and differentially private training [14]. Adversarial training solutions will have a task model and an adversarial model trained together in a game way. When the training task ends, a trained task model can be more robust against privacy leakage threats. Secure computing solution based on cryptographic tools can provide strong security guarantees. Differentially private training solutions like [14] could provide a strong privacy guarantee because the artificial noise is added to interfere the training phase. These solutions have good performance on defence against data leakage threats. However, existing solutions cannot achieve multiple desired features at the same time, like model quality, training efficiency and model privacy.

In this paper, we propose an alternative private DNN model publishing approach without any interference in the training phase. Specifically, we collect intermediate training results of multiple training tasks for the same DNN architecture to construct a new dataset. We name this new dataset as "parameter collection", which has all parameters of a DNN model as one entry. Each parameter in an entry could be seen as a feature of the whole dataset. Then by performing some learning algorithm on parameter collection, we could obtain an approximate distribution of model parameters in stable states. Once we have learned an approximate shape of parameter distribution, it is possible to construct a DNN model that is

The first two authors contribute equally to this work.

978-1-7281-6887-6/20/$31.00 ©2020 IEEE

entirely different from the real private model, by generating parameters on estimated distributions.

We find that the generated DNN model has almost the same testing accuracy as the really trained ones. Hence, we propose a private DNN model publishing approach based on the parameter generating method. To prevent potential privacy leakage of parameter collection, we construct our publishing solution with differentially private query mechanisms. To be persuasive, we prove that our publishing solution satisfies differential privacy formally and the generated DNN model is resistant to model privacy threats like membership inference attack experimentally. It should be noted that preserving model privacy in deep learning is challenging because there is a conflict between model quality (e.g. testing accuracy) and model privacy. It is almost impossible to achieve high model quality and low privacy loss in deep learning at the same time. Although strong private publishing solutions could meet privacy guarantees, sometimes it is necessary to relax privacy guarantees to save model quality. Hence, we also give a high-quality publishing solution by taking the hidden connection of parameters into consideration. Our main contributions in this paper can be summarized as follows.

1) We find parameter similarity in well-trained DNN models between separate training tasks. Based on this observation, we propose a private DNN model publishing solution by generating parameters within an approximately estimated space.
2) To achieve a tradeoff between model quality and privacy, we also propose a hybrid private model publishing solution by combining the original solution with a private parameter grouping method, which preserves connections between crucial parameters.
3) We formally prove that DNN models published by our solutions are differentially private. We also prove that published models are resistant to model privacy threats like membership inference attack experimentally. Compared with a state-of-the-art private model publishing solution, our solutions can improve model quality significantly.

II. Preliminary

A. Deep Learning as a Service

Training a DNN model commonly requires substantial datasets and huge computing resources, either of which could be the obstacle of users. Sharing well-trained DNN models could save considerable resources. Generally, we regard a data owner who wants to publish a DNN model trained with a private dataset as a publisher and call a user going to use the published model a client. In MLaaS scenario, the publisher may not only share well-trained DNN models but also provide intermediate results to meet the needs of different clients. In this paper, we will discuss the situation where one MLaaS provider serves as the publisher for one client. But it should be noted that our solutions will also work for multiple clients since each client is served independently.

Here we will briefly review some important steps in deep learning task. Given training dataset X and DNN model θ, training task is to find approximately optimal parameters in θ by minimizing the loss function $\mathcal{L}$ on input X. Assume that the optimizer used in minimizing procedure is mini-batch stochastic gradient descent (SGD) algorithm, which updates θ for a small batch of X. Assuming the batch size is N, then the total loss of θ on random input samples $\boldsymbol{x} = \{x_i | x_i \in d, 1 \leq i \leq N\}$ will be $\sum_{x \in \boldsymbol{x}} \mathcal{L}(\theta, x)$ for the t-th training iteration. Further, the gradients of θ should be estimated by $\frac{1}{N} \sum_{x \in \boldsymbol{x}} \nabla_\theta \mathcal{L}(\theta, x)$ approximately. Hence parameters of θ could be updated before the next iteration as $\theta^{t+1} = \theta^t - \frac{1}{N} \sum_{x \in \boldsymbol{x}} \nabla_\theta \mathcal{L}(\theta, x)$.

B. Differential Privacy

Differential privacy (DP) [15] has been a defacto standard for privacy-preserving data publishing and analysis in recent years. DP provides a general framework for building privacy-preserving mechanisms for specific applications by following DP-based mechanism construction.

Definition 1 (Differential Privacy): A random mechanism $M : \mathcal{D} \to \mathcal{R}$ with domain $\mathcal{D}$ and range $\mathcal{R}$ satisfies (ϵ, δ)-differential privacy if for any two adjacent inputs $d, d' \in \mathcal{D}$ and for any subsets of outputs $s \subset \mathcal{R}$ it holds that

$$\Pr[M(d) \in S] \leq e^\epsilon \Pr[M(d') \in S] + \delta. \tag{1}$$

This definition of differential privacy proposed in [16] ensures that original ϵ-DP can be broken with probability δ. To build an (ϵ, δ)-DP mechanism for function $f : \mathcal{D} \to \mathcal{R}$, artificial noise corresponding to the sensitivity S_f of function f is needed, where S_f can be defined as

$$S_f = \max_{d, d' \in \mathcal{D}} |f(d) - f(d')|. \tag{2}$$

If Gaussian distribution is used, then an (ϵ, δ)-DP mechanism for f can be achieved by

$$M(d) \triangleq f(d) + \mathcal{N}(0, S_f^2 \cdot \sigma^2), \tag{3}$$

where $\mathcal{N}(0, S_f^2 \cdot \sigma^2)$ is a Gaussian distribution with mean 0 and standard deviation $S_f \sigma$. Various DP techniques have been proposed for different DP-mechanism designs. In this paper, we will mainly involve exponential mechanism (EM) [17].

We will use exponential mechanism (EM) proposed in [18], which is generally designed for query function $q : \mathcal{D}^n \times \mathcal{R} \to \mathcal{R}$, which will assign a real valued score to data pair (d, r) drawn from $\mathcal{D}^n \times \mathcal{R}$. Given $d \in \mathcal{D}^n$, EM will return an $r \in \mathcal{R}$ which will maximize score $q(d, r)$ approximately. To meet that higher score will occur more frequently, a base measure μ associated with r is designed in [18].

Definition 2: For any function $q : \mathcal{D}^n \times \mathcal{R} \to \mathcal{R}$, and base measure μ over $\mathcal{R}$, we define $\varepsilon_q^\epsilon(d)$ as choosing r with probability proportional to $\exp(\epsilon q(d, r)) \times \mu(r)$.

Following the theorem in [18], ε_q^ϵ satisfies $(2\epsilon\Delta_q)$-DP.

C. Privacy Threat against Model Publishing

Generally, once the whole DNN model is published, any client who is granted to access the model could be an adversary and is capable of performing arbitrary computation on model

parameters including some serious white-box attacks which have been reported in [7], [10], [11], [19]. We assume that the adversary has no prior information about the training dataset of the published DNN model. The adversary also has no idea how the model is obtained (by training from scratch or tuning on a pre-trained model). Meanwhile, there are two assumptions about the publisher. One is that the publisher is honest to users and should ensure the quality of the published model. The other one is that the publisher has unlimited computing resources. The publisher could train countless DNN models before it decides to publish any one of them.

