Learning Model-Based Privacy Protection under Budget Constraints

Junyuan Hong,[1] **Haotao Wang,**[2] **Zhangyang Wang,**[2] **Jiayu Zhou**[1]
[1] Department of Computer Science and Engineering
Michigan State University, East Lansing, MI 48823, USA
[2] Department of Electrical and Computer Engineering
University of Texas at Austin, Austin TX 78712, USA
hongju12@msu.edu, htwang@utexas.edu, atlaswang@utexas.edu, jiayuz@msu.edu

Abstract

Protecting privacy in gradient-based learning has become increasingly critical as more sensitive information is being used. Many existing solutions seek to protect the sensitive gradients by constraining the overall privacy cost within a constant budget, where the protection is hand-designed and empirically calibrated to boost the utility of the resulting model. However, it remains challenging to choose the proper protection adapted for specific constraints so that the utility is maximized. To this end, we propose a novel *Learning-to-Protect* algorithm that automatically learns a model-based protector from a set of non-private learning tasks. The learned protector can be applied to private learning tasks to improve utility within the specific privacy budget constraint. Our empirical studies on both synthetic and real datasets demonstrate that the proposed algorithm can achieve a superior utility with a given privacy constraint and generalize well to new private datasets distributed differently as compared to the hand-designed competitors.

Introduction

Trustworthy machine learning has recently drawn a great attention in both industry and research communities (Tang et al. 2017; Ding, Kulkarni, and Yekhanin 2017; Erlingsson, Pihur, and Korolova 2014), due to the increasing awareness in protecting the usage of sensitive data for training. Differential Privacy (DP) (Dwork et al. 2006b; Dwork 2006) is considered to be a *de facto* way to quantitatively protect the training data and is widely studied in the machine learning community. In DP, the privacy cost is measured by the probability difference (i.e., privacy leakage) of a random mechanism aggregating similar datasets. The smaller the difference becomes, the less chance an attacker can get private information from the output.

One major problem setting of DP studies is the differentially private model publishing (Yu et al. 2019), where model parameters are published given a constant DP budget for accessing the private dataset. For gradient-based learning, one way to achieve this goal is protecting the gradients by perturbation and constraining the accumulated privacy cost (Bassily, Smith, and Thakurta 2014; Abadi et al. 2016; Wu et al. 2017). Following the idea, variant protection strategies are proposed with different model update rules (Wang, Ye, and Xu 2017),

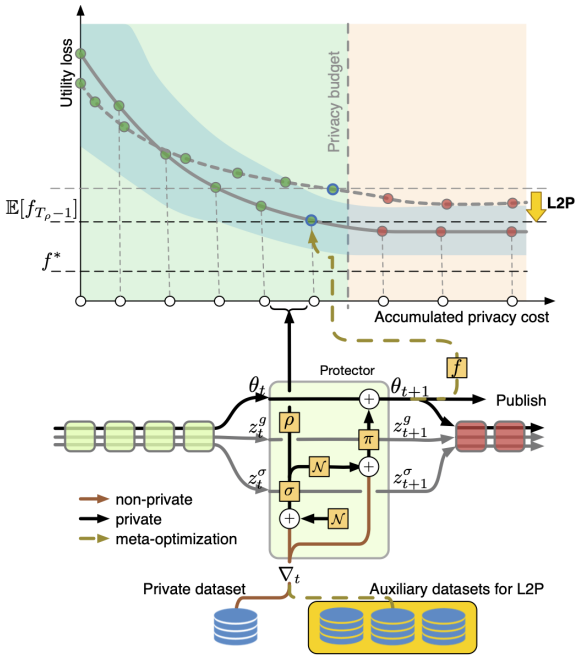

Figure 1: Illustration of the model-based protector and the Learning-to-Protect framework. Two RNN cells, σ and π, represent the adaptive noise scheduler and the utility projector in a protector, respectively.

dynamic perturbation (Yu et al. 2019) or both (Lee and Kifer 2018). Despite the technical differences in these approaches, the same practical target is to *find a protection strategy maximizing utility (the quality of the published model) under a given privacy budget constraint.*

Both empirically (Abadi et al. 2016) and analytically (Wang and Xu 2019), it has been shown that the best protection strategy including hyper-parameters of update rules, budget schedule and etc., significantly depends on the budget constraint. For a specific classes of objective functions, the upper bound on the utility loss can be analytically derived as a function of the protection strategy (Wang, Ye, and Xu 2017). Minimizing the upper bound directly outputs a protection strategy (the step budget and learning rates) parameterized by the budget. However, the principle fails when there is no tight

analytical bound, e.g., for optimizers, like Adam (Kingma and Ba 2015) and model-based optimizers (Andrychowicz et al. 2016; Li and Malik 2017), or for some non-smooth objectives. Alternatively, an empirical way can be used to find such a strategy, such as performing random search and validating them privately (Chaudhuri, Monteleoni, and Sarwate 2011; Gupta et al. 2010; Hay et al. 2009) or non-privately on public auxiliary datasets (Yu et al. 2019; Wu et al. 2017). However, this is costly in privacy or could be inefficient and far away from optimal when the hyper-parameter space is huge, e.g., the model-based ones. Therefore, an efficient search algorithm with fast descent instructions is strongly demanded for a given constraint.

To address the aforementioned challenges, we propose Learning to Protect (L2P) as shown in Fig. 1, that automatically searches for a *protector* conducting an optimal protection strategy, on Stochastic Gradient Descent (SGD), which protects the gradients and meanwhile maximizes the utility within the budget constraint. A protector is composed of a *projector* updating the model parameters, and a *scheduler* allocating budgets at each iteration. We generalize the traditional hand-crafted protectors to model-based ones which is meta-trained to maximize the utility by gradient descent. The followed challenge is to meta-optimize the objective under budget constraint which is solved by Lagrangian methods. In summary, our contribution includes:

1) Propose a data-driven L2P framework to train a model-based protector;

2) Develop a batch algorithm for L2P that improves the scalability of the budget-constrained meta-learning;

3) Conduct extensive empirical evaluation of the optimality, generalization and scalability of the L2P framework on convex and non-convex problems.

