Differentially-Private Deep Learning from an Optimization Perspective

Liyao Xiang[1,2], Jingbo Yang[2], and Baochun Li[2]

[1]John Hopcroft Center for Computer Science, Shanghai Jiao Tong University
[2]Department of Electrical and Computer Engineering, University of Toronto

***Abstract*—With the amount of user data crowdsourced for data mining dramatically increasing, there is an urgent need to protect the privacy of individuals. Differential privacy mechanisms are conventionally adopted to add noise to the user data, so that an adversary is not able to gain any additional knowledge about individuals participating in the crowdsourcing, by inferring from the learned model. However, such protection is usually achieved with significantly degraded learning results. We have observed that the fundamental cause of this problem is that the relationship between model utility and data privacy is not accurately characterized, leading to privacy constraints that are overly strict. In this paper, we address this problem from an optimization perspective, and formulate the problem as one that minimizes the accuracy loss given a set of privacy constraints. We use sensitivity to describe the impact of perturbation noise to the model utility, and propose a new optimized additive noise mechanism that improves overall learning accuracy while conforming to individual privacy constraints. As a highlight of our privacy mechanism, it is highly robust in the high privacy regime (when $\epsilon \to 0$), and against any changes in the model structure and experimental settings.**

***Index Terms*—crowdsourcing, data mining, differential privacy, deep learning, optimization**

I. Introduction

A recent proliferation of machine learning has empowered many data mining applications, such as image recognition and product recommendation. Individual data, such as medical records, personal images, locations, or social media data, are crowdsourced by third-party authorities such as data miners. Unfortunately, the user data may be sensitive, and can easily fall prey to adversarial attacks. For instance, with a membership inference attack [1] or a user-linkage attack, an adversary can infer whether an individual participates in the crowdsourcing, even if that individual remains anonymous.

Many recent works have proposed various schemes to provide users differential privacy guarantee in mining crowdsourced data to perform a wide range of tasks such as social media data outsourcing [2], demand reporting [3], spatio-temporal data publishing [4], or tree-based mining [5]. Differential privacy [6] has been proposed as a mathematical concept, as well as a privacy-preserving mechanism to constrain the adversary's power to infer about an individual with any outside information.

In this paper, we focus on the privacy issue in general machine learning tasks, especially where deep learning is concerned, as they are gaining popularity as analysis tools these days. Usually, a trusted third-party authority, such as Amazon machine learning service, collects private data from each individual, trains a model over these data, and eventually publishes the model for use. Differential privacy mechanisms can be naturally extended to machine learning models. For example, *parameter perturbation* [7], [8] perturb the model parameters with privacy constraints to prevent the white-box inference about any individual participated in the training.

In our threat model, we assume that the adversary has *white-box* access to the learning model on the authority, possesses arbitrary auxiliary information, and the privacy mechanism is publicly known. The problem is that, when differential privacy mechanisms add noise to model parameters, the utility of the learning result degrades as the privacy requirement becomes more stringent. For example, Abadi *et al.* [8] are only able to achieve an accuracy of 90% with the MNIST dataset, whereas the accuracy with an unperturbed model can reach 99%. While Abadi *et al.* [8], by using the moments accountant, have already improved the utility significantly over strong composition [7], the performance is still far inferior to unperturbed learning, especially in the regime where the privacy requirement is stringent.

Fundamentally, such a wide performance gap can be attributed to the lack of a rigorous characterization of the relationship between the model utility and privacy constraints. In previous work, noise is added to the model parameters without considering its impact on the learning accuracy.

We have observed that model parameters can be perturbed in carefully optimized directions such that the model accuracy loss is minimized. By intentionally adding more noise to the parameters that have less impact on the output, we can achieve a higher accuracy while satisfying the differential privacy constraints. With an approximation of the accuracy loss, we formulate the challenge of designing the optimal differential privacy mechanism for deep learning models as an optimization problem, which searches for a probability density function (pdf) of the perturbation noise to minimize a weighted model distortion under differential privacy constraints.

However, such an optimization problem is non-trivial to solve, due to the high-dimensional parameter space of deep learning models. From a practical perspective, we propose to reach a compromise between mathematical complexity and runtime efficiency, and design a new differential privacy mechanism with a focus on its efficient implementation.

Highlights of our original contributions in this paper are as follows. We have introduced an optimized additive noise mechanism to minimize the perturbation impact on the model while satisfying differential privacy constraints. Our privacy mechanism is implemented as a building block to the privacy-preserving learning framework and is tuned to run efficiently. Experiments on a variety of learning models and datasets have shown that, our mechanism is able to significantly improve the learning accuracy over the state-of-the-art while protecting individual privacy.

II. Related Work

Models trained over sensitive data can be a significant threat to the privacy of such data [1], [9]. In the context of differential privacy, a number of algorithms have been proposed.

Following the principle of differentially-private stochastic gradient descent [10], Shokri *et al.* [7] let participants train their own datasets privately and selectively share small subsets of their models' key parameters. Even that a small percentage (< 0.1) of the parameters are perturbed and shared, their composition method still consumes a large amount of the privacy budget, which is way beyond a meaningful privacy guarantee. By exploiting higher moments of the privacy loss variable, the accounting method proposed by Abadi *et al.* [8] reduces the total amount of additive noise significantly. However, it only achieves an accuracy of 90% (with a privacy budget of $\epsilon = 0.5, \delta = 10^{-5}$) on MNIST.

Compared to [7], [8], our work characterizes the relationship between model utility and the privacy constraints for the first time. So far, existing differential privacy mechanisms are mostly heuristic, which is detrimental to the results as an overly conservative privacy constraint usually requires an overwhelming amount of noise to be added. Another critical drawback in existing works [7], [8] is that their composition methods are suboptimal. In contrast, we adopt the optimal composition theorem [11] and further amplify the privacy guarantee with input sampling [12]. This is the best general composition result that can be achieved so far.

When the cost function is convex or strongly convex, some approaches have been proposed to achieve optimal or near optimal utility bounds. The utility is defined as the worst-case (over inputs) expected excess empirical risk [13]. The approaches include gradient perturbation [13]–[15], output perturbation [13], [15], [16], and objective perturbation [16]. In our work, we do not rely on the convexity of the cost function and focus on the algorithm's real-world performance.

A wide variety of literature tries to improve the differential privacy mechanism from different perspectives. The optimal mechanism in differential privacy [17], [18] proves that the optimal noise probability distribution has a correlated multidimensional staircase-shaped pdf when dimension $d = 2$ [17] and the discrete query output settings [18]. However, their conclusion can hardly be applied to high-dimensional scenarios which our work targets at.

III. Background

In this section, we first introduce some preliminaries related to the concept of differential privacy, and then review stochastic gradient descent (SGD).

A. Differential Privacy

Differential privacy is originally introduced to ensure that the ability of an adversary to inflict damage to any set of users is independent of whether any individual opts in to, or out of, the dataset [6]. Such an ability prevents any adversary from gaining additional information about any individual. The privacy guarantee is expressed by the logarithmic distance between the posterior probability distributions of two adjacent inputs given the outputs. Adjacent inputs are defined on two sets between which their distance is one unit. Different metrics of the distance can be used, which leads to variations of the differential privacy concept. We use ϵ to define the upper bound of the distribution distance and δ to denote the residual probability. Formally, letting I and I' be the inputs, $\mathcal{O}$ be the output set, and $\mathcal{M}$ be the private mechanism, we have

Definition 1. *A mechanism $\mathcal{M}$ is (ϵ, δ)-differentially private if for all adjacent inputs I and I', and all possible output $\mathcal{O}$,*

$$\Pr[\mathcal{M}(I) \in \mathcal{O}] \leq e^{\epsilon} \cdot \Pr[\mathcal{M}(I') \in \mathcal{O}] + \delta. \tag{1}$$

In the special case of $\delta = 0$ we call $\mathcal{M}$ ϵ-differentially private.