III. Private Model Publishing

When a MLaaS provider shares a well-trained DNN model, it is critical to ensure that model privacy will not be disclosed. Rather than violating privacy, we find that statistical methods could also help preserve DNN model privacy. In this section, we will introduce a new private DNN model publishing solution which is composed of a statistical method and a differentially private mechanism. Furthermore, we will show that DNN models published with our solution give competitive model quality while privacy leakage is bounded within a fixed and acceptable privacy budget.

The privacy disclosed from a DNN model is mainly due to the precise fitting of private model. After learning sufficiently, parameters of a DNN model will contain enough information about training materials which makes privacy leakage possible. Hence, the most straightforward solution is perturbing model parameters to distort the mapping between training data and parameters directly. But this simple solution will cause notable damage to model quality. To tackle this problem, several elaborate solutions have been proposed recently, including noisy gradient based solutions [14], [20] and perturbed loss function based solution [21]. These solutions achieve privacy-preserving DNN model publishing by injecting artificial noise into model parameters continuously during the training phase. These explicit noise based solutions can give a strong privacy guarantee by sacrificing some model quality. However, there are also some drawbacks to this kind of solution. Explicit additional noise based solutions cannot guarantee the convergence of training, especially when high-level noise is used. Besides, the privacy budget of noise injection solutions increases along with training iterations dynamically. In other words, total privacy cost of explicit noise based solutions may exceed a fixed budget before training convergence.

To solve these problems, we propose an alternative privacy-preserving DNN model publishing solution, which is not based on explicit additional noise. The basic idea of our solution is to use the approximate estimation of each parameter to construct an artificial DNN model, which is illustrated in Figure 1. Since we estimate parameter distribution and sample representatives both in the parameter's stable state, there is no need to worry about training convergence. However, there are new challenges to be faced with. First, parameters of a well-trained DNN model can be seen as a combination of specific parameter states, which is supposed to be a local optimum (maybe the global optimum) in searching space. For privacy concerns, we cut off connections between parameters and treat each parameter separately. This will lead to some publishing models far away from local optimums. Second, without explicit noise injecting, we need another metric to replace the noise level to control the privacy budget. Third, both quality and privacy of the publishing model should be bounded.

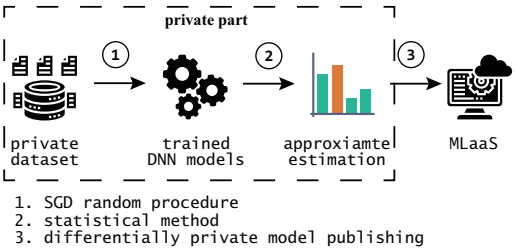

Fig. 1. Data flows of our basic idea.

A. Private Parameter Generating

We assume that a model set $\Pi = \{\pi_1, \pi_2, \ldots, \pi_M\}$ is constructed by training the same DNN architecture separately with the same training dataset. Assuming the total parameter amount in any model is N, we denote the set of all parameters in any model $\pi_i, i \in [1, M]$ by $\boldsymbol{\theta}^i$, $|\boldsymbol{\theta}^i| = N$. Although parameters of DNN model is organized in layers, we can simply flatten all parameters and treat them as a vector here, i.e. $\boldsymbol{\theta}^i = \{\theta_1^i, \theta_2^i, \ldots, \theta_N^i\}$. Then we define "parameter collection" Θ as a new dataset taking $\boldsymbol{\theta}^1, \boldsymbol{\theta}^2, \ldots, \boldsymbol{\theta}^M$ as its entries. Then any parameter θ_j^i located in position j in $\boldsymbol{\theta}^i$ of model π_i can be seen as an attribute of entry $\boldsymbol{\theta}^i, i \in [1, M]$.

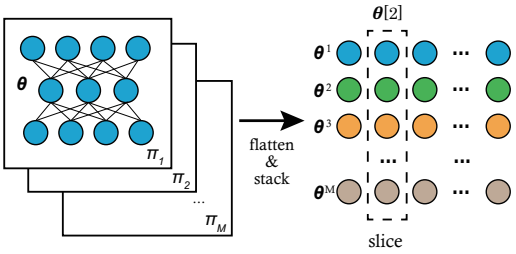

Fig. 2. Parameter collection and its slices.

As shown in Figure 2, if we slice Θ vertically, then we can get parameters located in the same position for all models. We will use $\boldsymbol{\theta}[j], j \in [1, N]$ to indicate vertical slices of Θ, which are elementary datasets that we are going to protect. If we treat DNN model publishing as a query of parameter collection, then publishing a single parameter θ_j can be seen as a query of slice $\boldsymbol{\theta}[j]$. When publishing a well-trained DNN model, all parameters can be seen published simultaneously. Hence, we will consider each parameter independently in this section. More complex discussion about non-independent parameters publishing will be given in the next section.

Generally, the publisher uses the mini-batch SGD method to optimize a DNN model with fixed hyperparameters. The optimizing process may vary for repetitive training tasks because of the random procedure of SGD. As long as the randomness can be ensured, it is almost impossible to reproduce an

accomplished SGD optimization process again. Hence, it is reasonable to say that $\boldsymbol{\theta}[j]$ has distinct diversity in multiple training tasks, for any $j \in [1, N]$. To be more intuitive, we show observations of three parameters in the final state for multiple training tasks in Figure 3. The diversity of parameter observations is significant. Since each slice $\boldsymbol{\theta}[j]$ of parameter collection Θ has such distinct diversity, we can take advantage of this characteristic to construct our private model publishing solution. Now we introduce some essential definitions.

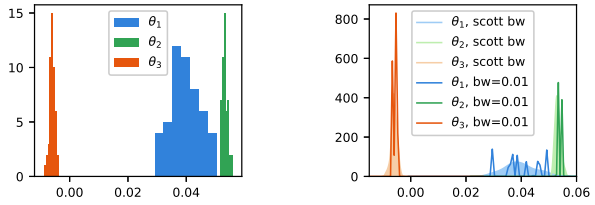

Fig. 3. Histogram (left) and corresponding estimation (right) of parameters.

- **Neighboring datasets** for client's query of each parameter are two neighboring slices of parameter collection, i.e. $\boldsymbol{\theta}[j]$ and $\boldsymbol{\theta}'[j]$ while two slices differ only on one element, i.e. $|\boldsymbol{\theta}[j]| = M, |\boldsymbol{\theta}'[j]| = M - 1$. The relation between training dataset and client's queried dataset is shown in Figure 1. As shown in the figure, parameter collection contains private information no less than any single DNN model. All information inside dashed line box in Figure 1 is private.
- **Query function** varies widely in differentially private mechanism designs. We give another new query function design for DNN model publishing here. Given parameter collection Θ, there is no need for clients to query the original training dataset since Θ has sufficient information to construct a full-functional DNN model. Instead, Θ will be queried by a composite function, which is composed of a statistical procedure f and a sampling procedure g. Specifically, we will use kernel density estimation (KDE) as function f and use exponential mechanism (EM) to construct g. When Θ is queried, each slice $\boldsymbol{\theta}[j]$ for any $j \in [1, N]$ is queried independently. Hence, we are actually going to construct $f_j(\boldsymbol{\theta}[j])$ and $g_j(f_j(\boldsymbol{\theta}[j]))$ for any $j \in [1, N]$ separately.
- **Model Privacy** of a specific DNN model can be defined in a differentially private manner. If a query result of $\boldsymbol{\theta}[j]$ for any $j \in [1, N]$ cannot tell the result is obtained from $\boldsymbol{\theta}[j]$ or its neighboring dataset $\boldsymbol{\theta}'[j]$, we say the model privacy regarding this query is preserved. Otherwise, we say the model privacy is disclosed. Please note this definition is different from data privacy definition in previous works which focus on specific data samples while model privacy focuses on model parameters. A more formal definition of model privacy with a privacy budget will be given in the next part.

