Improve the Gradient Perturbation Approach for Differentially Private Optimization

Da Yu *
School of Data and Computer Science
Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou, P.R.China
yuda3@mial2.sysu.edu.cn

Huishuai Zhang
Microsoft Research
Beijing, 100080
Huishuai.Zhang@microsoft.com

Wei Chen
Microsoft Research
Beijing, 100080
wche@microsoft.com

Abstract

We investigate the gradient perturbation approach for differentially private (DP) optimization and improve its performance by utilizing the optimization properties of gradient descent. One property is that historical gradient are often used as a momentum to facilitate the convergence. However, momentum gradient descent with usual coefficient does not perform well in the DP case. Instead, we specifically choose a coefficient based on our analysis. Another property is that the gradient tends to zero as optimization algorithm converges. Inspired by this, we intentionally decrease the gradient norm clip bound, which corresponds to decrease the sensitivity in the differential privacy computation. This allows us to run more training steps while keeping the same privacy guarantee. Our approaches are applicable for both convex and non-convex optimizations with differential privacy constraint. We demonstrate that our approaches can significantly improve the training/test performance under a privacy budget via various experiments.

1 Introduction

Machine learning has achieved great success across many domains including computer vision, natural language processing, etc. However, the model learned from training data may leak unexpected information of individual samples (Fredrikson et al., 2015; Shokri et al., 2017), which could violate laws and contracts of protecting user's privacy. One provable way to guarantee the training model with individual privacy is running the training algorithm with differential privacy constraint. Differential privacy (Dwork et al., 2006a,b) is a well-defined and rigorous standard for privacy. By its definition, differential privacy focuses on eliminating the influence of individual participants. Thus a differential privacy algorithm learns the statistical information of the population, rather than the individual, which makes it become a golden rule for protecting the information of individual user.

Usually, the machine learning problem is often formulated as an optimization problem

$$arg \min_{\theta \in \mathbb{R}^p} \sum_{x_i \in D} f(\theta; x_i), \tag{1}$$

where D is the dataset, f is the loss function and θ is the model parameter to optimize. We note that $\sum_{x_i \in D} f(\theta; x_i)$ is the objective and the solution $\hat{\theta}$ to the problem (1) is called the output.

*Work done while this author was an intern at Microsoft Research.

Differentially private algorithm should have indistinguishable outputs when there is a small change in the input dataset.

Specifically, three approaches have been proposed for differentially private optimization: *Objective Perturbation*, *Output Perturbation* and *Gradient Perturbation*.

Objective Perturbation (Chaudhuri et al., 2011; Kifer et al., 2012; Iyengar et al.) perturbs the objective function by adding a random regularization term and releases the minima of the perturbed objective. When the objective function is (strongly) convex, objective perturbation achieves certain differential privacy guarantee. Output Perturbation (Wu et al., 2017; Zhang et al., 2017) perturbs the output parameters after the training procedure where the variance of the adding noise corresponds the maximum distance of the output parameters for any two neighboring input datasets. The differential privacy guarantee is also be achieved when the objective is (strongly) convex. Intuitively, the more stable the optimization algorithm is, the less the noise is injected.

Although *Objective perturbation* and *Output perturbation* can reach a theoretical lower bound of utility cost when objective is (strongly) convex (Bassily et al., 2014), *Gradient perturbation* still receives a lot of attention because it admits public training process which is necessary for distributed optimization setting. Moreover, gradient perturbation can achieve differential privacy guarantee even for nonconvex objectives, which are ubiquitous in modern machine learning applications.

Gradient Perturbation (Bassily et al., 2014; Abadi et al., 2016) perturbs the released gradient in each training step. Differential privacy of the whole training procedure can be guaranteed by applying the *composition property* if each release is differentially private. In this paper, we improve the performance of gradient perturbation approach by employing the optimization properties of gradient descent when training machine learning models. Our main contribution contains:

- We propose to utilize the historical released noisy gradient as a momentum to facilitate the training as people usually do in the non-private case. We analyze the variance of noisy momentum and suggest that the momentum coefficient shouldn't be chosen the same as in the non-private case. Then we verify the improvement via empirical study.
- Based on the fact that the gradient magnitude tends to zero as the algorithm converges, we propose to decrease the gradient norm clip bound over the training process. This allows us to train the model with more steps for the same privacy budget and hence could achieve better performance. Empirical study verifies the effectiveness.

2 Our algorithm

Our algorithm, Algorithm 1, is based on the DP-SGD algorithm (Abadi et al., 2016) with two new components *utilizing historical noisy gradients* and *adjusting gradient norm clip bound*. We use the moments accountant (Abadi et al., 2016) method to track the privacy cost of the training process.