A common category of differential privacy mechanisms is achieved by adding noise generated from a given distribution, for instance, Gaussian or Laplace distribution, to the output. It can be proved that the perturbed output satisfies Eqn. (1). It is widely believed that the tradeoff between privacy and utility lies in that when the additive noise has a higher magnitude, the corresponding privacy level would increase, but the resulting accuracy of the model would decrease.

B. Stochastic Gradient Descent

Stochastic gradient descent (SGD) is a common optimization technique in machine learning. Differential privacy mechanisms have been applied to SGD for preserving data privacy in [7], [8], [10], etc.

In general, a deep neural network is denoted by a multi-dimensional function $\mathbf{F} : \mathcal{X} \mapsto \mathcal{Y}$ featured by a set of parameters $\boldsymbol{\theta}$. We use $\boldsymbol{\theta} \in \mathbb{R}^d$ to represent the flattened vector of parameters where d is its dimension. $\mathbf{X} \in \mathcal{X}$ is a training set from the training data space and $\mathbf{Y} \in \mathcal{Y}$ is the corresponding targeted output. Letting $C = \|\mathbf{F}(\mathbf{X}, \boldsymbol{\theta}) - \mathbf{Y}\|$, the training process is to find $\boldsymbol{\theta}$ such that

$$\underset{\boldsymbol{\theta}}{\text{minimize}}\ C. \tag{2}$$

C represents the cost which is the discrepancy between the output predicted by $\mathbf{F}$ and the target $\mathbf{Y}$. Various forms of the cost function can be applied, such as square error for linear regression, or the logistic regression cost function. With SGD, we repeatedly pick training examples (usually in mini batches) and compute the gradient of the cost function with

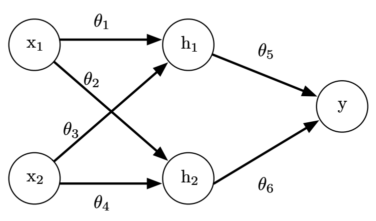


Fig. 1. A multi-layer perceptron architecture.

respect to each parameter. The parameters are updated in the opposite direction of the gradients to minimize the total cost. Differentially private SGD, or noisy SGD, preserves privacy by randomizing the gradient at each update, which inhibits the adversary from inferring about the training data.

IV. Optimized Additive Noise Scheme

In this section, we first show an observation that perturbation on parameters have different impact on cost which is closely associated with training accuracy. Based on that, we approximate the utility depending on the perturbation noise and formulate the problem as one that seeks the pdf of the perturbation noise that minimizes the accuracy loss.

A. Model Sensitivity

We ask the question: for a trained model, if its parameters are perturbed by the same amount of noise, would the perturbed model enjoy the same level of accuracy? For simplicity, we use a cost function to evaluate the model utility — a smaller cost directly implies a higher utility.

The result of distorting model parameters is usually complicated: it may improve the utility for some input samples, but undermine the utility of more. Thus the overall influence of parameter distortion is better gauged by the cost function. We calculate the *derivatives* of the parameters w.r.t. the cost function to approximate the effect of distortion.

The example in analysis is a tiny multi-layer perceptron architecture (Fig. 1). The architecture can be considered a basic unit of a neural network, consisting of an input layer $\{x_1, x_2\}$, a hidden layer $\{h_1, h_2\}$, weights across different layers $\{\theta_1, ..., \theta_6\}$, and the output y. Neurons in the hidden layers apply the sigmoid function $\phi(t) = \frac{1}{1+e^{-t}}$ to the weighted input layer. For instance, $h_1(\mathbf{x}) = \phi(\theta_1 x_1 + \theta_3 x_2 + b_1)$ where θ_1, θ_3 are weights and b_1 is a bias. The output is $y = \theta_5 h_1 + \theta_6 h_2 + b_3$. Weights and biases are parameters of the model and are tuned during training.

We use the perceptron to evaluate a function $f(x_1, x_2) = x_1$ AND x_2 which outputs a binary number given $x_1, x_2 \in [0, 1]$. When x_1, x_2 are not integers, they are rounded up to the closest integer. For example, $f(0.3, 0.7) = 0$. We train on 1000 samples for 150 epochs to obtain the model parameters. Then we randomly choose any two parameters to add noise and record the resulting cost.

In a group of experiments, we perturb θ_1 and b_3 by a value in $(0, 0.3)$ and get the resulting costs as shown by Fig. 2(a). The perturbation also leads to variation in the derivatives $\frac{\partial C}{\partial \theta_1}$ and $\frac{\partial C}{\partial b_3}$, of which we only show the variation of $\frac{\partial C}{\partial \theta_1}$ in Fig. 2(b) due to space constraints. Likewise, Fig. 2(c) shows the cost when θ_6 and b_3 are perturbed.

From Fig. 2(a), one can tell that the least amount of additive noise (coordinate $(0.0, 0.0)$) does not always yield the least cost. Actually, the cost decreases with an increasing θ_1, which is in accordance with the direction indicated by $\frac{\partial C}{\partial \theta_1}$. On the other hand, the cost increases with an increase of b_3. Naturally, one would consider adding more noise to θ_1 to keep the cost minimal. We have similar observations in Fig. 2(c). As $\frac{\partial C}{\partial \theta_6}, \frac{\partial C}{\partial b_3} > 0$, any positive noise would increase the cost and such increase is even higher if we add a greater amount of noise to b_3 than to θ_6 as $\frac{\partial C}{\partial \theta_6} < \frac{\partial C}{\partial b_3}$.

The lessons learned from Fig. 2(a) and Fig. 2(c) are that the sensitivity of cost function is most likely heterogeneous in the parameter space, and the derivative can be an indicator of the sensitivity. For the cost to stay minimal, a higher magnitude of noise is preferred to add to parameters with a lower sensitivity, and it also applies the other way around.

B. Problem Formulation

Following the SGD framework and our observation in Sec. IV-A, we choose to add carefully calibrated noise to each clipped gradients. The additive noise is sampled from a multi-dimensional distribution that minimizes the total perturbation cost according to their respective sensitivity.

Assume that

$$\mathbf{w} = (w_1, w_2, ..., w_d) \in \mathcal{D}^d$$

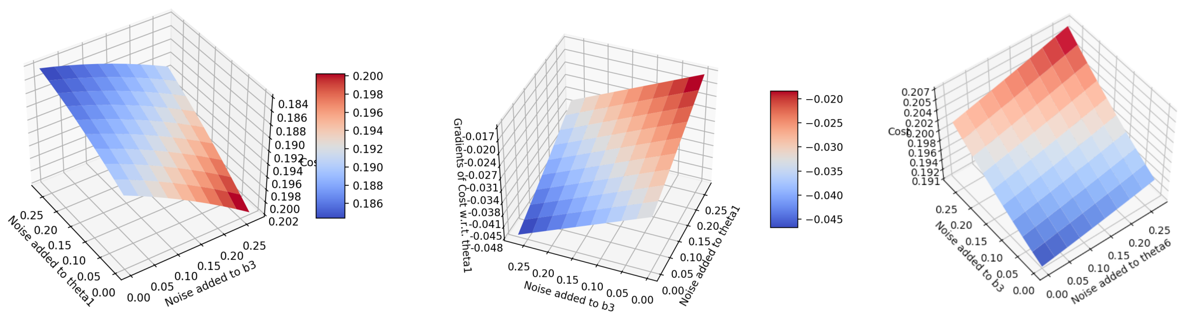
is the derivative vector of the cost on all training examples taken w.r.t. a total of d parameters, and $\mathcal{D}^d \subseteq \mathbb{R}^d$ represents the feasible domain. Note that $\mathbf{w}$ simply represents the model sensitivity, which is different from the gradient taken to update the model parameters in each iteration of SGD. Our additive noise mechanism is $\mathcal{K}(\mathbf{x}) = \mathbf{x} + \mathbf{z}$ where $\mathbf{z} = (z_1, ..., z_d)$ is drawn from a multi-dimensional probability distribution $\mathcal{P}$ defined by the optimization problem shown later.