Since elements in slice $\boldsymbol{\theta}[j], j \in [1, N]$ are collected from individual DNN models trained with the same DNN architecture and the same dataset, each element can be seen as a data sample drawn from some distribution. Based on this conjecture, we use KDE (proposed in [22]) to approximately estimate the distribution of elements in $\boldsymbol{\theta}[j]$. More specifically, we have estimator for elements in $\boldsymbol{\theta}[j]$ as

$$f_{j,b}(\boldsymbol{\theta}[j]) = \frac{1}{M \times b} \sum_{\theta \in \boldsymbol{\theta}[j]} \phi(\frac{\theta_j - \theta}{b}), \tag{4}$$

where b is a bandwidth (also known as smoothing parameter) of the estimator, ϕ is normal density function. Please note that the smoothing parameter b should be set empirically. Without causing any ambiguity, we will ignore subscript b in the rest. Having distribution of any slice $\boldsymbol{\theta}[j], j \in [1, N]$ in parameter collection Θ approximately estimated as $f_j(\boldsymbol{\theta}[j])$, the next step is to design a sampling procedure g to output parameters to construct a fully-functional DNN model. Since more precise parameters we sample, higher probability is to reveal model privacy. To solve this problem, we will use EM to construct our sampling procedure. We will also show that the KDE sampling approach can be perfectly integrated into EM.

To design an EM based private publishing solution, it is important to define a proper score function $u : \mathcal{R}^M \times \mathcal{R} \to \mathcal{R}$, mapping pairs of parameter collection slice and output parameter to real-valued scores. Since EM tries to output some element of $\mathcal{R}$ with the maximum possible score, we can give the formal definition of u by associating KDE result with output score. Then we have

$$u_j(\boldsymbol{\theta}[j], \theta_j) = \int_{\theta_j - \frac{\delta}{2}}^{\theta_j + \frac{\delta}{2}} f_j(\boldsymbol{\theta}[j]), \tag{5}$$

where δ is a small window for random sampling which could be optimized empirically. As proposed in [15], score function for EM can be arbitrarily sensitive in its range, which means the sensitivity of u_j should be defined as

$$\Delta_{u_j} = \max_{\theta_j \in \mathcal{R}} |u_j(\boldsymbol{\theta}[j], \theta_j) - u_j(\boldsymbol{\theta}'[j], \theta_j)|, j \in [1, N]. \tag{6}$$

For parameter $\theta_j, j \in [1, N]$, we set privacy budget ϵ_j. Then we will be ready to introduce the whole model publishing solution for the publisher.

1) The publisher performs multiple SGD optimizing tasks separately with the same dataset and the same DNN architecture to obtain M DNN models, composing Π.
2) Then construct parameter collection Θ by flattening and stacking parameters $\boldsymbol{\theta}^i$ of each DNN model in Π, $|\boldsymbol{\theta}^i| = N$, $i \in [1, M]$.
3) Slice Θ vertically to get isolated datasets $\boldsymbol{\theta}[j]$, consisting of parameter of all models in Π located in the same position, for all $j \in [1, N]$.
4) Perform KDE on each $\boldsymbol{\theta}[j]$ and obtain approximate estimation $f(\boldsymbol{\theta}[j])$ for the j-th parameter of DNN model.
5) Set a privacy budget ϵ_j for each $\boldsymbol{\theta}[j]$. Based on $f(\boldsymbol{\theta}[j])$ and $u(\boldsymbol{\theta}[j], \theta_j)$, generate each parameter θ_j with probability proportional to $\exp(\frac{\epsilon_j u(\boldsymbol{\theta}[j], \theta_j)}{2\Delta_{u_j}})$.
6) Squeeze all parameters $\boldsymbol{\theta} = \{\theta_j | j \in [1, N]\}$ into DNN layers and publish the model.
7) Set a quality threshold δ_d. Test the model to be published, if test accuracy is below the threshold δ_d, then

deny the model and go to Step 5. Otherwise, publish the model and terminate.

We summarize the above steps as differentially private parameter generating (DP-PG) algorithm and give its sketch in Algorithm 1. Keyword *Test* and *Sample* used in Algorithm 1 are functions which give test accuracy of a model and parameter samples following EM respectively.

B. Privacy and Quality Guarantees

Our main purpose is to keep DNN model privacy leakage within a limited privacy budget while providing high model quality. The private dataset directly accessed by any legal client should be Θ as shown in Figure 1. For any parameter θ_j, $j \in [1, N]$, we define a mechanism for the DP-PG algorithm as $M_{\text{DP-PG}}(\boldsymbol{\theta}[j], f_j, p_j) = \theta_j$. The corresponding privacy loss caused at output θ_j can be defined as

$$b(\theta_j; M_{\text{DP-PG}}, \boldsymbol{\theta}[j], \boldsymbol{\theta}'[j]) = \frac{\Pr[M_{\text{DP-PG}}(\boldsymbol{\theta}[j], f_j, p_j) = \theta_j]}{\Pr[M_{\text{DP-PG}}(\boldsymbol{\theta}'[j], f_j, p_j) = \theta_j]}.$$

Definition 3 (Model Privacy): For a model collection Π, given any neighboring datasets $\boldsymbol{\theta}[j]$ and $\boldsymbol{\theta}'[j]$ for any parameter θ_j, $j \in [1, N]$, if $b(\theta_j; M_{\text{DP-PG}}, \boldsymbol{\theta}[j], \boldsymbol{\theta}'[j])$ can be bounded by a fixed privacy budget, then the published model $\boldsymbol{\theta} = \{\theta_j | j \in [1, N]\}$ preserves model privacy under this privacy budget.

Now we will discuss how to determine the privacy budget of our DP-PG algorithm based publishing solution. By following the theorem of EM proposed in previous work [23], we can give the privacy guarantee of each parameter output of the DP-PG algorithm by proving mechanism $M_{\text{DP-PG}}$ preserves differential privacy.

Corollary 1: Given a score function $u : (\mathcal{R}^M \times \mathcal{R}) \to \mathcal{R}$, output parameter θ_j of the DP-PG algorithm is $(\epsilon_j, 0)$-DP, if θ_j is chosen with probability proportional to $\exp(\frac{\epsilon_j u(\boldsymbol{\theta}[j], \theta_j)}{2\Delta_{u_j}})$, for any $j \in [1, N]$.