Related Work

Private Learning. Recent years witnessed increasing attention to the privacy risk associated to learning from sensitive training data. For example, an attacker could retrieve the training data from the models generated by the widely-used empirical risk minimization (ERM) (Fredrikson, Jha, and Ristenpart 2015). Many efforts have been devoted toward privacy-preserving learning. With the introduction of differential privacy (DP) (Dwork 2006), we are now able to measure and defend the risk quantitatively (Bassily, Smith, and Thakurta 2014; Kifer, Smith, and Thakurta 2012; Rubinstein et al. 2012; Talwar, Guha Thakurta, and Zhang 2015). The main idea is to introduce stochastic perturbations to the learning process, and the perturbations can be done in any query operations (Dwork 2008), such as gradient computations (Abadi et al. 2016) or objective evaluation (Chaudhuri, Monteleoni, and Sarwate 2011). When proper noise is introduced before publishing the model, such as Gaussian mechanism (Dwork and Roth 2013), one can no longer easily retrieve the training data by resampling (Fredrikson, Jha, and Ristenpart 2015).

Adaptive Privacy Perturbation. The key to achieving high utility in privacy-preserving learning is the perturbation control. (Zhang et al. 2013) improved the performance of models in a stochastically private manner by selecting the gradient candidates. (Lee and Kifer 2018) proposed adaptively and privately querying the effects of the noised gradient updates. Both mechanisms rely on querying the model outputs for several times via an exponential noise mechanism (Dwork 2006) which degrades the effectiveness. Instead, (Balle and Wang 2018) showed a simple adaptive scaling based on the noised value is capable for reducing expectation error. Inspired by this idea, our gradient protector sequentially predicts optimization updates based on current and previous protected gradients which reduce both query times and privacy costs.

Meta-Learning and Knowledge Transfer between (Private and Public) Tasks. The Learning-to-Learn (L2L), by gradients (Andrychowicz et al. 2016; Li and Malik 2017) or by extracted optimization information (Chen et al. 2016), trains a sequential model from a set of learning problems, which predicts the model updates for a new task according to the current gradients. The L2L is shown to greatly reduce the effort in tuning parameters. The proposed L2P shares the same spirit and transfers the privacy protection knowledge from an auxiliary task to a new private task. One benefit is that no extra privacy cost will be introduced when the auxiliary task uses public data. Closely related approaches are (Wu et al. 2017) which uses public data for hyper-parameter tuning, (Yu et al. 2019) which leverages the public validation set for dynamically scheduling the privacy noise and (Zhou, Wu, and Banerjee 2020) which learns projection by a small public set. Moreover, Papernot *et al.*transfer the private knowledge to public tasks (Papernot et al. 2018). The line of literature provides evidence of transferability of knowledge between private and public learning tasks.

Algorithm 1 Protected Stochastic Gradient Descent

Require: Total privacy budget ρ_{tot} with max iteration number $T_{\max}$, dataset $D = \cup_{t=0}^{T-1} D_t$, with $D \subseteq \mathcal{D}^N$, random batches D_t, a loss function $f : \Theta \times \mathcal{D}^N \to \mathbb{R}$ in parameter space Θ, gradient clipping norm C_g, and initial model parameter θ_1.

1: **for** $t \in 1, \cdots, T_{\max}$ **do**
2: $\quad \nabla_{t,i} \leftarrow \nabla_\theta f(\theta; x_i)|_{\theta=\theta_t}, \forall x_i \in D_t$
3: $\quad \nabla_t \leftarrow \sum_{i=1}^{L} \nabla_{t,i} / \max(1, \|\nabla_{t,i}\|_2 / C_g)$
4: $\quad g_t, \rho_t \leftarrow$ Protect(∇_t), e.g., $g_t = \nabla_t + C_g \sigma_t \mathcal{N}(0, I)$, $\rho_t = \rho(\sigma_t)$ or Algorithm 2
5: $\quad \rho_t \leftarrow f_S(\rho_t; q = |D_t|/|D|)$
6: $\quad$ **if** $f_C(\rho_{1:t}) > \rho_{\text{tot}}$ **then** break
7: $\quad \theta_{t+1} \leftarrow \theta_t + g_t$
8: Output θ_t, $f_C(\rho_{1:t-1})$

Private Learning with Budget Constraint

A traditional private learning is Private Stochastic Gradient Descent (PSGD) (Algorithm 1) which was first proposed by Bassily, Smith, and Thakurta (2014). In each step, a mini-batch of randomly selected private samples are used to compute gradients which will be protected by Gaussian noise. The iterations will terminate when the privacy risk is over our expectation, namely *privacy budget*. To quantify the privacy risk, the Differential Privacy is used. In the following, we

present the fundamentals for calculating the privacy cost of publishing the last-iterate model.

Privacy Measurement

In Algorithm 1, the privacy computation involves three basic operations: Gaussian noising $\rho(\sigma_t)$, dynamic composition $f_C(\rho_{1:t})$ and subsampling amplification $f_S(\rho_t; q)$. In this paper, we use (ρ, ∞)-tCDP (or ρ-zCDP) for computing the privacy cost of Gaussian noising, (ρ, ω)-tCDP for composition and subsampling amplification. Though ω could vary on need, the privacy cost of the output model is determined by the minimal ω of all privacy operations[1]. Since the iteration privacy cost will not be over the budget, the ω is lower bounded by a constant ω_a and ρ is upper bound by ρ_a. In addition, constraint also result in different conditions for the subsampling amplification, including the bound of ρ when subsampling. The varying ω will introduce additional complexity in computing total privacy cost[2], which should be avoided by using its lower bound ω_a. To distinguish from tCDP, We call the (ρ, ω_a)-tCDP with $\rho \leq \rho_a$ as the (ρ, a)-ctCDP (Definition 0.1) where a represents a constraint constant.

Definition 0.1 ((ρ, a)-ctCDP)**.** Let $a > 0$, $\omega_a = (1 + a) + \sqrt{a(a+1)}$, and $0 < \rho \leq \rho_a$ where ρ_a is defined as $\epsilon\sqrt{a(a+1)}[a + 1 - \sqrt{a(a+1)}]^{-1}[\sqrt{a(a+1)} - a]^{-1}$. A randomized algorithm $M : \mathcal{D}^n \to \mathbb{R}$ satisfies a-constrained (ρ, ω_a)-tCDP or (ρ, a)-ctCDP if, for all adjacent inputs $d, d' \in \mathcal{D}^n$ with only one different entry,

$$D_\alpha(M(d) \| M(d')) \leq \rho\alpha, \ \forall \alpha \in (1, \omega_a)$$

where $D_\alpha(\cdot\|\cdot)$ denotes the Rényi divergence (Rényi 1961) of order α.

With the notion of (ρ, a)-ctCDP, the corresponding basic privacy operations are $\rho(\sigma_t) = 1/(2\sigma_t^2)$ (Gaussian mechanism), $f_C(\rho_{1:t}) = \sum_{i=1}^t \rho_i$ (Dynamic composition) where ρ_i could differ by iteration and $f_S(\rho; q) = \mathcal{O}(q^2\rho)$ (Subsampling amplification). A brief comparison of (ρ, ω)-tCDP to other privacy metrics is as follows.