As discussed in previous sections, to keep the cost minimal, $\mathbf{z}$ should be chosen in accordance with the least sensitive direction of the cost function, and such a direction is indicated by $\mathbf{w}$. We express our objective function as

$$\underset{\mathcal{P}}{\text{minimize}} \int_{z_d} \dots \int_{z_1} \langle \mathbf{w}, \mathbf{z} \rangle \mathcal{P}(\mathrm{d}z_1 \dots \mathrm{d}z_d). \tag{3}$$

In this function, $\mathbf{w}$ represents the model sensitivity w.r.t. each parameter. Eqn. (3) can be interpreted as follows: we project the random noise $\mathbf{z}$ drawn from a particular probability distribution onto a space crafted by the weights $\mathbf{w}$, with the objective of minimizing the expectation of the total projected random noise.

More specifically, when the derivative $\frac{\partial C}{\partial \theta_i}$ is larger than 0, meaning that the cost will increase as θ_i increases, minimizing the objective function will push the additive noise to a direction where z_i is less than 0. Likewise, if $\frac{\partial C}{\partial \theta_i} > \frac{\partial C}{\partial \theta_j} > 0$, the cost is more sensitive to the changes of θ_i than to θ_j. The objective function suggests a perturbation direction such that



(a) (b) (c)

Fig. 2. (a) The cost variation when perturbing θ_1 and b_3. $\frac{\partial C}{\partial \theta_1} \in (-0.048, -0.017)$ and $\frac{\partial C}{\partial b_3} \in (0.017, 0.049)$. (b) The gradient $\frac{\partial C}{\partial \theta_1}$ with different perturbations. (c) The cost when perturbing θ_6 and b_3. $\frac{\partial C}{\partial \theta_6} \in (0.013, 0.022)$ and $\frac{\partial C}{\partial b_3} \in (0.033, 0.056)$.

less noise is added to θ_i than θ_j. Similar claims can be made when the derivative is less than 0.

Next we study the privacy conditions that $\mathcal{P}$ should satisfy. According to the convention [19], the additive noise is drawn from a probability distribution characterized by the global sensitivity. To distinguish from the model sensitivity, we use $\mathbf{g}^t$ to denote the gradients calculated at the t-th iteration. Generally, let $\mathbf{g}^t$ and $\mathbf{g}'^t$ be two gradient vectors respectively computed on training dataset $\mathbf{X}$ and $\mathbf{X}'$, and the two training datasets differ from each other by a single example instance. The global sensitivity is defined as

$$\alpha = \sup_{\forall \mathbf{X}, \mathbf{X}' \text{ s.t. } d(\mathbf{X}, \mathbf{X}')=1} \|\mathbf{g}^t - \mathbf{g}'^t\|, \tag{4}$$

where $d(\cdot)$ represents the Hamming distance, whereas $\|\cdot\|$ is the l_2 norm.

According to Def. (1), we shall have our mechanism $\mathcal{K}$ satisfy that, for any $\mathbf{X}$ and $\mathbf{X}'$ that $d(\mathbf{X}, \mathbf{X}') = 1$ and any output set $\mathcal{O}$:

$$\begin{aligned} &\Pr[\mathcal{K}(\mathbf{g}^t) \in \mathcal{O}] \le e^\epsilon \Pr[\mathcal{K}(\mathbf{g}'^t) \in \mathcal{O}] \\ \Rightarrow &\Pr[\mathbf{g}^t + \mathbf{z} \in \mathcal{O}] \le e^\epsilon \Pr[\mathbf{g}'^t + \mathbf{z} \in \mathcal{O}] \\ \Rightarrow &\Pr[\mathbf{z} \in \mathcal{O} - \mathbf{g}^t] \le e^\epsilon \Pr[\mathbf{z} \in \mathcal{O} - \mathbf{g}'^t] \\ \Rightarrow &\Pr[\mathbf{z} \in \mathcal{O}'] \le e^\epsilon \Pr[\mathbf{z} \in \mathcal{O}' + \mathbf{g}^t - \mathbf{g}'^t], \end{aligned} \tag{5}$$

where $\mathcal{O} - \mathbf{g}^t \triangleq \{\forall \mathbf{o} : \mathbf{o} - \mathbf{g}^t\}$. Letting $\mathbf{\Delta} = \mathbf{g}^t - \mathbf{g}'^t$, the privacy constraint w.r.t. global sensitivity can be written as

$$\Pr[\mathbf{z} \in \mathcal{O}'] \le e^\epsilon \Pr[\mathbf{z} \in \mathcal{O}' + \mathbf{\Delta}] \tag{6}$$

for any $\mathcal{O}'$ and $\|\mathbf{\Delta}\| \le \alpha$. Thus the differential privacy constraint of Eqn. (1) can be expressed as the probability distribution constraint on $\mathcal{P}$.

We assume $\mathbf{z}$ follow the probability distribution $\mathcal{P}$ with pdf $p(\mathbf{z})$. To satisfy Eqn. (6), it suffices to find p such that the logarithmic ratio of $\ln \frac{p(\mathbf{z})}{p(\mathbf{z}+\mathbf{\Delta})}$ is bounded when $\mathbf{\Delta}$ is bounded.

To simplify the problem, we assume p is a symmetric function and the objective function is also symmetric. Note that if the objective function is not symmetric, the optimal p is most likely asymmetric. We consider the perturbation cost as the total magnitude of the weighted noise. For example, the weight of z_i is $|w_i|$: a larger $|w_i|$ indicates that the unit increment in θ_i will lead to a larger perturbation in the output, and thus it is desired to keep the additive noise z_i small. We rewrite our optimization problem as follows:

$$\begin{aligned} &\underset{p}{\text{minimize}} \int_{\mathbf{z} \in \mathbb{R}^d} \|\mathbf{w} \circ \mathbf{z}\|_1 p(\mathbf{z}) \mathrm{d}\mathbf{z} \\ &\text{s.t. } \ln \frac{p(\mathbf{z})}{p(\mathbf{z}+\mathbf{\Delta})} \le \epsilon, \ \forall \|\mathbf{\Delta}\| \le \alpha, \mathbf{\Delta} \in \mathbb{R}^d. \end{aligned} \tag{7}$$

In the equations above, $\mathbf{w} \circ \mathbf{z}$ represents the entry-wise product of the two vectors. $\mathrm{d}\mathbf{z}$ is short for $\mathrm{d}z_1 \ldots \mathrm{d}z_d$. In essence, we search a probability distribution of the random noise that minimizes the expectation of the total amount of additive noise, and at the same time satisfying the requirement of differential privacy.