Proof: Bounding the sensitivity of query function is crucial for determining privacy loss. Now we will prove that sensitivity Δ_{u_j} is always within $[0, 1]$ for any neighboring datasets. Recall that $\forall j \in [1, N]$,

$$\begin{aligned}
\Delta_{u_j} &= \max_{\theta_j \in \mathcal{R}} \max_{\boldsymbol{\theta}[j], \boldsymbol{\theta}'[j]} |u_j(\boldsymbol{\theta}[j], \theta_j) - u_j(\boldsymbol{\theta}'[j], \theta_j)| \\
&= \max_{\theta_j \in \mathcal{R}} \max_{\boldsymbol{\theta}[j], \boldsymbol{\theta}'[j]} |\int_{\theta_j - \frac{\delta}{2}}^{\theta_j + \frac{\delta}{2}} f_j(\boldsymbol{\theta}[j]) - \int_{\theta_j - \frac{\delta}{2}}^{\theta_j + \frac{\delta}{2}} f_j(\boldsymbol{\theta}'[j])| \\
&= \max_{\theta_j \in \mathcal{R}} \max_{\boldsymbol{\theta}[j], \boldsymbol{\theta}'[j]} |\int_{\theta_j - \frac{\delta}{2}}^{\theta_j + \frac{\delta}{2}} \frac{1}{M \times b} \sum_{\theta \in \boldsymbol{\theta}[j]} \Phi(\frac{\theta_j - \theta}{b}) - \\
&\quad \frac{1}{(M-1) \times b} \sum_{\theta \in \boldsymbol{\theta}'[j]} \Phi(\frac{\theta_j - \theta}{b})| \\
&< \max_{\theta_j \in \mathcal{R}} \max_{\theta_0 \in \boldsymbol{\theta}[j], \notin \boldsymbol{\theta}'[j]} |\int_{\theta_0 - \frac{\delta}{2}}^{\theta_0 + \frac{\delta}{2}} \frac{1}{M \times (M-1) \times b}| \\
&< 1.
\end{aligned}$$

Actually, if we choose $\delta < b$, we can have $\Delta_{u_j} < \frac{1}{M \times (M-1)}$. Since Δ_{u_j} is no more than 1, we can say that $M_{\text{DP-PG}}$ preserves $(\epsilon_j, 0)$-differential privacy as long as we choose θ_j with probability proportional to $\exp(\frac{\epsilon u(\boldsymbol{\theta}[j], \theta_j)}{2\Delta_{u_j}})$, for any fixed privacy budget ϵ_j. For the whole publishing DNN model, we have N parameters in total. According to the parallel composition theorem proposed in [15], we can conclude that the privacy budget of publishing $\boldsymbol{\theta}$ is $\max\{\epsilon_1, \epsilon_2, \ldots, \epsilon_N\}$. ■

Meanwhile, the EM is supposed to give a strong utility guarantee because the output decreases exponentially while the quality score falls off [15]. In our DP-PG algorithm, we treat each parameter in $\boldsymbol{\theta}$ as an agent and design independent publishing mechanism respectively. For slice $\boldsymbol{\theta}[j]$, let $\text{OPT}_u(\theta_j) = \max_{\theta \in \mathcal{R}} u(\boldsymbol{\theta}[j], \theta_j)$ denote the maximum utility score of any possible $\theta_j \in \mathcal{R}$. We can measure the utility of parameter θ_j in terms of $\text{OPT}_u(\theta_j)$. Assume that the sampling interval of KDE output distribution is β, which means any two neighboring samples on estimated distribution have β spatial distance. Then range $\mathcal{R}$ can be quantized. To ensure range $\mathcal{R}$ finite in the analysis, we assume long tails of the estimated distribution are truncated and area within $(\text{MAX}_{\theta_j}, \text{MIN}_{\theta_j})$ will be kept. This will lead to $|\mathcal{R}| = (\text{MAX}_{\theta_j} - \text{MIN}_{\theta_j})/\beta$. Based on this result, we can bound parameter utility of the DP-PG algorithm by following theorem and corollary about EM proposed in [15] directly. Please note that proof of our corollary is straightforward and will be omitted here.

Corollary 2: If $M_{\text{DP-PG}}(\boldsymbol{\theta}[j], f_j, p_j)$ outcomes θ_j with probability proportional to $\exp(\frac{\epsilon u(\boldsymbol{\theta}[j], \theta_j)}{2\Delta_{u_j}})$, published parameter utility of θ_j can be bounded by $\Pr[u(M_{\text{DP-PG}}(\boldsymbol{\theta}[j], f_j, p_j)) \leq \text{OPT}_u(\theta_j) - \frac{2\Delta_u}{\epsilon_j}(ln(|\mathcal{R}|) + t)] \leq e^{-t}$, $\forall \theta_j \in \boldsymbol{\theta}$.

Algorithm 1 Differentially Private Parameter Generating (DP-PG) Algorithm

```
initial θ = 0;
while Test(θ) ≤ δ_d do
    for j = 1 to N do
        f_b(θ_j) = KDE(θ[j]);
        u_j(θ[j], θ_j) = ∫_{θ_j − δ_g/2}^{θ_j + δ_g/2} f_j(θ[j]);
        θ_j = Sample(exp(ε_j u_j(θ[j], θ_j) / (2Δ_{u_j})));
    end for
end while
```

IV. Model Publishing with Parameters Grouping

In our basic publishing solution, we handle each parameter of a DNN model separately, which poses a threat to the stability of the published model. One immediate result is that the DNN model quality will be frustrated occasionally because connections between parameters are broken. In this section, we will show that this circumstance can be relieved by relaxing the parameter assumption of separate generating. It has been reported that DNN model parameters have potential connections and different levels of importance in design space [24], [25]. Based on this observation, we propose a hybrid publishing solution in this section, which combines our DP-PG algorithm with selective parameters grouping. The core idea is to preserve parameter connections within a selected portion of the DNN model while preserving model privacy. Parameters outside the selected portion will be published using our original solution. In the rest of this section, we will show how to deal with those selected parameters.

A. Private Parameters Grouping

Inspired by previous work on DNN model compression [26], we will do parameter grouping to preserve parameter connections within each DNN layer. Assume a specific DNN model π_0 to be published consisting of L layers. All parameters in the l-th layer are denoted as $\boldsymbol{\theta}_{(l)}$, $l \in [1, L]$. If we flatten parameters of the l-th layer into a vector, then we can denote the i-th parameter of $\boldsymbol{\theta}_{(l)}$ as $\theta_{(l,i)}$, $i \in [1, n_l]$ where n_l is parameter amount of the l-th layer. For the l-th layer, we sort $\boldsymbol{\theta}_{(l)}$ by parameter significance and get result $\boldsymbol{\theta}'_{(l)}$ in descending order. A predefined selection ratio γ_l will be used to control the size of parameter selection. Specifically, parameters stored in top γ_l of $\boldsymbol{\theta}'_{(l)}$ will be selected for grouping. Other parameters will be dealt with the DP-PG algorithm and published separately.

To capture connections between parameters in the selected portion $\boldsymbol{\theta}_{(l)sel} = \{\theta_i | 1 \leq i \leq \gamma_l n_l]\}$, clustering algorithm will be used to group parameters by euclidean distance. Meanwhile, in order to preserve the privacy of clustering result, a private k-means clustering approach [27] will be used. After running private k-means algorithm on $\boldsymbol{\theta}_{(l)sel}$, k clustering centroids $C = \{c_1, c_2, \ldots, c_k\}$ will be given. Parameters grouped into the same cluster can be represented by the corresponding cluster centroid. Although parameters connection can be preserved in this way, their unique characteristics will be destroyed, which will lead to the loss of model quality. To tackle this problem, we propose another differentially private mechanism upon private k-means to publish tuned parameter values for those parameters in the same group.

Generally, a cluster denoted by $K^i = \{\theta_1^{K^i}, \theta_2^{K^i}, \ldots, \theta_{|K^i|}^{K^i}\}$ has clustering centroid c_i, where $K^i \subset \boldsymbol{\theta}_{(l)sel}$, for $i \in [1, k]$. For any $\theta_j^{K^i} \in K^i$, we design an obfuscated distance query function $o_i : \mathcal{R}^{|K^i|} \to \mathcal{R}$,

$$o_i(\theta_j^{K^i}) = c_i - \theta_j^{K^i} + Lap(\frac{\Delta_{o_i}}{\epsilon_{K^i}}), \tag{7}$$

where ϵ_{K^i} is the privacy budget of distance query for cluster K^i, Δ_{o_i} is l_1-sensitivity of query function o_i regarding neighboring datasets in $\mathcal{R}^{|K^i|}$ (say K^{i1}, K^{i2}). Then we have

$$\Delta_{o_i} = \max_{||K^{i1} - K^{i2}||_1 = 1} ||o_i(\theta_j^{K^{i1}}) - o_i(\theta_j^{K^{i2}})||_1. \tag{8}$$

We summarize this differentially private parameter grouping and publishing (DP-PGP) solution in Algorithm 2. Please note that parameter connections' preserving is controlled by selection ratio γ. When $\gamma = 0$ for all of the layers, DP-PGP algorithm will be equivalent to DP-PG algorithm.