We briefly introduce how moments accountant works. It defines a privacy loss (random variable) at each step and then the overall privacy loss is upper bounded by the sum of privacy loss of each step via the composition property. Specifically, for two neighboring data sets $D, D^{'}$ (add/remove one example from D), auxiliary input aux, an outcome o, a mechanism $\mathcal{M}$. The privacy loss at o is

$$c(o; \mathcal{M}, aux, D, D^{'}) \triangleq \log \left(\frac{Pr[\mathcal{M}(aux, D)] = o}{Pr[\mathcal{M}(aux, D')] = o} \right), \tag{2}$$

where the auxiliary input aux is the output of all the previous mechanisms, i.e., all previous noisy gradients in our case. The corresponding moment generating function (log version) at λ is given by

$$\alpha_{\mathcal{M}}(\lambda; aux, D, D^{'}) \triangleq \log \mathbb{E}_{o \sim \mathcal{M}(aux, D)}[\exp(\lambda c(o; \mathcal{M}, aux, D, D^{'}))]. \tag{3}$$

To guarantee the differential privacy, it's necessary to bound all possible $\alpha_{\mathcal{M}}(\lambda; aux, D, D^{'})$, and hence one defines

$$\alpha_{\mathcal{M}}(\lambda) \triangleq \max_{aux, D, D^{'}} \alpha_{\mathcal{M}}(\lambda; aux, D, D^{'}). \tag{4}$$

In Algorithm 1, $\alpha_{\mathcal{M}}(\lambda)$ can be calculated based on the gradient norm bound C, noise variacne σ and sampling probability q. Then the moment generating function of the overall privacy loss is upper bounded by $\alpha_{\mathcal{M}}(\lambda) \leq \sum_{i=1}^{k} \alpha_{\mathcal{M}_i}(\lambda)$, where $\mathcal{M}_i$ is the random mechanism at step i.

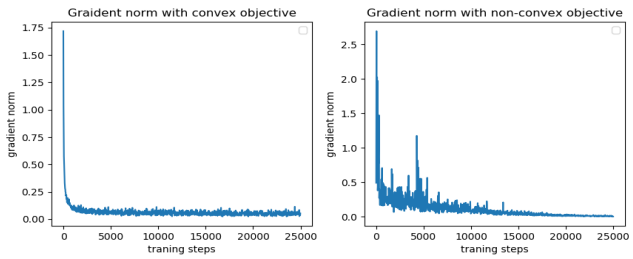

Figure 1: The trend of gradient norm during training for non-convex and convex models.

In the end, the parameters (ϵ, δ) of differentially private mechanism M can be computed by $\delta = \min_{\lambda} \exp(\alpha_{\mathcal{M}}(\lambda) - \lambda\epsilon)$ for any $\epsilon > 0$. More details are available in Abadi et al. (2016).

Utilizing historical noisy gradients: In machine learning, it's common to use momentum to facilitate the convergence of gradient based algorithms. Gradient perturbation approach also desires fast convergence speed because for a given privacy budget, faster convergence implies larger budget at each step, and consequently smaller noise and better accuracy.

Momentum is initialized as $m_0 = 0$. Let $\widetilde{g}_t$ be the noisy gradient at step t, we update momentum via the following formula:

$$m_t = cm_{t-1} + \widetilde{g}_t, \qquad \text{where } c \in (0, 1). \tag{5}$$

Let $b_t \sim \mathcal{N}(0, \sigma^2)$ be the noise sample at each step. Use n_t denotes the overall noise in m_t, then $n_t = c^0 b_t + \ldots + c^{t-2} b_2 + c^{t-1} b_1$. It's easy to check that the variance of n_t is

$$\sigma_t^2 = \frac{1 - c^{2t}}{1 - c^2} \sigma^2$$

If momentum coefficient c is large , the variance at each step is also large.(i.e. $\sigma_t^2 \approx 5\sigma^2$ when use usual coefficient 0.9). If c is small, we are not able to leverage the effectiveness of momentum. We study the influence of different coefficients empirically in the appendix.

Adjusting the gradient norm clip bound: Abadi et al. (2016) uses the same gradient norm clip bound across the whole training process. However, as the training goes, the gradient tends to zero for convergent algorithms (see Figure 1). With the recognition of this property, we decrease the clip bound gradually during training while keeping the variance of noise the same. This saves privacy budget and allows us to train more steps.