Prior to our work, Geng *et al.* [20] proved that the optimal pdf of the random noise to minimize its l_1 norm is a symmetric and staircase-shaped function when $d = 2$. However, adding the least amount of noise is not exactly what we want according to the observations of Sec. IV-A. Our goal is to release a perturbed model with high accuracy while satisfying the differential privacy property at the same time.

C. Main Result

To solve Eqn. (7), we further assume p is a multivariate Gaussian distribution of which each dimension is independent from each other, so we are able to present the pdf in a product form: $p(\mathbf{z}) = \prod_{i=1}^d p_i(z_i)$ with each $p_i(z_i)$ being the pdf of $\mathcal{N}(0, \sigma_i^2)$ from which the random noise z_i is drawn. Thus, the problem reduces to the follows:

$$\begin{aligned} &\underset{\sigma_1,\ldots,\sigma_d}{\text{minimize}} \int_{\mathbf{z} \in \mathbb{R}^d} \|\mathbf{w} \circ \mathbf{z}\|_1 p(\mathbf{z}) \mathrm{d}\mathbf{z} \\ &\text{s.t. } \Pr\left[\ln \frac{p(\mathbf{z})}{p(\mathbf{z}+\Delta)} > \epsilon\right] < \delta, \ \forall \|\Delta\| \le \alpha, \Delta \in \mathbb{R}^d. \end{aligned} \tag{8}$$

In this section, we will show how we solve Eqn. (8). First, we rewrite the objective function as

$$\int_{\mathbf{z}\in\mathbb{R}^d} \|\mathbf{w}\circ\mathbf{z}\|_1 p(\mathbf{z})\mathrm{d}\mathbf{z} = \sum_{i=1}^{d}\int_{\mathbf{z}\in\mathbb{R}^d} |w_i z_i|\cdot p(\mathbf{z})\mathrm{d}\mathbf{z}$$
$$= \sum_{i=1}^{d}\int_{z_i} |w_i z_i|\cdot \frac{1}{\sqrt{2\pi}\sigma_i}\exp\Big(-\frac{z_i^2}{2\sigma_i^2}\Big)\mathrm{d}z_i$$
$$= 2\sum_{i=1}^{d}\frac{|w_i|}{\sqrt{2\pi}}\int_0^{+\infty}\frac{z_i}{\sigma_i}\exp\Big(-\frac{z_i^2}{2\sigma_i^2}\Big)\mathrm{d}z_i = \sum_{i=1}^{d}\sqrt{2/\pi}|w_i|\sigma_i.$$

The last equation is due to $\int_0^{+\infty} x\exp(-x^2)\mathrm{d}x = 1/2$. Thus the objective is transformed to minimizing over the weighted sum of $\boldsymbol{\sigma} = (\sigma_1,\ldots,\sigma_d)$.

Next we analyze the constraints. We adopt the idea from moments accountant [8] that a higher moment of the privacy loss variable $c = \ln\frac{p(\mathbf{z})}{p(\mathbf{z}+\boldsymbol{\Delta})}$ is used for the differential privacy constraint. The reason is that the constraint on a higher moment of c is able to give a larger feasible range of the constrained variables, which facilitates to find a global optimal solution to Eqn. (8). By the Markov inequality, we transform the constraint as:

$$\begin{aligned}&\Pr[c>\epsilon] = \Pr[\lambda c > \lambda\epsilon] = \Pr[\exp(\lambda c) > \exp(\lambda\epsilon)]\\ &\le \mathbb{E}[\exp(\lambda c)]/\exp(\lambda\epsilon) \le \delta,\end{aligned} \tag{9}$$

for any positive integer λ. We rewrite c with the expression of Gaussian distributions, and let $\boldsymbol{\Delta} = (\Delta_1,\ldots,\Delta_d)$ to get

$$c = \ln\frac{p(\mathbf{z})}{p(\mathbf{z}+\boldsymbol{\Delta})} = \ln\prod_{i=1}^{d}\frac{p(z_i)}{p(z_i+\Delta_i)}$$
$$= \ln\prod_{i=1}^{d}\frac{\exp\Big(-z_i^2/2\sigma_i^2\Big)}{\exp\Big(-(z_i+\Delta_i)^2/2\sigma_i^2\Big)} = \sum_{i=1}^{d}\frac{2z_i\Delta_i+\Delta_i^2}{2\sigma_i^2}.$$

Since $z_i \sim \mathcal{N}(0,\sigma_i^2)$, we have

$$c \sim \mathcal{N}\Big(\sum_{i=1}^{d}\frac{\Delta_i^2}{2\sigma_i^2}, \sum_{i=1}^{d}\frac{\Delta_i^2}{\sigma_i^2}\Big)$$

due to the additive and scaling properties of Gaussian distributions. Then the Markov inequality of Eqn. (9) suggests that

$$\mathbb{E}[\exp(\lambda c)]/\exp(\lambda\epsilon) = \exp\Big(\lambda\cdot\frac{1}{2}\sum_{i=1}^{d}\Delta_i^2/\sigma_i^2 - \lambda\epsilon\Big) \le \delta,$$
$$\text{i.e.,}\quad \sum_{i=1}^{d}\Delta_i^2/\sigma_i^2 \le 2\epsilon + \frac{2}{\lambda}\ln\delta.$$

Letting $\tau(\epsilon,\delta) \triangleq 2\epsilon + (2/\lambda)\ln\delta$, Eqn. (8) can be rewritten as

$$\underset{\boldsymbol{\sigma}}{\text{minimize}} \quad \sum_{i=1}^{d}|w_i|\cdot\sigma_i \tag{10a}$$
$$\text{subject to} \quad \sum_{i=1}^{d}\Delta_i^2/\sigma_i^2 \le \tau(\epsilon,\delta) \tag{10b}$$
$$\forall\Delta_i,\ \text{such that}\ \sum_{i=1}^{d}\Delta_i^2 \le \alpha^2. \tag{10c}$$

The constraint of Eqn. (10c) can be replaced with an equality since any solution satisfies $\sum_{i=1}^{d}\Delta_i^2 = \alpha^2, \forall\Delta_i$ would have to satisfy Eqn. (10c), and it also applies the other way around. We further transform Eqn. (10) into a minimax problem equivalently:

$$\min_{\boldsymbol{\sigma}}\max_{\boldsymbol{\Delta}} \quad \sum_{i=1}^{d}|w_i|\cdot\sigma_i + \mu\cdot s\big(\sum_{i=1}^{d}\Delta_i^2/\sigma_i^2 - \tau(\epsilon,\delta)\big) \tag{11a}$$
$$\text{subject to} \quad \sum_{i=1}^{d}\Delta_i^2 = \alpha^2, \tag{11b}$$
$$\mu > 0, \tag{11c}$$

where $s(x)$ is the step function of which the output is zero when $x \le 0$ and one when $x > 0$. We show that Eqn. (11) is equivalent to (10) when $\mu \to +\infty$.

Proof. For simplicity, we use the following notations: $g(\boldsymbol{\sigma}) = \sum_{i=1}^{d}|w_i|\cdot\sigma_i$, $h(\boldsymbol{\sigma},\boldsymbol{\Delta}) = \sum_{i=1}^{d}\Delta_i^2/\sigma_i^2$, and $f(\boldsymbol{\sigma},\boldsymbol{\Delta}) = g(\boldsymbol{\sigma}) + \mu\cdot s(h(\boldsymbol{\sigma},\boldsymbol{\Delta}) - \tau(\epsilon,\delta))$. Assuming $\boldsymbol{\sigma}^*$ is the optimal solution to Eqn. (10) and $\boldsymbol{\sigma}', \boldsymbol{\Delta}'$ are the solutions to Eqn. (11) where $\boldsymbol{\sigma}^* \ne \boldsymbol{\sigma}'$. Then we have $f(\boldsymbol{\sigma}',\boldsymbol{\Delta}') \le f(\boldsymbol{\sigma}^*,\boldsymbol{\Delta}')$ since $\boldsymbol{\sigma}'$ minimizes $f(\boldsymbol{\sigma},\boldsymbol{\Delta})$.