B. Privacy Analysis

The main difference between DP-PG algorithm and DP-PGP algorithm is parameter grouping. Since only the selected parameters will be grouped, the selection ratio is important to privacy analysis. For the l-th layer, $l \in [1, L]$, no parameters will be selected if we set $\gamma_l = 0$, which means there is no parameter to be grouped. In this case, the DP-PGP algorithm will be equivalent to the DP-PG algorithm while publishing the l-th layer.

Algorithm 2 Differentially Private Parameter Grouping and Publishing (DP-PGP) Algorithm.

initial $\boldsymbol{\theta} = 0$;
while $Test(\boldsymbol{\theta}) \leq \delta_d$ **do**
  for $i = 1$ to N **do**
    $f_b(\theta_i) = KDE(\boldsymbol{\theta}[i])$;
    $u_i(\boldsymbol{\theta}[i], \theta_i) = \int_{\theta_i - \frac{\delta_g}{2}}^{\theta_i + \frac{\delta_g}{2}} f_i(\boldsymbol{\theta}[i])$;
  end for
  for model π_0:
  for $l = 1$ to L **do**
    $\boldsymbol{\theta}'_{(l)}$ = sort $\boldsymbol{\theta}_{(l)}$ in descenting order;
  end for
  for $i = 1$ to n_l **do**
    if $i \leq \gamma_l \times n_l$ **then**
      $\boldsymbol{\theta}_{(l)sel} = \boldsymbol{\theta}_{(l)sel} + \theta_i$;
    else
      $\theta_i = Sample(\exp(\frac{\epsilon u_i(\boldsymbol{\theta}[i], \theta_i)}{2\Delta_{u_i}}))$;
    end if
  end for
  $(\boldsymbol{K}, C) = DP - Kmeans(\boldsymbol{\theta}_{(l)sel})$;
  for $i = 1$ to k **do**
    for $j = 1$ to $|K^i|$ **do**
      $\theta_j = c_i + (c_i - \theta_j^{K^i}) + Lap(\frac{\Delta_{o_i}}{\epsilon_{K^i}})$;
    end for
  end for
end while

Corollary 3: If $\gamma_l = 0, l \in [1, L]$, the DP-PGP algorithm will be equivalent to the DP-PG algorithm.

For any $l \in [1, L]$, if $0 < \gamma_l \leq 1$, then parameter grouping will happen. Two parts of private parameter grouping may disclose private information, i.e. clustering and tuning. In the clustering phase, total privacy leakage caused by clustering algorithm may vary from the specific privacy-preserving clustering implementations. For conciseness, we will simply denote total privacy leakage caused by private clustering by ϵ_c and treat ϵ_c as a constant in the rest. In the tuning phase, parameters within each cluster will be handled by an independent Laplace mechanism. Given a privacy budget ϵ_{K^i} for parameters $\theta_j, j \in [1, |K^i|]$ in K^i, total privacy budget of tuning grouped parameters will be the maximal privacy budget across all clusters, i.e. $\max_{i=1}^{k} \epsilon_{K^i}$. For the parameters not selected for grouping, we can give privacy budget for each of them by following the corollary of DP-PG algorithm directly, which is $\max_{\theta_j \in \boldsymbol{\theta}_{(l)} - \boldsymbol{\theta}_{(l)sel}} \epsilon_j$. Now we can give total privacy budget of the l-th layer as

$$b(\boldsymbol{\theta}_{(l)}) = \max\{(\epsilon_c + \max_{i \in [1,k]} \epsilon_{K^i}), \max_{\theta_j \in \boldsymbol{\theta}_{(l)} - \boldsymbol{\theta}_{(l)sel}} \epsilon_j\}. \tag{9}$$

Taking all layers of a DNN model into account, we can conclude a generalized case of the first corollary.

Corollary 4: For a DNN model consisting of L layers, total privacy budget of the DP-PGP algorithm will be $\max_{l=1}^{k} b(\boldsymbol{\theta}_{(l)})$ while publishing the entire model.

V. Evaluation

We implement our solutions in python with Keras using TensorFlow backend. All experiments are performed on a server with 64 cores Intel(R) Xeon(R) CPU Gold 5122 @ 3.60 GHz and 2 NVIDIA GeForce RTX2080i graphics cards. We evaluate our solutions on two popular DNN architectures in practical uses which are defined and released in Keras source code. According to the training dataset of each DNN, we will call these two DNN *MNIST-Net* and *CIFAR-Net*. MNIST-Net consists of two convolutional layers and two fully connected layers, with 1,199,882 parameters in total. CIFAR-Net consists of four convolutional layers and two fully connected layers, with 1,250,858 parameters in total. More details about DNN architectures can be found in Keras source code.

The datasets we use for training are MNIST and CIFAR-10 datasets respectively. MNIST dataset [28] is a standard handwritten digits dataset including numbers from 0 to 9. CIFAR-10 dataset [29] is a popular image classification dataset consisting of 50000 training images and 10000 test images for 10 classes. To construct parameter collection for our solutions, we train this DNN for 50 times to get 50 different models and corresponding intermediate results. Each model is trained for at least 60 epochs with SGD optimizer to have accuracy above 99% without any privacy protection. All hyperparameters are the same for 50 times training. The batch size is 128 and the learning rate is 0.001. The baseline models are also trained in this setting.

A. Model Quality Evaluation

We first evaluate model quality of DP-PG and DP-PGP solutions. To give a more thorough study on private model quality, we compare our solutions with DP-SGD [14], which has been a state-of-art solution for private DNN learning. We also give a comparison with a baseline case which is not protected by any private solution. As for performance evaluation of our solutions, we will take two most important factors into account, training epoch and privacy budget. Parameter collections used for the DP-PGP and DP-PG are the same in all experiments in quality evaluation.

Training epoch. We evaluate our solutions by publishing models in different training epochs. Since DP-PGP is based on DP-PG, we find they have very similar performance on training epoch. So we will mainly show results of DP-PG algorithm here and the results of DP-PGP will be given in the next part. As shown in Figure 4 and Figure 5, we use three fixed privacy budgets to evaluate DP-PG solution while comparing with DP-SGD and the baseline model. DP-PG can achieve commensurate result on training and test metrics compared with baseline in all stages. When the privacy budget is large, publishing model will benefit from model aggregation to achieve a better performance than a single DNN model. This phenomenon has been studied in recent research work [30]. Since DP-SGD uses a dynamic privacy budget, we cannot fix it like our solutions. So we let it increase arbitrarily. For MNIST dataset, the privacy budget may exceed 8 for DP-SGD. But its test accuracy will not exceed 96%. For CIFAR-10 dataset, the privacy budget of DP-SGD may exceed 3 after 50 epochs. But its accuracy cannot be better than DP-PG with $\epsilon = 1$. Please note that quality results have some unexpected fluctuations because we show experiment results of a random round for general case instead of an averaged result across multiple rounds or a selected result of the best round.