- (ϵ, δ)-DP with advanced composition (Bassily, Smith, and Thakurta 2014) does not support non-uniform privacy budget allocation.
- (ϵ, δ)-DP with Moment Accountant (Abadi et al. 2016) has a tight bound (in a limited range of ϵ and δ) on the composition which, however, is expansive to compose privacy costs in the sense of time complexity. Our metric has a linear composition which is efficient and simple.
- ρ-zCDP has linear composition operation but a weak subsampling amplification property under shuffle subsampling (Yu et al. 2019), i.e., $f_S(\rho, q)\mathcal{O}(q\rho)$ for one epoch.
- (α, ϵ)-RDP (Mironov 2017) have similar privacy operations. But it is less tight than (ρ, ω)-tCDP (Bun et al. 2018).

[1]The ρ is also determined by ω (see Theorem .4).

[2]Intensive comparison of ρ and ω has to take place to satisfy the conditions of privacy operations.

Algorithm 2 Model-based Protection

Require:A gradient clipping norm C_g, an optimizer model $(\sigma(\cdot), \pi(\cdot))$, initial hidden states z_1^g, z_1^σ.

1: $s_t \leftarrow \sqrt{\|\nabla_t\|_2^2 + (2|D_t| - 1)C_g^2\sigma_g\zeta_t}, \ \zeta_t \sim \mathcal{N}(0, 1)$
2: $\sigma_t, z_{t+1}^\sigma \leftarrow \sigma(s_t/|D_t|, z_t^\sigma)$
3: $\tilde{\nabla}_t \leftarrow \frac{1}{|D_t|}(\nabla_t + C_g\sigma_t\nu_t), \nu_t \sim \mathcal{N}(0, I)$
4: $g_t, z_{t+1}^g \leftarrow \pi(\tilde{\nabla}_t, z_t^g)$
5: $\rho_t \leftarrow f_C(\rho(\sigma_g), \rho(\sigma_t))$
6: **Output:** g_t, ρ_t

When (ρ, a)-ctCDP maintains the privacy operations of (ρ, ω)-tCDP, it simplifies the privacy parameters by using a constant a, which makes handling privacy parameters easier. More details of the ctCDP and its basic operations are discussed in appendix. Empirical comparisons of the DP bounds and time complexity are enclosed in appendix.

Learning to Protect

In traditional private learning, the privacy schedule strategy and optimizer hyper-parameters are determined by tuning on the public dataset (Wu et al. 2017) or using private tuning algorithm (Gupta et al. 2010; Hay et al. 2009). Using private data for such parameter tuning can be expensive. In fact, knowledge from similar learning experiences can be transfered. Perhaps the most representative and relevant example is the *learning-to-learn* (Andrychowicz et al. 2016) where we revise the gradient based on the experience from other learning tasks. And the choice of such auxiliary learning tasks is commonly not sensitive to learning process. Motivated by the prior successful work, we propose to leverage knowledge of learning tasks from public auxiliary datasets to perform privacy parameter tuning, which effectively saves privacy budgets during the tuning process.

Model-Based Protection

The most attractive attribute of machine learning, especially, deep learning, is the powerful generalization ability of the model learned from data. To cast the protector search problem as a learning problem, it is natural to leverage the generalization ability of Deep Neural Networks (DNNs) and design the protector as a DNN model. In comparison, hand-designed can also be learned only if its objective function is differentiable w.r.t. its hyper-parameters. Now, the method is a direct extension from L2L to the privacy task as Algorithm 2 where we use the properties of ctCDP to compute the privacy costs. Due to the simple properties of the ctCDP, we can easily compute the privacy cost by linear operations and pay a small number of costs for using small noise.

Technically, the protector is composed of: *Scheduler* ($\sigma(\cdot)$) predicts the step noise scale σ_t corresponding to the budget allocation; *Projector* ($\pi(\cdot)$) predicts the step update g_t aiming to boost the optimization performance. In Algorithm 2, we omit the parameters of the two models in formulations and introduce the latent states (z) to pass history information.

More generally, the protector models can hand-designed ones such as the SGD or Adam with uniform schedule.

Both $\sigma(\cdot)$ and $\pi(\cdot)$ are Recurrent Neural Networks (RNNs) in this paper. Inherently, RNNs can approximate the second order information by memorizing history gradients, which is shown to be beneficial in reducing (noise) variance, e.g., private SVRG (Wang, Ye, and Xu 2017) In addition, the model-based scheduler can introduce dynamics such that the privacy budget is allocated adaptively for each step. Therefore, it is possible to enhance the resulting model by avoiding waste of budget on unnecessary places (Lee and Kifer 2018).

When model-based $\sigma(\cdot)$ and $\pi(\cdot)$ are used, we show in Theorem 0.1 that the learning algorithm could still satisfy the preset privacy constraint.

Theorem 0.1 (Privacy guarantee of model-based gradient descent)**.** *Suppose a gradient-based algorithm Algorithm 1 is protected by Algorithm 2 and $\sigma(\cdot)$ and $\pi(\cdot)$ are crafted fully independently from the private data. The output of the algorithm, i.e., θ_t (assuming the loop stop at step t), is $\hat{\rho}$-ctCDP where $\hat{\rho} \leq \rho_{tot}$, if $f_C(\cdot)$, $f_S(\cdot)$ and $\rho(\cdot)$ are defined under ctCDP.*

Proof. In brief, the privacy guarantee is because the noise lowers the probability of information leakage, then the $\sigma(\cdot)$ and $\pi(\cdot)$ uses the noised gradients without further leakage, and last the privacy losses in iterations are composable. The detailed proof is delayed to Theorem .5. □

Meta-Optimization with Constraint

Considering the general optimizers characterized above, we propose learning the gradient-based private optimizer by gradient descent. A full L2P training starts from selecting a public auxiliary dataset, according to the non-private attributes of the private dataset, e.g., data type (image). Then, meta-train the protector $(\sigma(\cdot), \pi(\cdot))$ on the auxiliary dataset by solving

$$\min_{\pi,\sigma,T} \mathbb{E}\left[\tilde{F}(\sigma,\pi,T)\right], \text{ s.t. } h_T(\sigma;\rho_{\text{tot}}) = 0 \qquad (1)$$

where expectation is w.r.t. both the unrolled optimization process and the variety of loss functions. For clarity we use the following brief notations: $f_t \triangleq f(\theta_t;\pi,\sigma)$, $\sigma_t \triangleq \sigma(s_t, z_t^\sigma)$, $h_t(\sigma;\rho) \triangleq f_C(\{\rho(\sigma_t)\}_{t=1}^T) - \rho$, and $\tilde{F}(\sigma,\pi,T) \triangleq f(\theta_T;\pi,\sigma) = f_T$ where $\tilde{F}$ is the random meta-objective, θ_t is depending on θ_{t-1} by unrolling the optimization (Algorithm 1). Though ∇_{t-1} depends on the θ_{t-1}, for tractability we treat it as a constant observation with zero gradient. π, σ and T are alternately updated until converged.