$\boldsymbol{\sigma}^*$ must satisfy Eqn. (10b) for any $\boldsymbol{\Delta}$ that meets Eqn. (10c). Since $\boldsymbol{\sigma}'$ satisfies Eqn. (11b) which indicates the condition of Eqn. (10c), the value of the step function is set to zero in $f(\boldsymbol{\sigma}^*,\boldsymbol{\Delta}')$. Hence the step function in $f(\boldsymbol{\sigma}',\boldsymbol{\Delta}')$ should also be zero otherwise its value would go to infinity violating the inequality. For its step function to be zero, $h(\boldsymbol{\sigma}',\boldsymbol{\Delta}') \le \tau(\epsilon,\delta)$ must hold for the $\boldsymbol{\Delta}'$ satisfying Eqn. (11b) at the same time, which means $\boldsymbol{\sigma}',\boldsymbol{\Delta}'$ can also satisfy Eqn. (10b) and (10c). This implies that $\boldsymbol{\sigma}'$ is the optimal solution to Eqn. (10) since $f(\boldsymbol{\sigma}',\boldsymbol{\Delta}') = g(\boldsymbol{\sigma}') \le g(\boldsymbol{\sigma}^*) = f(\boldsymbol{\sigma}^*,\boldsymbol{\Delta}')$. It is contradictory to the condition that $\boldsymbol{\sigma}^*$ is the optimal solution and $\boldsymbol{\sigma}^* \ne \boldsymbol{\sigma}'$.

Therefore, $\boldsymbol{\sigma}^* = \boldsymbol{\sigma}'$ and thus solving Eqn. (11) is equivalent to solving Eqn. (10). Proof completes. □

To solve Eqn. (11), we use a sigmoid function to approximate the step function, and alternatively update the value of $\boldsymbol{\sigma}$ and $\boldsymbol{\Delta}$ while gradually increasing μ. Since the feasible domain of Eqn. (11) is non-convex, and is hard to be transformed into a convex one, standard optimization techniques such as interior-point methods can be used to find a solution. We adopt projected gradient ascent (descent): at each step we move in the direction of the negative gradient, and then 'project' onto the feasible set.

Composition. Solving Eqn. (8) only shows how to provide privacy guarantee in a single iteration. In practice, the SGD algorithm takes many iterations until achieving satisfactory accuracy. Unfortunately, the iterative computation process will expose the training set multiple times, which leads to a degraded privacy level.

We adopt the advanced composition theorem for differential privacy [11] and privacy amplification by a sampling approach [13] in our mechanism. Assuming that we randomly select a mini batch from the training dataset with sampling probability

q in each iteration, for k iterations to achieve (ϵ, δ)-differential privacy, the t-th iteration should satisfy (ϵ^t, δ^t)-differential privacy such that:

$$\epsilon^t = \epsilon / \sqrt{16q^2 k \log(e + \frac{\epsilon}{\delta})}, \delta^t = \delta/2k. \tag{12}$$

D. Privacy Mechanism

Our mechanism can be considered an instantiation of the differentially private SGD. Overall, to achieve (ϵ, δ)-differential privacy for k iterations of SGD, we first obtain the differential privacy requirement (ϵ^t, δ^t) for each single iteration, and then adopt the optimized additive noise mechanism in Sec. IV-C to meet the differential privacy requirement per iteration. Alg. 1 shows the details of the proposed mechanism.

Algorithm 1 Optimized Additive Noise Mechanism

Input: Training dataset $(\mathbf{X}, \mathbf{Y})$, total number of training examples n, cost function $C(\cdot)$, clipping value α, learning rate η^t, total iterations k, privacy parameters (ϵ, δ)

Output: $\boldsymbol{\theta}^k$

```
1:  Compute per-iteration privacy parameters $(\epsilon^t, \delta^t)$ by Eqn. (12).
2:  for $t \in \{0, ..., k-1\}$ do
3:      Compute model sensitivity: $\mathbf{w} = \frac{1}{n} \cdot \frac{\partial C(\mathbf{X},\mathbf{Y})}{\partial \boldsymbol{\theta}^t}$.
4:      Compute $\sigma_1, ...\sigma_d$ by substituting $\mathbf{w}, \alpha, \epsilon^t, \delta^t$ to Eqn. (8).
5:      Sample $\mathbf{z} = (z_1, \ldots, z_d)$ where $z_i \sim \mathcal{N}(0, \sigma_i^2)$.
6:      Randomly sample a batch $\mathcal{B}$ of training data $(\mathbf{X}_\mathcal{B}, \mathbf{Y}_\mathcal{B})$ with probability $q$.
7:      for $(x, y) \in (\mathbf{X}_\mathcal{B}, \mathbf{Y}_\mathcal{B})$ do
8:          Compute $\mathbf{g}_x^t = \nabla_{\boldsymbol{\theta}^t} C(\boldsymbol{\theta}^t, x, y)$.
9:          Clip by $\alpha$: $\bar{\mathbf{g}}_x^t = \mathbf{g}_x^t / \max(1, \|\mathbf{g}_x^t\|/\alpha)$.
10:     end for
11:     Compute the average: $\bar{\mathbf{g}}_\mathcal{B}^t = 1/|\mathcal{B}| \sum_{x \in \mathcal{B}} \bar{\mathbf{g}}_x^t$.
12:     Add noise: $\tilde{\mathbf{g}}_\mathcal{B}^t = \bar{\mathbf{g}}_\mathcal{B}^t + \mathbf{z}$.
13:     Update $\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t - \eta^t \tilde{\mathbf{g}}_\mathcal{B}^t$.
14: end for
15: Return $\boldsymbol{\theta}^k$.
```

Theorem 1. *Alg. 1 satisfies (ϵ, δ)-differential privacy.*

Proof. We first show that one iteration of computation satisfies (ϵ^t, δ^t)-differential privacy. To prove that, we need to show the definition of differential privacy Eqn. (1) holds true for any pair of adjacent inputs. Let $\mathbf{X}$ and $\mathbf{X}'$ be any pair of training dataset with a single entry difference, and w.l.o.g., we have $\mathbf{X} = \mathbf{X}' \cup x$. Since n is very large, $\frac{1}{n}C(\mathbf{X}, \mathbf{Y}) = \frac{1}{n}C(\mathbf{X}', \mathbf{Y}) + \frac{1}{n}C(x, y) \approx \frac{1}{n}C(\mathbf{X}', \mathbf{Y})$ and thus the model sensitivity $\mathbf{w}(\mathbf{X}) \approx \mathbf{w}(\mathbf{X}')$. As the solution to Eqn. (8) only depends on $\mathbf{w}, \alpha, \epsilon^t, \delta^t$, the probability distribution of $\mathbf{z}$ is independent to query outputs.