Privacy budget. To demonstrate relationship between model quality and privacy budget, we generate DNN models in well-trained states with various epsilon values. But DP-SGD has a narrow privacy budget range which may lead to an epsilon value larger than 1 in the first epoch. As shown in Figure 6, our DP-PG solution introduces significantly less interference in model quality when compared with DP-SGD for different epsilon values. The model quality is affected by a small privacy budget (e.g. $\epsilon \leq 0.1$) of DP-PG, which is rather conservative for practical use. According to this result, we recommend to setting $\epsilon \leq 1$ for MNIST-Net for a good balance between privacy and quality. For CIFAR-Net, the privacy budget will significantly increase for acceptable accuracy because training data within this dataset has less similarity than MNIST. Even though, DP-PG can approximate the baseline when $\epsilon \geq 2$.

When we evaluate the relationship between model quality and privacy budget for the DP-PGP solution, privacy budgets of DP-Kmeans and Laplace mechanism should be taken into account. We assume an averaged DNN model in the well-trained state is the target model to be published for DP-PGP and we will use it as the baseline. It is complicated to measure published model quality for all possible privacy budgets of DP-PGP because there are three different privacy budgets of its components respectively. To be succinct, we set $\epsilon_j = \epsilon_1$, $\epsilon_c = \epsilon_2$, $\forall \theta_j \in \boldsymbol{\theta_{(l)}} - \boldsymbol{\theta_{(l)sel}}$, $\forall l \in [1, L]$ for all layers' DP clustering process, $\epsilon_{K^i} = \epsilon_3$ for all clusters, $i \in [1, k]$. Grouping selection ratio is 0.2. Quality evaluation of MNIST-Net regarding ϵ_1, ϵ_2, and ϵ_3 are shown in Figure 8, Figure 9 and Figure 10 respectively.

The privacy budget of EM has a small influence on model quality for DP-PGP because it has been proved to be sufficiently small for an acceptable accuracy in DP-PG evaluation. So there is no significant trend regarding ϵ_1 in Figure 8. As shown in Figure 9 and Figure 10, the privacy budgets of DP-Kmeans and Laplace mechanism have main influences on model quality for DP-PGP solution. Specifically, when $\epsilon \geq 2$, the model quality can be commensurate with the baseline. It should be noted that if some DP-Kmeans method with a smaller error is applied in DP-PGP, the privacy budget can be smaller than this result. If $\epsilon_2 \geq 1$, $\epsilon_2 \geq 1$, Laplace mechanism will cause the main effect on model quality for the DP-PGP solution. In this case, the privacy budget of the Laplace mechanism should be at least 1.5 to achieve an acceptable test accuracy.

We also evaluate privacy budgets of all components of DP-PGP with CIFAR-10 dataset. We find that the privacy budgets of EM and DP-Kmeans have a nearly negligible influence on model quality when compared with the privacy budget of the Laplace mechanism. Since the Laplace mechanism seems the

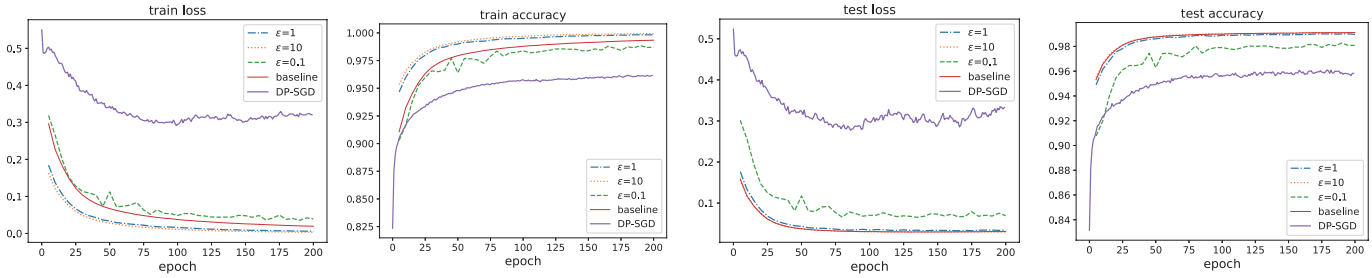

Fig. 4. Model quality of MNIST-Net with fixed privacy budgets in different training stages (DP-PG).

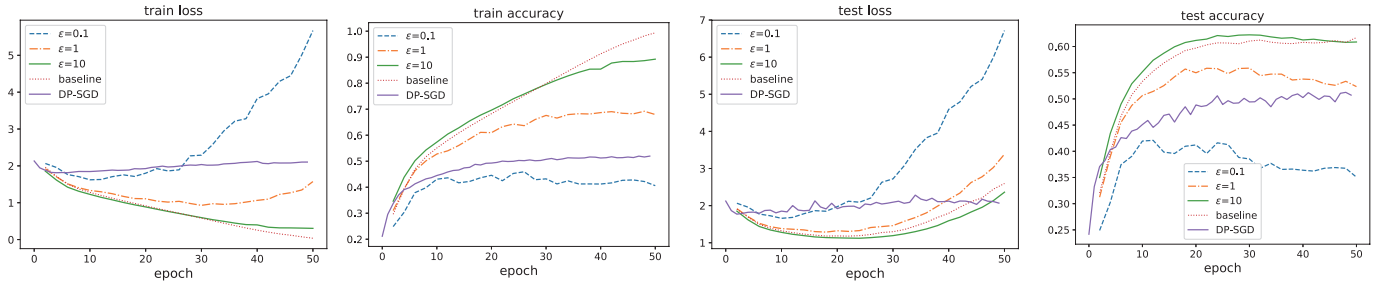

Fig. 5. Model quality of CIFAR-Net with fixed privacy budgets in different training stages (DP-PG).

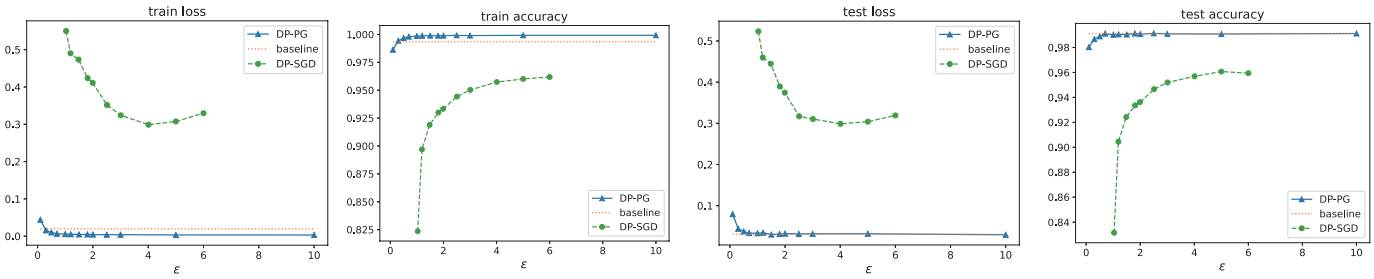

Fig. 6. Model quality of MNIST-Net with various privacy budgets (DP-PG).