Optimize projector. Optimizing π follows the standard gradient descent and additional tricks can be found in (Andrychowicz et al. 2016). Notably, the value of T varies if the scheduler is dynamic. Since the sole purpose of projector training is finding the proper updates and does not depend on the private constraint, we can continue training it when budget is used up. Thus, in the training we use a constant T that is larger than the maximal iteration.

Optimize scheduler. For simplicity, let us first consider the case when T is fixed and the objective is constrained by the budget as shown in Eq. (1). Given a constant T, we define the meta-objective as $F(\sigma) = \mathbb{E}[f(\theta_T;\pi,\sigma)]$. The optimization can be practically solved using Augmented Lagrangian:

$$L^{\text{aug}}(\sigma;\rho_{\text{tot}}) = F(\sigma) - zh_T + \|h_T\|_2^2/(2\mu) \qquad (2)$$

where $h_T = h_T(\sigma;\rho_{\text{tot}})$, μ is a positive hyper-parameter and z is the Lagrangian multiplier. Augmented Lagrangian algorithm has been well studied in literatures, for example, (Nocedal and Wright 1999) (Chapter 17). The detailed algorithm and analysis on the gradient effects of the objective are in appendix. Also, we provide principled analysis of the influence of σ_t (characterized by derivatives). In brief, we show that a decaying schedule of σ and denoising π will be preferred for the utility of the final model.

Private meta-optimization. When public auxiliary dataset is unavailable, we may use the private dataset to meta-train the projector and scheduler. One method is proposed by Li et al. (2020) for meta-training initialization which is a straightforward gradient noising mechanism. For the gradient-based meta-learning, the strategy is similar by extending the number of unrolling iterations from 1 to T. The meta-gradients are privatized by injecting DP Gaussian noise which consumes some privacy cost from the overall budget. Such an algorithm result in a problem of the trade-off between auxiliary and major tasks. If more privacy budgets are used for auxiliary task, then the major task may not have enough budget for accurate learning. In reverse, if few auxiliary budget is available for training, the major task can also be less accurate. In traditional private algorithms, a similar trade-off occurs between private hyper-parameter tuning and private training. Such a discussion may be beyond our focus. Thus, we leave it as an open problem for the future.

Softly Constrained Optimization for Scheduler

Directly optimizing the objective in Eq. (1) has two challenges. The first is gradient vanishing, and the more critical one is the requirement of T to be differentiable w.r.t. σ due to the coupling of σ and T. To tackle this issue, we reformulate the problem by defining the indicating function which is 1 on 0 input and 0 otherwise: $\tilde{F}(\sigma) = \sum_{t=1}^T \mathbb{I}_t f_t$, $\mathbb{I}_t = \mathbb{I}(h_t(\sigma;\rho_{\text{tot}}))$, where we assume $h_t(\sigma)$ could be zero at some integer t. Now, the hard constraint is implicitly embedded into the weights when T could be any constant such that the loss can converge. However, this objective is still non-differentiable w.r.t σ. To resolve the issue, let $\mathcal{I}(\cdot)$ be a differentiable approximation of the hard indicating function. Then, we rewrite the objective function: $\tilde{F}(\sigma) \approx \hat{F}(\sigma) = \sum_{t=1}^T \mathcal{I}_t f_t / \sum_{t=1}^T \mathcal{I}_t$, $\mathcal{I}_t = \mathcal{I}(h_t(\sigma))$ which is normalized to formulate a weighted average. The gradients over σ can be easily obtained as:

$$\partial\hat{F}(\sigma) = \frac{\sum_{t=1}^T \mathcal{I}_t \partial f_t}{\sum_{t=1}^T \mathcal{I}_t} + \frac{\sum_{t=1}^T (f_t - \hat{F})\partial\mathcal{I}_t}{\sum_{t=1}^T \mathcal{I}_t}, \qquad (3)$$

where we omit the partial differential denominator for brevity. Remarkably, the first term is the weighted gradient descent of utility objective while the second is to find the time when $f_t = \hat{F}$ or $f_t = f_{t+1}$ for t s.t. $\mathcal{I}_t > 0$. In practice, let T_ρ

be the number of steps when the privacy budget is just used up and we adopt a tent function: $\mathcal{I}(h_t(\sigma;\rho)) = \max\{1 - |h_t(\sigma;\rho)|/\rho, 0\}\mathbb{I}(t \in \mathcal{T})$ where $\mathcal{T} = \{T_\rho - \Delta T, \cdots, T_\rho + \Delta T - 1\}$ for some constant ΔT. The approximated function always includes ΔT steps within the weighted-average. Thus, the $\mathcal{I}_t \equiv 0$ will not be a possible solution.

Batch Algorithm for Long-Unrolling Optimization

One critical challenge in the L2P is the unrolling length, i.e., T (the number of iterations that the protector runs). To constrain the privacy cost within the budget, we need to unroll the optimization until the budget is used up and then back-propagate gradients from the last step to the first step. It is well known that sequential models, e.g., LSTM, will suffer from the the gradient vanishing. Moreover, the unrolled iterations has to be stored in memory waiting for back-propagation, which will consumes T times the space complexity of the model size. To mitigate the issue, in (Andrychowicz et al. 2016), a batch algorithm is used where a series of non-overlapped short spans of optimization are unrolled consecutively and optimized independently. For the projector, we can ignore the constraint by using a fixed scheduler and therefore the batch meta-optimization can be applied directly. In addition, we extend the idea to the constrained meta-optimization posed by the L2P.

Briefly, we decouple the averaged objective in Eq. (1) into B batches:

$$\bar{F}(\sigma, \pi, T) = \frac{1}{B}\sum\nolimits_{b=1}^{B} \bar{F}_i,\ \bar{F}_i = \frac{1}{|\mathcal{B}_b|}\sum\nolimits_{t\in\mathcal{B}_b} f_t, \quad (4)$$

where $\mathcal{B}_b$ is a batch of steps, namely a subset of $\{1, \cdots, T\}$ for $b \in \{1, \cdots, B\}$. Without loss of generality, we assume each batch is of the same size, consecutive and non-overlapped.