Since $d(\mathbf{X}, \mathbf{X}') = 1$, $\|\bar{\mathbf{g}}_\mathcal{B}^t(\mathbf{X}) - \bar{\mathbf{g}}_\mathcal{B}^t(\mathbf{X}')\| \leq \alpha$. Let events $\mathcal{S} = \{\mathbf{z} : \ln \frac{p(\mathbf{z})}{p(\mathbf{z}+\Delta)} > \epsilon^t\}$, and $\mathcal{S}^c = \{\mathbf{z} : \ln \frac{p(\mathbf{z})}{p(\mathbf{z}+\Delta)} \leq \epsilon^t\}$, for any $\|\Delta\| \leq \alpha$. Then for any $\mathcal{O}$,

$$\begin{aligned}
&\Pr[\mathcal{K}(\bar{\mathbf{g}}_\mathcal{B}^t(\mathbf{X})) \in \mathcal{O}] = \Pr[\bar{\mathbf{g}}_\mathcal{B}^t(\mathbf{X}) + \mathbf{z} \in \mathcal{O}] \\
&= \Pr[\bar{\mathbf{g}}_\mathcal{B}^t(\mathbf{X}) + \mathbf{z} \in \mathcal{O} \cap \mathcal{S}^c] + \Pr[\bar{\mathbf{g}}_\mathcal{B}^t(\mathbf{X}) + \mathbf{z} \in \mathcal{O} \cap \mathcal{S}] \\
&\leq \exp(\epsilon^t) \Pr[\bar{\mathbf{g}}_\mathcal{B}^t(\mathbf{X}') + \mathbf{z} \in \mathcal{O} \cap \mathcal{S}^c] + \Pr[\mathcal{S}] \\
&\leq \exp(\epsilon^t) \Pr[\bar{\mathbf{g}}_\mathcal{B}^t(\mathbf{X}') + \mathbf{z} \in \mathcal{O}] + \delta^t \\
&= \exp(\epsilon^t) \Pr[\mathcal{K}(\bar{\mathbf{g}}_\mathcal{B}^t(\mathbf{X}')) \in \mathcal{O}] + \delta^t.
\end{aligned}$$

The second inequality holds due to the constraint of Eqn. (8). Thus the t-th iteration of Alg. 1 is (ϵ^t, δ^t)-differentially private, and the composition result of k iterations is (ϵ, δ)-differentially private. Proof completes. □

V. Evaluation

Despite its theoretical guarantee, it remains a challenge to solve Eqn. (8) due to the high-dimensional model parameters. In this section, we introduce our implementation of the privacy mechanisms on TensorFlow and its code-level optimization. In the latter part of the section, we compare the evaluation results of our design against the state-of-the-art.

A. Implementation

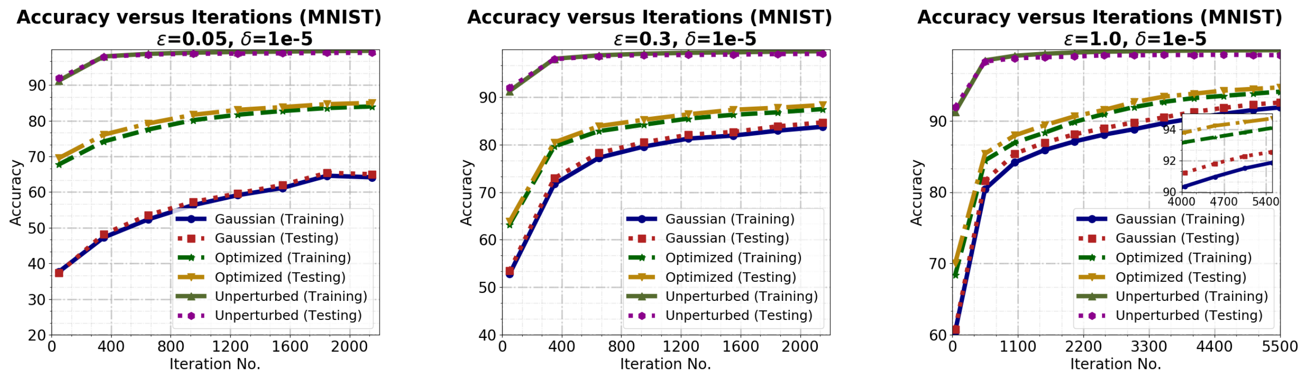
We have developed a general-purpose framework on TensorFlow, which supports MNIST, SVHN, CIFAR-10 and customized datasets with pluggable models and a convenient user interface. As a baseline, we train models on different datasets to their state-of-the-art accuracy without any privacy constraint.

The privacy mechanisms have been implemented as a noise generation module in our framework. The module contains two noise generators: a Gaussian noise generator which implements the moments accountant method [8] and our optimized noise generator that implements the optimized additive noise mechanism in Alg. 1.

To implement the optimized additive noise mechanism, we first adopted the log barrier method with gradient descent to solve Eqn. (11), but found it too slow to converge. We finally use the projected gradient descent which, in each iteration, moves in the opposite direction of the gradient projected to the constraint set. The method takes around 50 iterations to converge.

Despite the optimization technique used, the problem still faces computational challenges due to its high dimensionality. For example, with our `Numpy` version of the noise generators, it takes significant time to process even a single batch of data.

We realize that `Numpy`-based optimizer and noise generator slows the entire process down as it cannot take advantage of GPUs. In each iteration, it needs to evaluate the model sensitivity by running a part of, or the entire graph, gets the value and then run the noise generator on CPU. It is inefficient as the noise generator is not using the GPU. Thus we re-implemented the noise generator as a part of the computation graph using tensor operations. However, while the approach speeds up the Gaussian noise generator, we surprisingly find it barely improve the performance of the optimized noise generator. As we found, it is because generating a random vector



(a) (b) (c)

Fig. 3. Accuracy results at different privacy levels: $\epsilon = 0.05, 0.3, 1.0$ correspond to high, medium and low privacy levels. Our mechanism achieves 84.24%/85.19%, 87.59%/88.55%, and 94.09%/94.69% training/testing accuracy correspondingly. The figures show a significant improvement over the moments accountant method.

from a high-dimensional distribution is extremely inefficient using tensor operations. Hence our final version of the noise generator includes two parts — a tensor computing the noise distribution and a `Numpy` noise generator. The average batch processing time of the Gaussian scheme, our optimized scheme and the unperturbed baseline are shown in Table. I, which shows the noise generator only adds moderate computation overhead to the baseline. The running time is measured on AWS EC2 `p2.xlarge` instances.

TABLE I
AVERAGE BATCH PROCESSING TIME (S)

	Unperturbed	Gaussian	Optimized
MNIST	0.344	0.623	0.937
SVHN	0.341	0.713	0.912
CIFAR-10	0.83	1.437	1.968

B. Experimental Setup

The default experimental setting is given in Table II. A *lot* is the set of training instances grouped together for adding noise, which is independent of the concept *batch*, but a lot typically contains multiple batches.

MNIST is a standard image dataset for handwritten digit recognition, with each image containing 28×28 gray-level pixels. The hidden fully-connected layer contains 128 units. If trained without perturbation, the model reaches 99.79%/99.19% training/testing accuracy after 30 epochs, which is on par with the state-of-the-art. SVHN is a real-world dataset which comes from house numbers in Google Street View images. To apply our privacy mechanisms, we pre-train the model for 20 epochs, fix its convolution layers and re-initialize the fully-connected layers. After re-initialization, the testing accuracy drops from 90% to below 10%, which is close to random choices. The pre-trained convolutional neural network has been used in various machine learning services due to its capability to transfer some information from other datasets. Each image in CIFAR-10 dataset follows $32 \times 32 \times 3$ RGB format. We also adopt the pre-training approach and the accuracy falls below 10% after re-initialization.