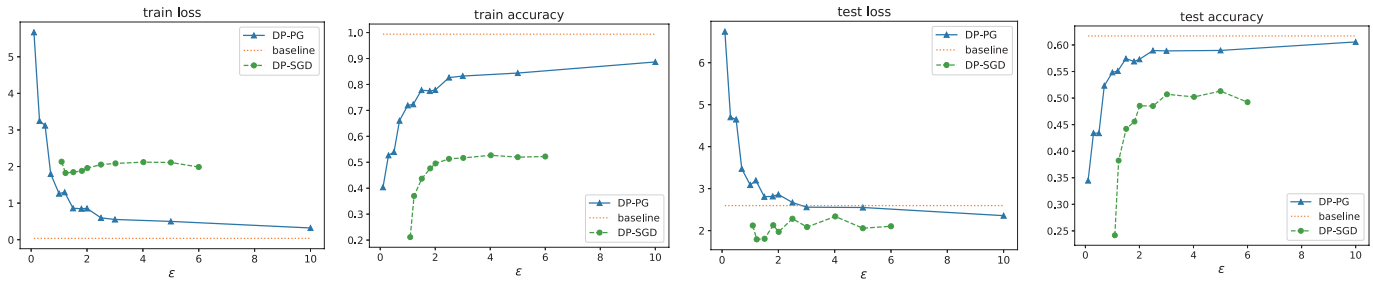

Fig. 7. Model quality of CIFAR-Net with various privacy budgets (DP-PG).

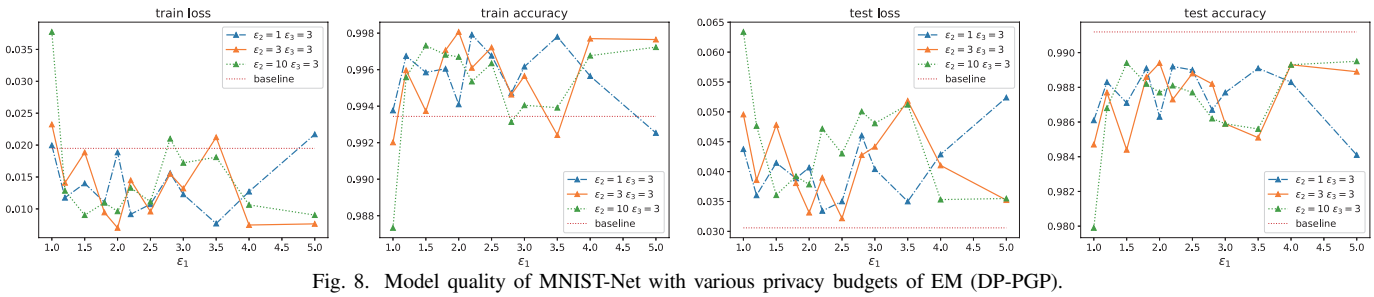

Fig. 8. Model quality of MNIST-Net with various privacy budgets of EM (DP-PGP).

most important part of DP-PGP with CIFAR-Net, we give ϵ_3 result here and skip results of ϵ_1 and ϵ_2 due to page limit.

As shown in Figure 11, the privacy budget of DP-Kmeans is restricted to be small, i.e. $\epsilon_2 = 1$ for the more conspicuous

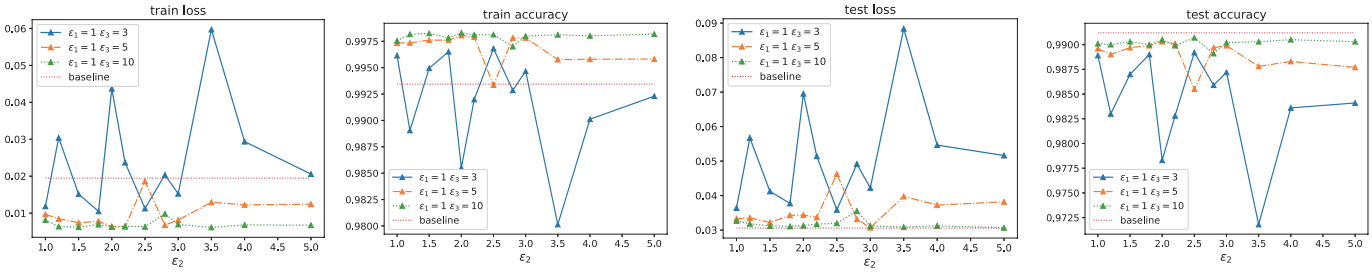

Fig. 9. Model quality of MNIST-Net with various privacy budgets of DP-Kmeans (DP-PGP).

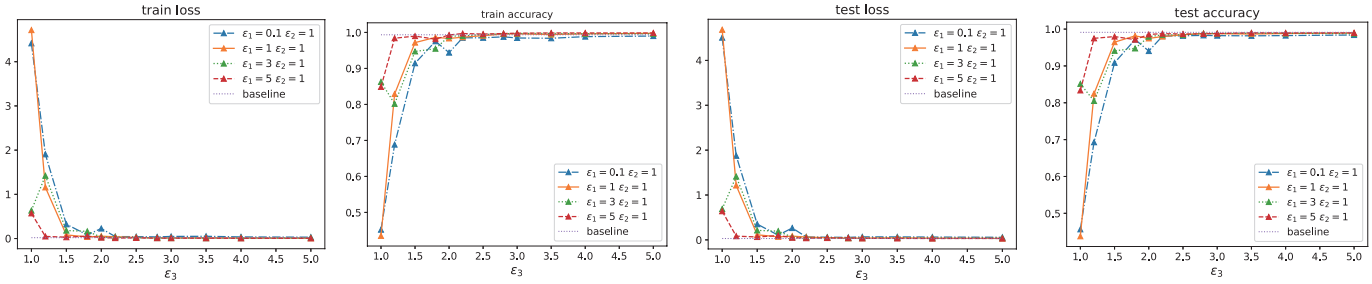

Fig. 10. Model quality of MNIST-Net with various privacy budgets of Laplace mechanism (DP-PGP).

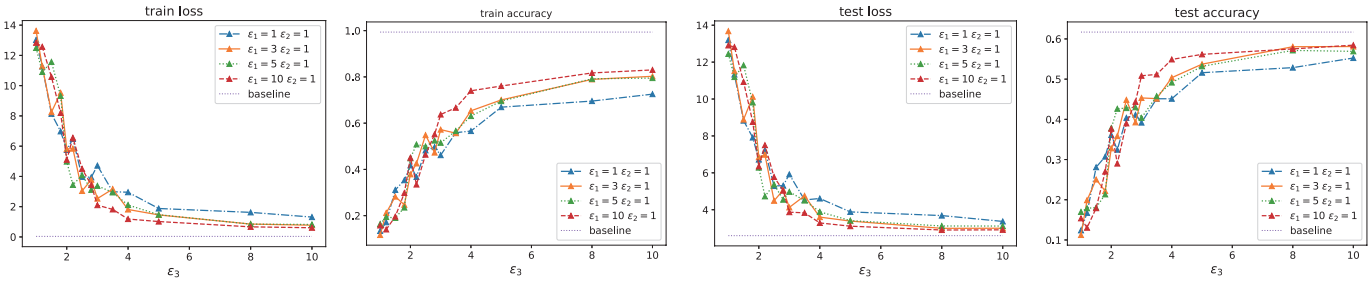

Fig. 11. Model quality of CIFAR-Net with various privacy budgets of Laplace mechanism (DP-PGP).

effect of ϵ_3. In this case, ϵ_3 must be larger than 3 to achieve acceptable model quality on CIFAR-10 dataset. But the privacy budget of the Laplace mechanism can be relaxed if we give more privacy budget for the DP-Kmeans method. This result also confirms the correctness of the total privacy budget of DP-PGP, which combines the privacy budgets of DP-Kmeans and Laplace mechanism as an expression, where they share a total privacy budget together.