In our experiment, we use a linear strategy to decrease gradient norm clip bound. Let T be the total running steps and C be the initial clip bound. We set gradient norm clip bound at step t as

$$C_t = \frac{C}{\min(2, 1 + \frac{t}{T})}, \tag{6}$$

so that it decreases the clip bound linearly to $1/2$ of initial value.

3 Experiments

We compare our algorithms with DP-SGD to demonstrate the effectiveness of using the momentum and the varying clip bound in both non-convex and convex setting.

For non-convex objective, we use a shallow neural network consisting of two convolution layers with rectifier linear unit (ReLU) activation and a fully connected layer followed by a softmax layer. The output channels of two convolution layers are 10 and 20, respectively, and the convolution kernels are all with size 5×5. The fully connected layer has input dimension 50 and output dimension 10. For the convex case, we use a fully connected layer (784×10) followed by a softmax layer.

The hyperparameters are same for both convex and non-convex models. Learning rate is set as 0.02 and sampling probability is 0.01. The momentum coefficient is set as 0.6 based on our observation. For given privacy parameters, we choose the minimum variance(with precision 0.01) for algorithm

Algorithm 1: Amended DP-SGD with momentum and varying clip

Input : Data set: $D = \{x_1, \ldots, x_n\}$. Loss fucntion: $\sum \frac{1}{n} f(\theta, x_i)$. Privacy parameters:$(\epsilon, \delta)$. Sampling probability q. Target epoches T. Gradient norm bound C. Learning rate η.

Output : Differentially private parameter set θ

Initialize $t = 0$, momentum $m_0 = 0^p$ and randomly initialize θ_t

Initialize $\widetilde{\sigma}$ based on privacy parameters and q, T

$\sigma = C\widetilde{\sigma}$ and $C_0 = C$

while $true$ **do**

  Compute privacy cost α_t using variance $\frac{C}{C_t}\widetilde{\sigma}$.

  Cast α_t into (ϵ, δ_t)-differential privacy.

  if $\delta_t > \delta$ **then**

    Output parameter set θ_t with guarantee of (ϵ, δ)-differential privacy and return.

  end

  Choose each $x_i \in D$ with probability q to construct minibatch J_t.

  $b_t \sim \mathcal{N}(0, \sigma^2 I_{p\times p})$

  for *each* $x_i \in J_t$ **do**

    $g_t(x_i) = \nabla_{\theta_t} f(\theta, x_i)$

    $\overline{g}_t(x_i) \leftarrow g_t(x_i)/\max(1, \frac{\|g_t(x_i)\|}{C_t})$ *//clip each* $g_t(x_i)$ *using* C_t

  end

  $\widetilde{g}_t = \frac{1}{|J_t|}(\sum_{x_i \in J_t} \overline{g}_t(x_i) + b_t)$ *//add noise so we can differentially private release* $\widetilde{g}_t$

  Update m_t use formula (5). Adjust gradient norm bound use (6)

  $\theta_{t+1} = \theta_t - \eta m_t$, $t = t + 1$

end

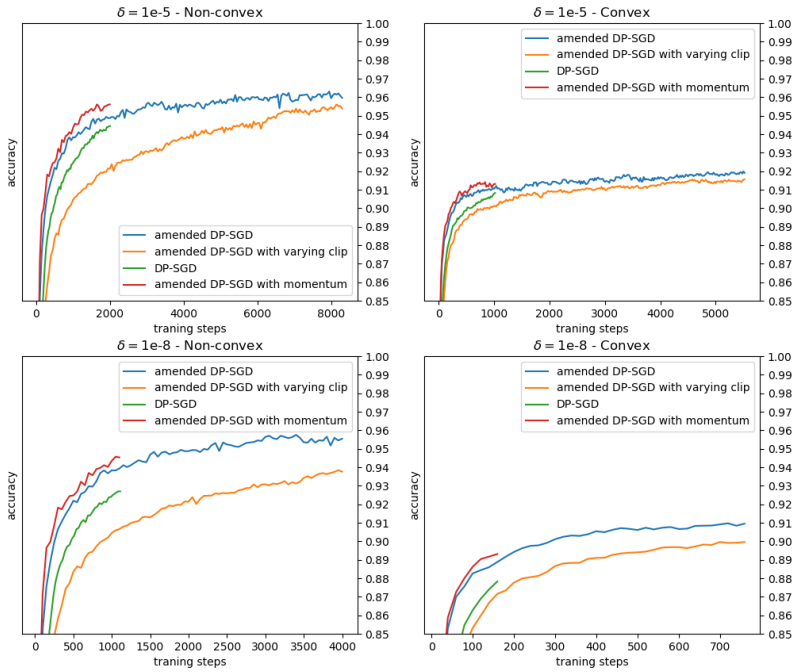

Figure 2: The performance of our algorithms, $\epsilon = 2$.