TABLE II
EXPERIMENTAL SETUP

	MNIST	SVHN	CIFAR-10
examples (training/testing)	55000/5000	73257/26032	60000/10000
No. of convolutional layers	2	2	6
No. of fully-connected layers	2	3	3
batch size	500	200	500
lot size	1000	1000	1000
clip value	4.0	0.2	0.2

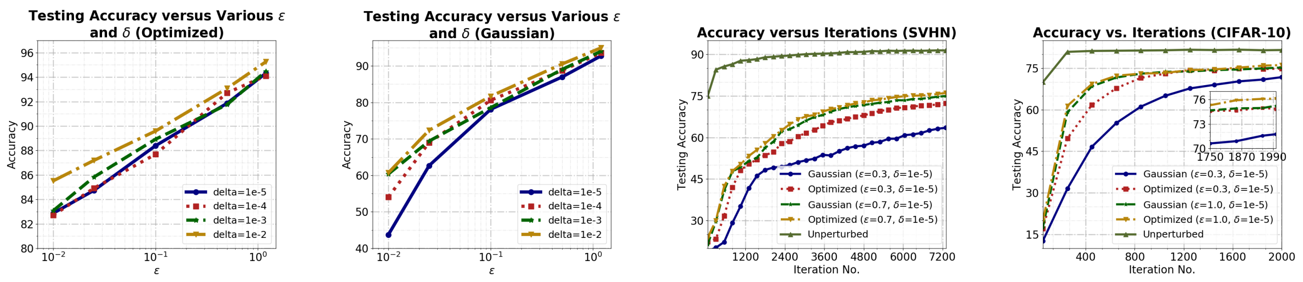
C. Comparison with Moments Accountant

We compare our mechanism with the baseline and the moments accountant method on three datasets. The baseline is the unperturbed case and the latter represents the state-of-the-art differentially private SGD algorithm in related works.

MNIST. Fig. 3(a)-3(c) give the per-iteration training and testing accuracies for different mechanisms when the privacy level is high, medium, and low, which correspond to the cases that a large, moderate, and small amount of perturbation noise is added to the model. The results show that the optimized noise generator surpasses the Gaussian noise generator in terms of accuracy at all levels, more significantly when the privacy level is high.

Specifically, when $\epsilon = 0.05$, training/testing accuracy is as high as 84.24%/85.19% for the optimized noise generator, which is much higher (29%/28%) than that of the Gaussian mechanism. It can be observed that its accuracy ramps up quickly in the first few iterations. Likewise, we obtain a training/testing accuracy of 87.59%/88.55% when $\epsilon = 0.3$, which still improves by 4% compared to the Gaussian mechanism. At $\epsilon = 1.0$, the training/testing accuracy of the optimized mechanism reaches 94.09%/94.69% which is close to the unperturbed case. It is clear that our privacy mechanism has increasing advantage over the previous work as the privacy constraint gets stricter.

One may argue that Fig. 3(a)-3(c) do not fairly reflect the tradeoff between privacy and utility since the algorithm runs for different numbers of iterations at varying privacy levels. In fact, the increase of iterations has contradictory effects to



(a) (b) (c) (d)

Fig. 4. (a)(b) Testing accuracy on MNIST under a variety of privacy settings: the optimized additive noise mechanism offers a better tradeoff. (c) Testing accuracy of SVHN: when $\epsilon = 0.3/0.7$, the optimized mechanism achieves $72.35\%/76.15\%$ and the Gaussian mechanism achieves $63.62\%/74.69\%$ testing accuracy. (d) Testing accuracy of CIFAR-10: the optimized mechanism achieves $74.85\%/76.1\%$ respectively for $\epsilon = 0.3/1.0$. And the Gaussian mechanism obtains $71.75\%/75.23\%$ respectively for the same ϵs.

accuracy: while it continues to improve training accuracy by minimizing the cost function, the model also suffers from further perturbations. By Eqn. (12), we adjust the number of iterations k according to different ϵs, such that appropriate privacy parameters can be chosen per iteration.

To eliminate the concern about iteration numbers, we conduct experiments for different groups of (ϵ, δ) where we report its testing accuracy after the model has been trained for 30 epochs. The results for the optimized and the Gaussian mechanisms are respectively shown in Fig. 4(a) and Fig. 4(b). Training accuracy result is similar thus omitted due to space constraints. The general trend is that the accuracy grows as ϵ or δ increases, which shows the tradeoff between model utility and privacy. Although both mechanisms achieve a similar accuracy at the low privacy regime, for example, $(\epsilon = 1.2, \delta = 10^{-2})$, the difference in the high privacy regime $(\epsilon = 0.01, \delta = 10^{-5})$ is substantial: the optimized mechanism achieves a training/testing accuracy of $81.7\%/82.87\%$, while the Gaussian mechanism only reaches $43.95\%/43.75\%$. This discrepancy verifies that our optimized noise addition mechanism is particularly effective with tight privacy constraints.

SVHN. Our baseline model for SVHN achieves 91.35% testing accuracy. The result in Fig. 4(c) shows that the optimized mechanism performs better than the Gaussian at both low privacy (when $\epsilon = 0.7$) and high privacy (when $\epsilon = 0.3$) levels. The improvement over the Gaussian mechanism is 2% and 13.7% respectively.

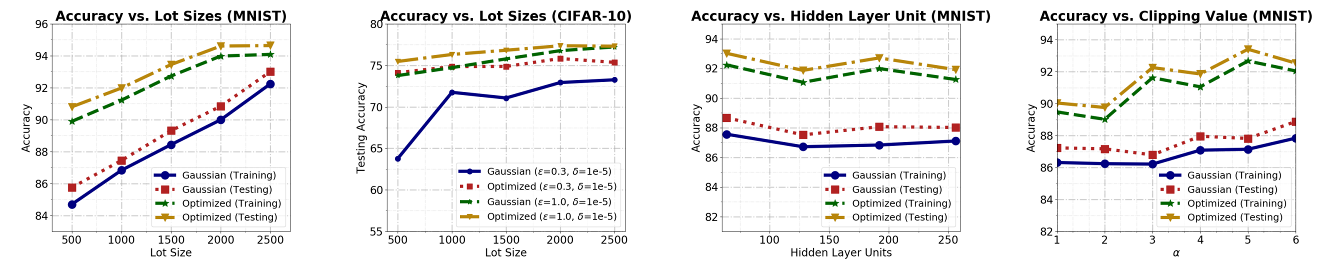
CIFAR-10. Although it is more challenging to achieve high accuracy on more complicated datasets like CIFAR-10, we show that our privacy mechanism still maintains its superior performance even under strict privacy constraints. Fig. 4(d) shows the per-iteration testing accuracy when the model is trained with or without the privacy guarantee. While the baseline testing accuracy is 81.51%, the optimized mechanism reaches 76.1% and 74.85% respectively when $\epsilon = 1.0$ and 0.3. And the Gaussian mechanism only obtains 75.23% and 71.75% correspondingly for these two privacy levels. The result again supports the claim that our mechanism has a larger advantage in the high privacy regime.