Batch augmented Lagrange algorithm. The augmented objective in Eq. (2) cannot be decomposed into batches, due to the quadratic term in the objective. We first break the constraint into B batch constraints and one global constraint. Now, let r be a set of constants $\{r_1, \cdots, r_B\}$, we can decompose the constraints into:

$$h_b \triangleq h(\sigma; r_b, \mathcal{B}_b) = f_C(\{\rho(\sigma_t)\}_{t\in\mathcal{B}_b}) - r_b,$$
$$h_r \triangleq h(r; \rho_{\text{tot}}) = f_C(\{r_b\}_{b=1}^{B}) - \rho_{\text{tot}}.$$

The decomposition enables us to use augmented Lagrange by introducing μ_b and z_b for each batch constraint. For r, we optimize the following simplified objective:

$$\mathcal{L}^{\text{aug}}(r) = \sum\nolimits_{b=1}^{B} \frac{\|\hat{\rho}_b - r_b\|_2^2}{2\mu_b} - z_r h_r + \frac{\|h_r\|_2^2}{2\mu_r} \quad (5)$$

where $\hat{\rho}_b = f_C(\{\rho(\sigma_t)\}_{t\in\mathcal{B}_b}) - z_b\mu_b$ and μ_b and z_b are the AL variables. The formulation and algorithm is straightforward as shown in appendix.

Softly-constrained batch meta-optimization. To save the memory, we want to immediately forget the used batch data but only keep the $\bar{F}_i$ and necessary variables. We find the first term in the augmented loss is the surrogate utility loss, since the $\hat{\rho}$ will increase for reducing the noise in batch and enhancing the utility as a result. Thus, there is no need to maintain the batch information except $\hat{\rho}_i$. Notice that in Eq. (3), the first term is merely the weighted-average utility loss gradients while the second term is due to the soft constraint. Thus, we construct a new objective to trade-off utility and constraint:

$$\mathcal{L}^{\text{aug}}(r) = \sum\nolimits_{b=1}^{B} \frac{1}{2\mu_b}\|\hat{\rho}_b - r_b\|_2^2 + \hat{F}(r), \quad (6)$$

$$\hat{F}(r) = \frac{\sum_{b\in\mathcal{T}_B}\mathcal{I}(h_b)\bar{F}_b}{\sum_{b\in\mathcal{T}_B}\mathcal{I}(h_b)} \quad (7)$$

where $\mathcal{T}_B = \{B_\rho - \Delta B, \cdots, B_\rho + \Delta B - 1\}$ and $T_\rho \in \mathcal{B}_{B_\rho}$ where B_ρ such that $\sum_{b=1}^{B_\rho-1}\rho_b - \rho \le 0$ and $\sum_{b=1}^{B_\rho-1}\rho_b - \rho > 0$. We assume $F_i(\sigma, \pi)$ is non-differentiable w.r.t. σ to avoid the back-propagation between batches. Practically, we use $\Delta B = 1$ to avoid involving too many batches, as using more batches requires a more strict stability of the loss. Therefore, the optimal condition is $\bar{F}_i = \hat{F}(r)$ or $\bar{F}_i = \bar{F}_j$ according to Eq. (3) with $\partial\bar{F}_i = 0$. Furthermore, we can show:

$$P\left(\left|\sum\nolimits_{t\in\mathcal{B}_j} f_t/|\mathcal{B}_i| - \sum\nolimits_{t\in\mathcal{B}_i} f_t/|\mathcal{B}_i|\right| > \epsilon\right) \le \frac{2\,\text{Var}[f_t]}{\epsilon^2|\mathcal{B}_i|},$$

which means the failure possibility of the optimal condition is decreased by the batch size. This is stabler than the non-batch algorithm (see appendix).

Experiments

In this section, we demonstrate the effectiveness of the L2P training. We study the following: the *optimality* of the proposed L2P on maximizing utility on the auxiliary datasets in comparison to the baselines; the *generalization* of L2P optimizers to tasks on data of different distributions; and its *scalability* on the different unrolling lengths. In the following experiments, we use a 2-layer coordinate-wise Long-Short Term Memory (LSTM) (Andrychowicz et al. 2016) both for $\pi(\cdot)$ and $\sigma(\cdot)$. Unless otherwise specified, 20 units of hidden variables are used in each LSTM and ctCDP is the privacy measurement. Implementation details and additional experiments are available in appendix.

Baseline Methods

We compare the L2P with four state-of-the-art differential privacy algorithms: a) SGD-Adv (Bassily, Smith, and Thakurta 2014), which achieves differential privacy by adding Gaussian noise to stochastic mini-batch gradients; b) SGD-MA (Abadi et al. 2016), which uses Moments Accountant to determine the variance of the Gaussian noise being added to the gradients; c) OutPert (Zhang et al. 2017), which first updates the parameters by a fixed number of vanilla gradient descent steps and then adds noise on the parameters, using up all privacy budget in a single step; d) ObjPert (Kifer, Smith, and Thakurta 2012), which adds linear perturbations to a convex loss functions; e) AGD (Lee and Kifer 2018), which adaptively determines the privacy budget for each update step during gradient descent for improved

utility. Notably, AGD is the state-of-the-art adaptive perturbation method which significantly outperforms other adaptive methods (Lee and Kifer 2018). Therefore, we only include AGD as the baseline of adaptive methods. Results of the non-private method (denoted as NonPrivate) will be shown as the upper bound of all DP algorithms. The non-private method optimizes the loss function using unperturbed gradients. Experimental setup and hyper-parameters tuning of baselines follows (Lee and Kifer 2018) and their code.

Learning Objectives and Datasets

We conduct experiments on both convex learning problems and non-convex ones. For convex we use logistic regression and for non-convex ones we use Multi-Layer Perceptron (MLP) (2 layers by default) with sigmoid activation and 20 units in the hidden layers. The learning tasks use above objectives to build binary classification models on three commonly used datasets, including IPUMS-BR, IPUMS-US and MNIST35. Under a given (ϵ, δ)-DP constraint, we compare the utility performance by test accuracies and final training losses averaged on 10 repetitions.

To simulate the realistic scenario when public auxiliary datasets are available and effective for improving the utility of private tasks, we design experiments using the following two datasets, IPUMS and MNIST35. In IPUMS, we consider the case of conducting analysis on detailed census data of a local community, which may leak residents' information. The other scenario is building a system using a handwritten dataset collected from a small group of individuals. In these cases, public census dataset and handwritten datasets are available and can be used as auxiliary datasets.