B. Privacy Evaluation

Membership inference attack has been proved effective for published DNN models in [7], [31]. To verify privacy preserving performance of our solutions, we perform a membership inference attack against published models with DP-PG and DP-PGP. We also evaluate this attack against DNN models with DP-SGD and the baseline model for comparison. Since the membership inference attack achieves its best performance on CIFAR-10 dataset [31], we will do all privacy evaluations on CIFAR-10 dataset. Also, as reported in [7], the inference accuracy of the attack largely depends on the over-fitting of the DNN model, we will publish DNN models with our solutions in over-fitting states, which will result in a frustrated test accuracy.

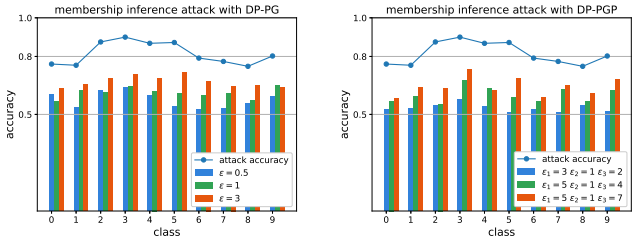

Fig. 12. Membership inference attack against published models.

We show the inference attack accuracy[1] and test accuracy of model published with DP-PG and DP-PGP solutions in Figure 12. When the adversary attacks against benign DNN model without any privacy protection, inference accuracy varies along with different classes. In the best case of an over-fitting model, the inference accuracy almost reaches 90% on class label 3. However, when we apply our DP-PG for model publishing with $\epsilon = 1$, the adversary can reach an inference accuracy around 57% on all classes, which is nearly random guessing. According to a recent study [31] on membership inference

[1]Please note that the membership inference attack accuracy and DP-SGD protection effects reported here may have differences with original works. Because we use different hyperparameters and settings. But we will use fixed settings and hyperparameters strictly for all attack experiments.

attack against DP-SGD, when model test accuracy is 60.7% on CIFAR-10 dataset, attack accuracy can be almost 70% on two classes against DP-SGD. When model test accuracy is 45%, attack accuracy can achieve higher than 60% on one class against DP-SGD. However, DP-PG can get test accuracy higher than 55% when attack accuracy is lower than 60% on all classes. The attack against model published with DP-PGP may get better accuracy because DP-PGP gives more private information of parameters to trade for better model quality. When DP-PGP uses a total privacy budget 5 ($\epsilon_1 = 5, \epsilon_2 = 1, \epsilon_3 = 4$), attack accuracy is no more than 70% on any class while test accuracy is about 50% on CIFAR-10 dataset.

VI. Conclusion

In this paper, we report an observation about DNN model training, which shows that parameters in well-trained states have similar patterns for separate training tasks even with randomness involved. Based on this observation, we propose DP-PG, a DNN model parameter generating algorithm for sharing well-trained DNN models with model privacy preserved. But model quality of DP-PG will be limited by privacy budget because parameter connections are broken by DP-PG solution. To moderate this situation, we give another solution DP-PGP, which can get higher model quality when the privacy budget increases. DP-PGP uses different privacy budget calculations from DP-PG. Hence, it will give a higher total privacy budget than DP-PG in most cases. We have borrowed some ideas from DNN model compression when we design DP-PGP. We find it an interesting way to combine these two topics together. We will keep studying on these topics in our future work. Meanwhile, we will also try some more powerful tools like autoencoder to give a more accurate estimation of DNN model parameters.

Acknowledgement

The authors would like to thank all the reviewers for their helpful comments. This work was supported in part by BK20190294, NSFC-61902176, 61872179, 61425024, 61872176 and National Key R&D Program of China (2018YFB1004301).

References

[1] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang, "The expressive power of neural networks: A view from the width," in *Advances in neural information processing systems*, 2017, pp. 6231–6239.

[2] Y. You, A. Buluç, and J. Demmel, "Scaling deep learning on gpu and knights landing clusters," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2017, pp. 1–12.

[3] BVLC, "Caffe model zoo," https://github.com/BVLC/caffe/wiki/Model-Zoo.

[4] PyTorch, "Pytorch hub," https://pytorch.org/hub/research-models.

[5] TensorFlow, "Tensorflow models," https://github.com/tensorflow/models.

[6] J. Zhang, Z. Gu, J. Jang, H. Wu, M. P. Stoecklin, H. Huang, and I. Molloy, "Protecting intellectual property of deep neural networks with watermarking," in *Proceedings of the 2018 AsiaCCS*, 2018, pp. 159–172.

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

[8] M. Nasr, R. Shokri, and A. Houmansadr, "Machine learning with membership privacy using adversarial regularization," in *Proceedings of the 2018 ACM SIGSAC CCS*, 2018, pp. 634–646.

[9] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *2019 IEEE Symposium on Security and Privacy*, 2019, pp. 691–706.

[10] K. Ganju, Q. Wang, W. Yang, C. A. Gunter, and N. Borisov, "Property inference attacks on fully connected neural networks using permutation invariant representations," in *Proceedings of the 2018 ACM SIGSAC CCS*, 2018, pp. 619–633.

[11] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22nd ACM SIGSAC CCS*, 2015, pp. 1322–1333.

[12] B. Hitaj, G. Ateniese, and F. Pérez-Cruz, "Deep models under the gan: information leakage from collaborative deep learning," in *Proceedings of the 2017 ACM SIGSAC CCS*, 2017, pp. 603–618.

[13] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC CCS*, 2017, pp. 1175–1191.

[14] 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 2016 ACM SIGSAC CCS*, 2016, pp. 308–318.

[15] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, pp. 211–407, 2014.

[16] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of cryptography conference*, 2006, pp. 265–284.

[17] Z. Huang and S. Kannan, "The exponential mechanism for social welfare: Private, truthful, and nearly optimal," in *2012 IEEE FOCS*, 2012, pp. 140–149.

[18] F. McSherry and K. Talwar, "Mechanism design via differential privacy." in *2007 IEEE FOCS*, vol. 7, 2007, pp. 94–103.

[19] Z. Yang, J. Zhang, E.-C. Chang, and Z. Liang, "Neural network inversion in adversarial setting via background knowledge alignment," in *Proceedings of the 2019 ACM SIGSAC CCS*, 2019, pp. 225–240.

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

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

[22] S. J. Sheather and M. C. Jones, "A reliable data-based bandwidth selection method for kernel density estimation," *Journal of the Royal Statistical Society: Series B*, pp. 683–690, 1991.

[23] K. Talwar and F. McSherry, "Mechanism design via differential privacy," in *2007 IEEE FOCS*, 2007, pp. 94–103.

[24] Y. LeCun, J. S. Denker, and S. A. Solla, "Optimal brain damage," in *Advances in neural information processing systems*, 1990, pp. 598–605.

[25] S. Han, J. Pool, J. Tran, and W. J. Dally, "Learning both weights and connections for efficient neural networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems*, 2015, pp. 1135–1143.

[26] T. Li, B. Wu, Y. Yang, Y. Fan, Y. Zhang, and W. Liu, "Compressing convolutional neural networks via factorized convolutional filters," in *IEEE CVPR*, 2019, pp. 3977–3986.

[27] D. Su, J. Cao, N. Li, E. Bertino, and H. Jin, "Differentially private k-means clustering," in *Proceedings of the sixth ACM conference on data and application security and privacy*, 2016, pp. 26–37.

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

[29] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Citeseer, Tech. Rep., 2009.

[30] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent," in *Advances in Neural Information Processing Systems*, 2017, pp. 5330–5340.

[31] M. A. Rahman, T. Rahman, R. Laganière, N. Mohammed, and Y. Wang, "Membership inference attack against differentially private deep learning model." *Transactions on Data Privacy*, pp. 61–79, 2018.