Comparison with prediction perturbation methods. There is actually no straightforward comparison between our work and private aggregation of teacher ensembles [21], [22], as the requirement and pre-processing of the dataset is entirely different. For example, we do not require public data as in some cases such data would be unavailable. Nevertheless, we adopt their reported results on the same dataset to have an overall idea on the performance. On MNIST, PATE [22] achieves 98.5% accuracy with $\epsilon = 1.97$, $\delta = 10^{-5}$, while we achieve 94.69% accuracy when $\epsilon = 1.0$, $\delta = 10^{-5}$. On SVHN, LNMax [21] reaches 82.7% accuracy at $\epsilon = 5.04$, $\delta = 10^{-6}$ while our mechanism obtains 76.15% at $\epsilon = 0.7$, $\delta = 10^{-5}$. In both cases, our mechanism is able to gain similar accuracy performance over much stricter privacy constraints. In fact, we consider in reality, differential privacy is only valid when ϵ is small (mostly $\epsilon < 1.0$), but most previous works have not addressed the accuracy performance at such a privacy level.

D. *Sensitivity to Model Structures*

Privacy and accuracy cannot be discussed without the specific neural network structure. We show that the optimized mechanism is highly robust to the change of model structures and hyperparameters. We consider the following hyperparameters which may affect the result: lot size, the number of hidden layer units, and the l_2 clipping value α. For each group of experiments, we reuse the previous setting except for the controlled hyperparameters.

From Fig. 5(a) and Fig. 5(b), we observe that the gap between the optimized and the Gaussian mechanism is larger when the lot size is smaller. Intuitively, when the lot size is small, a greater amount of noise is accumulated to the model. With the moments accountant method, a greater amount of noise always leads to poorer accuracy. When the optimized mechanism is applied, the accuracy on MNIST first increases with the lot size, reaches the peak at around 2000, and then stays the same. The accuracy on CIFAR-10 merely increases by 2% at $\epsilon = 1.0$ and $\epsilon = 0.3$ as the lot size changes from 500 to 2500. This is consistent with our earlier observation that a greater amount of additive noise does not necessarily lead to a lower accuracy. Since the optimized noise mechanism calibrates noise carefully to minimize the cost, the model accuracy suffers little from the varying lot size.



(a) (b) (c) (d)

Fig. 5. (a)(b) Testing accuracy vs. lot sizes on MNIST and CIFAR-10: the optimized mechanism is robust to the lot size changes. (c) Both privacy mechanisms are invariant to model structure changes. (d) The choice of clipping value has moderate impact on both privacy mechanisms.

From Fig. 5(c), we can tell both mechanisms are invariant to the neural network structure changes. Despite the variation in the number of hidden layer units, the fluctuation in training/testing accuracy is less than 2% for both mechanisms. However, the optimized mechanism is more sensitive to the change in the clipping value as shown in Fig. 5(d). This is because when the clipping value is small, the change in the parameters is constrained by the clipping value thus the accuracy suffers; but we don't observe the same trend in the Gaussian mechanism, since the additive noise is too overwhelming when the clipping value is large.

VI. Conclusion

In this paper, we seek an optimized differential privacy mechanism for performing privacy-preserving learning over crowdsourced user data. The problem is formulated as an optimization that minimizes the accuracy loss over a set of differential privacy constraints. The high dimensionality of the problem is a major obstacle, so we tackle it from both a theoretical and an engineering perspective. Our evaluations on MNIST, SVHN, and CIFAR-10 have shown clear evidence that our proposed privacy mechanism improves the model accuracy at all privacy levels, especially in the high privacy regime, yet only adding a negligible runtime overhead.

References

[1] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership Inference Attacks against Machine Learning Models," in *Proc. of the 2017 IEEE Symposium on Security and Privacy (S&P),*. IEEE, 2017, pp. 3–18.

[2] J. Zhang, J. Sun, R. Zhang, Y. Zhang, and X. Hu, "Privacy-preserving social media data outsourcing," in *Proc. of 2018 IEEE International Conference on Computer Communications (INFOCOM)*. IEEE, 2018.

[3] X. Lou, R. Tan, D. K. Yau, and P. Cheng, "Cost of Differential Privacy in Demand Reporting for Smart Grid Economic Dispatch," in *Proc. of 2017 IEEE International Conference on Computer Communications (INFOCOM)*. IEEE, 2017.

[4] Q. Wang, Y. Zhang, X. Lu, Z. Wang, Z. Qin, and K. Ren, "Rescuedp: Real-time spatio-temporal crowd-sourced data publishing with differential privacy," in *Proc. of 2016 IEEE International Conference on Computer Communications (INFOCOM)*. IEEE, 2016, pp. 1–9.

[5] L. Zhao, L. Ni, S. Hu, Y. Chen, P. Zhou, F. Xiao, and L. Wu, "InPrivate Digging: Enabling Tree-based Distributed Data Mining with Differential Privacy," in *Proc. of 2018 IEEE International Conference on Computer Communications (INFOCOM)*. IEEE, 2018.

[6] C. Dwork, "A Firm Foundation for Private Data Analysis," *Communications of the ACM*, vol. 54, no. 1, pp. 86–95, 2011.

[7] R. Shokri and V. Shmatikov, "Privacy-preserving Deep Learning," in *Proc. of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2015, pp. 1310–1321.

[8] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep Learning with Differential Privacy," in *Proc. of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2016, pp. 308–318.

[9] M. Fredrikson, S. Jha, and T. Ristenpart, "Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures," in *Proc. of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2015, pp. 1322–1333.

[10] S. Song, K. Chaudhuri, and A. D. Sarwate, "Stochastic Gradient Descent with Differentially Private Updates," in *Proc. of the 2013 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. IEEE, 2013, pp. 245–248.

[11] P. Kairouz, S. Oh, and P. Viswanath, "The Composition Theorem for Differential Privacy," *IEEE Transactions on Information Theory*, vol. 63, no. 6, pp. 4037–4049, 2017.

[12] A. Beimel, S. P. Kasiviswanathan, and K. Nissim, "Bounds on the Sample Complexity for Private Learning and Private Data Release," in *Theory of Cryptography Conference*. Springer, 2010, pp. 437–454.

[13] R. Bassily, A. Smith, and A. Thakurta, "Private Empirical Risk Minimization: Efficient Algorithms and Tight Error Bounds," in *Proc. of the IEEE 55th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2014, pp. 464–473.

[14] D. Wang, M. Ye, and J. Xu, "Differentially Private Empirical Risk Minimization Revisited: Faster and More General," in *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 2017, pp. 2719–2728.

[15] J. Zhang, K. Zheng, W. Mou, and L. Wang, "Efficient Private ERM for Smooth Objectives," in *Proc. of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*. AAAI Press, 2017, pp. 3922–3928.

[16] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially Private Empirical Risk Minimization," *Journal of Machine Learning Research*, vol. 12, no. Mar, pp. 1069–1109, 2011.

[17] Q. Geng and P. Viswanath, "The Optimal Mechanism in Differential Privacy," in *Proc. of the 2014 IEEE International Symposium on Information Theory (ISIT),*. IEEE, 2014, pp. 2371–2375.

[18] ——, "The Optimal Noise-Adding Mechanism in Differential Privacy," *IEEE Transactions on Information Theory*, vol. 62, no. 2, pp. 925–951, 2016.

[19] C. Dwork, A. Roth *et al.*, "The Algorithmic Foundations of Differential Privacy," *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 3-4, pp. 211–407, 2014.

[20] Q. Geng, P. Kairouz, S. Oh, and P. Viswanath, "The Staircase Mechanism in Differential Privacy," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 7, pp. 1176–1184, 2015.

[21] N. Papernot, M. Abadi, Ú. Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data," in *Proc. of the 5th International Conference on Learning Representations (ICLR)*, 2017.

[22] ——, "Scalable Private Learning with PATE," in *Proc. of the 6th International Conference on Learning Representations (ICLR)*, 2018.