IPUMS. IPMUS-International census database (Ruggles et al. 2018) and its processed version is from (Lee and Kifer 2018). The dataset includes population surveys from different countries. Individual information is recorded in IPUMS-BR and IPUMS-US for two similar tasks, querying the range of individuals' monthly ($> \$300$) or annual ($> \$25,000$) income. Notably that they are from different distribution (Brazil (BR) and America (US), respectively) and disjoint. Specifically, IPUMS-US (IPUMS-BR) includes $40,000$ ($38,000$) records of 58 (53) features which are scaled into $[0, 1]$. We randomly select 20% (80%) of the data for testing (training).

MNIST35. The MNIST dataset (Lecun et al. 1998) includes $70,000$ gray-scale handwritten digits. The 28-by-28 images are vectorized and scaled into $[0, 1]$. Classifiers are applied for classifying digits. We use the official training-testing splits for evaluating the learning algorithms, and we only utilize two classes (e.g., digit 3 and 5) to build a binary classification task.

Protection Strategies Learned by L2P

Results on IPUMS dataset with SVMs. We compare our method with the AGD on losses and budget usage of iterations. SVMs are trained using L2P and AGD on the IPUMS-US dataset. The results are shown in Fig. 2 after 100 repetitions. The budget curve of L2P is more smooth as compared to AGD. AGD needs to query the objective value irregularly for budget, whereas L2P adaptively allocates budgets by leveraging historical information, and is therefore stabler.

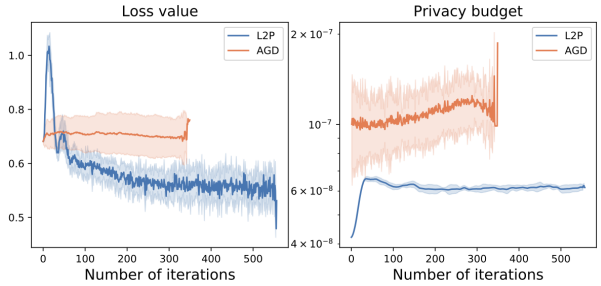

Figure 2: Training SVMs of $(0.05, 10^{-8})$-DP using a trained L2P protector and the AGD algorithm on the IPUMS-US dataset. The L2P protector is trained on the IPUMS-BR dataset. The iterate number for AGD is the times of gradient queries. Standard deviations are plotted as filled bonds.

In Fig. 2, a small amount of privacy budget is allocated by L2P at the beginning. It leads to larger noise in practice when the loss increases fast as witnessed in the loss curve. Also, this enables the protector to explore a wider range of the parameter space, and therefore this may help the optimizer get out of some local optimal regions and converge to a better solution. Due to the effort of the utility projector in L2P, the loss could be continuously lowered down while less privacy budget usage is needed than AGD. Specifically, when the privacy budget is inadequate, the L2P can flexibly allocate budgets and efficiently optimize the objective with private gradients. Because the step budgets are scheduled adaptively and the iterations will be terminated differently, the loss curves in the figure will abnormally fluctuate especially for AGD. The optimal condition of the soft-constrained objective (Eq. (3)) can be witnessed here. Namely, the L2P protector schedule the privacy budgets such that the optimization will not be terminated due to budget constraint until no obvious loss declines can be observed.

Generalization of L2P Optimizers

Given the L2P optimizer is trained on public auxiliary datasets, one critical concern is on the generalization of such an optimizer to new private dataset. In this section, we study the generalization of L2P to different private datasets. For clarity, there are three types of data in this study: 1) *Auxiliary meta-training data*, which should be publicly available (or with no privacy concern) and used for training protectors or tuning hyper-parameters; 2) *(Protected) private training data*, which serve as the training data for private learning tasks using the trained optimizer; 3) *Testing data* is where the privately trained models will be evaluated.

Setup. L2P optimizer is trained on the auxiliary dataset. For fair comparison, the auxiliary dataset is also used for pre-training models and validating hyper-parameters of other methods. The validation using public datasets was discussed in (Wu et al. 2017). Another protocol uses experience-based hyper-parameters, e.g., (Iyengar, Near, and Song 2019).

Generalization to different data distribution. We alternatively use one of IPUMS-US and IPUMS-BR as auxiliary datasets while the other is for private training. On the

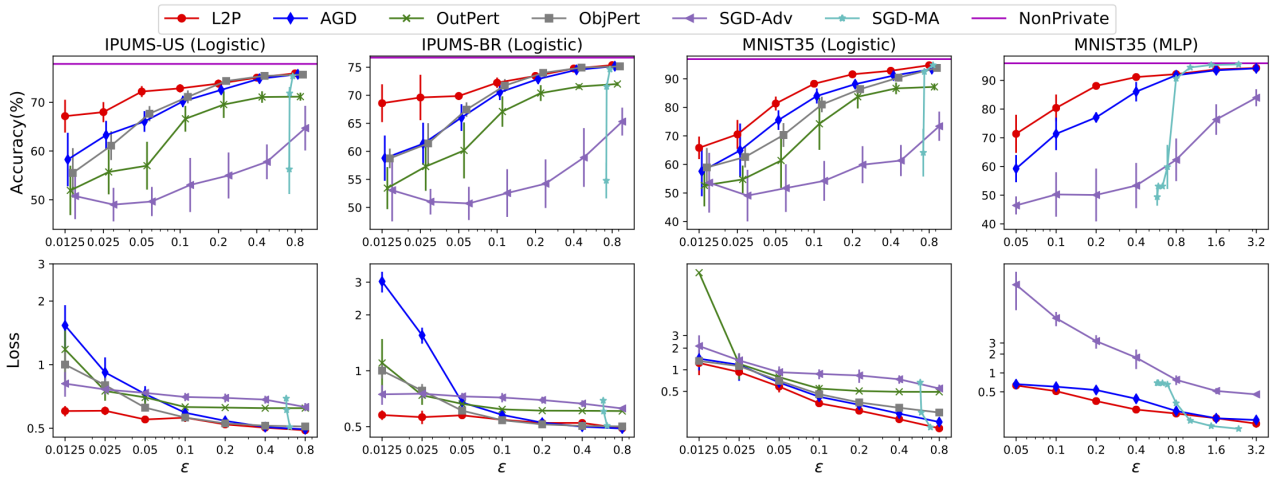

Figure 3: Test performance (top) and training loss values (bottom) by varying ϵ of logistic and MLP classifiers on IPUMS and MNIST35 datasets. The error bar presents the size of standard deviations. For better visualization, some horizontal offsets are added to every point.

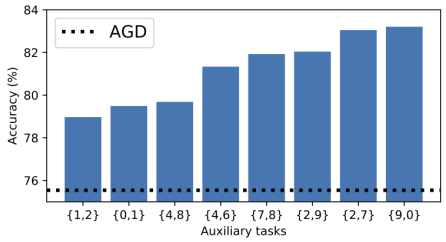

Figure 4: Performance on MNIST35 datasets using L2P protectors trained on varying auxiliary tasks. Each task contains two classes from the MNIST dataset.

MNIST2 dataset, The L2P protector is trained using all images of digits 4 and 6 and then is applied to learn to differentiate 3 and 5 (MNIST35). The protector is trained by a total of 50 batches of 20 steps and 100 epochs (1 epoch includes 1 scheduler update and 5 projector updates).

Results. We use the test accuracy and the final training loss value to gauge the utility performance. The privacy parameter δ is fixed as 10^{-8} while ϵ varies from 0.0125 to 0.8. Results are shown in Fig. 3. The non-private results are referenced as an upper bound of these algorithms. Remarkably, when the privacy requirement is high (i.e., low ϵ for differential privacy), L2P outperforms others with notable margins. When $\epsilon \geq 0.4$, the accuracies are less distinguishable for most methods. This is because that the DP noise is quite small when $\epsilon \geq 0.4$ and the adaptive strategies do not make an obvious difference. The SGD-MA is only effective in a narrow range of ϵ (Balle and Wang 2018) but has a good performance when ϵ is more than 0.8. Because that ϵ in SGD-MA is computed afterwards based on T, it has a different range.

Compatibility of auxiliary data. To test the influence of the auxiliary tasks, we report the performance on $\{3, 5\}$ when protectors are trained on different auxiliary datasets in Fig. 4 with $(0.05, 10^{-8})$-differential privacy using logistic models. Though all the protectors outperform the best baseline, 75.5% (AGD), we see that they are substantially influenced by the auxiliary tasks. The worst is achieved by $\{1, 2\}$ due to the obvious visual differences between the two tasks. We may conclude that when the auxiliary dataset is more visually like the private task, the transferring will perform better.

We also evaluate the optimizations for SVMs and quadratic problems in **??** . Since training optimizers for MLP is technically non-trivial due to the variety of parameter scales across different layers, for the purpose of reproducibility, we discussed the issue in appendix. The scalability of batch algorithms is discussed in appendix. Extra experiments are in the supplementary to show the generalization to different data domains.

Conclusion

In this paper, we proposed a Learning-to-Protect (L2P) framework, that learns a gradient protector from a set of auxiliary learning tasks, to protect the privacy and improve the utility during the learning of a sensitive task. Extensive experimental results showed that L2P outperformed hand-designed methods in small privacy budgets. Discussions, for example, the transferability of public tasks and the denoising effect of projectors, are included in **??** .

Acknowledgements

This material is based in part upon work supported by the National Science Foundation under Grant IIS-1749940 and Office of Naval Research N00014-20-1-2382.

References

Abadi, M.; Chu, A.; Goodfellow, I.; McMahan, H. B.; Mironov, I.; Talwar, K.; and Zhang, L. 2016. Deep Learning with Differential Privacy. In *CCS: Proceedings of the 2016*

ACM SIGSAC Conference on Computer and Communications Security, CCS '16, 308–318. New York, NY, USA: ACM.

Andrychowicz, M.; Denil, M.; Gómez, S.; Hoffman, M. W.; Pfau, D.; Schaul, T.; Shillingford, B.; and de Freitas, N. 2016. Learning to Learn by Gradient Descent by Gradient Descent. In *Advances in Neural Information Processing Systems 29*, 3981–3989. Curran Associates, Inc.

Balle, B.; and Wang, Y.-X. 2018. Improving the Gaussian Mechanism for Differential Privacy: Analytical Calibration and Optimal Denoising. In *International Conference on Machine Learning*, 394–403.

Barak, B.; Chaudhuri, K.; Dwork, C.; Kale, S.; McSherry, F.; and Talwar, K. 2007. Privacy, Accuracy, and Consistency Too: A Holistic Solution to Contingency Table Release. In *Proceedings of the Twenty-Sixth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '07, 273–282. New York, NY, USA: ACM.

Bassily, R.; Smith, A.; and Thakurta, A. 2014. Private Empirical Risk Minimization: Efficient Algorithms and Tight Error Bounds. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, 464–473.

Bernstein, G.; McKenna, R.; Sun, T.; Sheldon, D.; Hay, M.; and Miklau, G. 2017. Differentially Private Learning of Undirected Graphical Models Using Collective Graphical Models. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, 478–487. JMLR.org.

Bun, M.; Dwork, C.; Rothblum, G. N.; and Steinke, T. 2018. Composable and Versatile Privacy via Truncated CDP. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, 74–86. New York, NY, USA: ACM.

Bun, M.; and Steinke, T. 2016. Concentrated Differential Privacy: Simplifications, Extensions, and Lower Bounds. In *Theory of Cryptography*, volume 9985, 635–658. Berlin, Heidelberg: Springer Berlin Heidelberg.

Chaudhuri, K.; Monteleoni, C.; and Sarwate, A. D. 2011. Differentially Private Empirical Risk Minimization. *Journal of Machine Learning Research* 12(Mar): 1069–1109.

Chen, Y.; Hoffman, M. W.; Colmenarejo, S. G.; Denil, M.; Lillicrap, T. P.; Botvinick, M.; and de Freitas, N. 2016. Learning to Learn without Gradient Descent by Gradient Descent. *arXiv* .

Ding, B.; Kulkarni, J.; and Yekhanin, S. 2017. Collecting Telemetry Data Privately. In *Advances in Neural Information Processing Systems 30*, 3571–3580. Curran Associates, Inc.

Dwork, C. 2006. Differential Privacy. In *Automata, Languages and Programming*, Lecture Notes in Computer Science, 1–12. Springer Berlin Heidelberg.

Dwork, C. 2008. Differential Privacy: A Survey of Results. In *Theory and Applications of Models of Computation*, Lecture Notes in Computer Science, 1–19. Springer Berlin Heidelberg.

Dwork, C.; Kenthapadi, K.; McSherry, F.; Mironov, I.; and Naor, M. 2006a. Our Data, Ourselves: Privacy Via Distributed Noise Generation. In *Advances in Cryptology - EUROCRYPT 2006*, Lecture Notes in Computer Science, 486–503. Berlin, Heidelberg: Springer.

Dwork, C.; McSherry, F.; Nissim, K.; and Smith, A. 2006b. Calibrating Noise to Sensitivity in Private Data Analysis. In *Theory of Cryptography*, Lecture Notes in Computer Science, 265–284. Springer Berlin Heidelberg.

Dwork, C.; and Roth, A. 2013. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends® in Theoretical Computer Science* 9(3-4): 211–407.

Erlingsson, Ú.; Pihur, V.; and Korolova, A. 2014. RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, 1054–1067. Scottsdale, Arizona, USA: Association for Computing Machinery.

Fredrikson, M.; Jha, S.; and Ristenpart, T. 2015. Model Inversion Attacks That Exploit Confidence Information and Basic Countermeasures. In *CCS: Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, 1322–1333. New York, NY, USA: ACM.

Gupta, A.; Ligett, K.; McSherry, F.; Roth, A.; and Talwar, K. 2010. Differentially Private Combinatorial Optimization. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, Proceedings, 1106–1125. Society for Industrial and Applied Mathematics.

Hay, M.; Li, C.; Miklau, G.; and Jensen, D. 2009. Accurate Estimation of the Degree Distribution of Private Networks. In *2009 Ninth IEEE International Conference on Data Mining*, 169–178.

Hong, J.; Chen, H.; and Lin, F. 2018. Disturbance Grassmann Kernels for Subspace-Based Learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, 1521–1530. New York, NY, USA: ACM.

Iyengar, R.; Near, J. P.; and Song, D. 2019. Towards Practical Differentially Private Convex Optimization. In *2019 IEEE Symposium on Security and Privacy (SP)*, volume 1, 1–18.

Kifer, D.; Smith, A.; and Thakurta, A. 2012. Private Convex Empirical Risk Minimization and High-Dimensional Regression. In *Proceedings of the 25th Annual Conference on Learning Theory, COLT '12*, 40.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In *The 3rd International Conference for Learning Representations*. San Diego, CA.

Lecun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE* 86(11): 2278–2324.

Lee, J.; and Kifer, D. 2018. Concentrated Differentially Private Gradient Descent with Adaptive Per-Iteration Privacy Budget. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, 1656–1665. New York, NY, USA: ACM.

Li, J.; Khodak, M.; Caldas, S.; and Talwalkar, A. 2020. Differentially Private Meta-Learning. *International Conference on Learning Representations* .

Li, K.; and Malik, J. 2017. Learning to Optimize Neural Nets. *arXiv:1703.00441 [cs, math, stat]* .

Maaten, L.; Chen, M.; Tyree, S.; and Weinberger, K. Q. 2013. Learning with Marginalized Corrupted Features. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 410–418.

Mironov, I. 2017. Rényi Differential Privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, 263–275. Santa Barbara, CA, USA: IEEE.

Nocedal, J.; and Wright, S. J. 1999. *Numerical Optimization*. Springer Series in Operations Research. New York: Springer.

Papernot, N.; Song, S.; Mironov, I.; Raghunathan, A.; Talwar, K.; and Erlingsson, Ú. 2018. Scalable Private Learning with PATE. *International Conference on Learning Representations* .

Rényi, A. 1961. On Measures of Entropy and Information. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*. The Regents of the University of California.

Rubinstein, B. I. P.; Bartlett, P. L.; Huang, L.; and Taft, N. 2012. Learning in a Large Function Space: Privacy-Preserving Mechanisms for SVM Learning. *Journal of Privacy and Confidentiality* .

Ruggles, S.; Katie Genadek; Ronald Goeken; Josiah Grover; and Matthew Sobek. 2018. Integrated Public Use Microdata Series, Minnesota Population Center. http://international.ipums.org.

Talwar, K.; Guha Thakurta, A.; and Zhang, L. 2015. Nearly Optimal Private LASSO. In *Advances in Neural Information Processing Systems 28*, 3025–3033. Curran Associates, Inc.

Tang, J.; Korolova, A.; Bai, X.; Wang, X.; and Wang, X. 2017. Privacy Loss in Apple's Implementation of Differential Privacy on MacOS 10.12. *arXiv:1709.02753 [cs]* .

Wager, S.; Wang, S.; and Liang, P. S. 2013. Dropout Training as Adaptive Regularization. In *Advances in Neural Information Processing Systems 26*, 351–359. Curran Associates, Inc.

Wang, D.; and Xu, J. 2019. Differentially Private Empirical Risk Minimization with Smooth Non-Convex Loss Functions: A Non-Stationary View. *Proceedings of the AAAI Conference on Artificial Intelligence* 33(01): 1182–1189.

Wang, D.; Ye, M.; and Xu, J. 2017. Differentially Private Empirical Risk Minimization Revisited: Faster and More General. In *Advances in Neural Information Processing Systems 30*, 2722–2731. Curran Associates, Inc.

Wang, Y.-X.; Balle, B.; and Kasiviswanathan, S. 2020. Subsampled Rényi Differential Privacy and Analytical Moments Accountant. *Journal of Privacy and Confidentiality* 10(2).

Williams, O.; and Mcsherry, F. 2010. Probabilistic Inference and Differential Privacy. In *Advances in Neural Information Processing Systems 23*, 2451–2459. Curran Associates, Inc.

Wu, X.; Li, F.; Kumar, A.; Chaudhuri, K.; Jha, S.; and Naughton, J. 2017. Bolt-on Differential Privacy for Scalable Stochastic Gradient Descent-Based Analytics. In *Proceedings of the 2017 ACM International Conference on Management of Data*, SIGMOD '17, 1307–1322. New York, NY, USA: ACM.

Yu, L.; Liu, L.; Pu, C.; Gursoy, M. E.; and Truex, S. 2019. Differentially Private Model Publishing for Deep Learning. *proceedings of 40th IEEE Symposium on Security and Privacy* .

Zhang, J.; Xiao, X.; Yang, Y.; Zhang, Z.; and Winslett, M. 2013. PrivGene: Differentially Private Model Fitting Using Genetic Algorithms. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, SIGMOD '13, 665–676. New York, NY, USA: ACM.

Zhang, J.; Zheng, K.; Mou, W.; and Wang, L. 2017. Efficient Private ERM for Smooth Objectives. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, 3922–3928. AAAI Press.

Zhou, Y.; Wu, Z. S.; and Banerjee, A. 2020. Bypassing the Ambient Dimension: Private SGD with Gradient Subspace Identification. *arXiv:2007.03813 [cs, stat]* .