to run target epochs. We use a linearly decreasing function to vary the clip bound. It is initialized as 12 and decreased to 6 linearly in target epochs and stays unchanged after that. We compare the validation curves of the amended DP-SGD with momentum, the amended DP-SGD with varying clip bound, the amended DP-SGD and the original DP-SGD, which are depicted in Figure 2. As shown in Figure 2, the advantage of our algorithm becomes larger when privacy budget (running steps) is more constrained.

4 Acknowledgments

This work is supported by the National Key R&D Program of China (2018YFB1004404), National Natural Science Foundation of China (61472453, U1401256,U1501252,U1611264,U1711261, U1711262)

References

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *ACM SIGSAC Conference on Computer and Communications Security*, 2016.

Raef Bassily, Adam Smith, and Abhradeep Thakurta. Differentially private empirical risk minimization: Efficient algorithms and tight error bounds. *Annual Symposium on Foundations of Computer Science*, 2014.

Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 2011.

Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2006a.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, 2006b.

Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *ACM SIGSAC Conference on Computer and Communications Security*, 2015.

Roger Iyengar, Joseph P Near, Dawn Song, Om Thakkar, Abhradeep Thakurta, and Lun Wang. Towards practical differentially private convex optimization. In *IEEE Symposium on Security and Privacy*.

Daniel Kifer, Adam Smith, and Abhradeep Thakurta. Private convex empirical risk minimization and high-dimensional regression. In *Conference on Learning Theory*, 2012.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *IEEE Symposium on Security and Privacy (SP)*, 2017.

Xi Wu, Fengan Li, Arun Kumar, Kamalika Chaudhuri, Somesh Jha, and Jeffrey Naughton. Bolt-on differential privacy for scalable stochastic gradient descent-based analytics. In *ACM International Conference on Management of Data*, 2017.

Jiaqi Zhang, Kai Zheng, Wenlong Mou, and Liwei Wang. Efficient private erm for smooth objectives. *arXiv preprint arXiv:1703.09947*, 2017.

Appendix A The choices of momentum coefficient

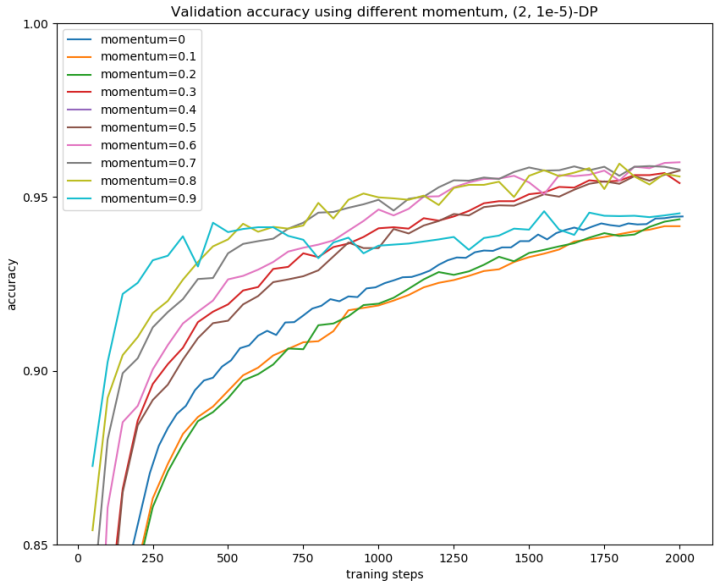

Figure 3: Validation accuracy in training process with different momentum coefficients. When $c = 0.9$, where $\sigma_t^2 \approx 5\sigma^2$, we see that the validation accuracy increases faster in the beginning but retains a relatively bad one finally. The best accuracy is achieved for $c = 0.6, 0.7$, which can leverage the effect of momentum and at the same time does not increase variance much.

Appendix B Performance comparison in a more restricted privacy setting

We compare our algorithm with DP-SGD in more restricted privacy setting ($\epsilon = 0.5, \delta = \frac{1}{n^2}$), where n is the number of training examples. For neighboring datasets, DP-SGD assumes one example is *deleted/added* in original dataset. We run the new experiments use a more popular definition(the same as Iyengar et al. and Wu et al. (2017)): one example is *modified* in original dataset. The new definition increase the sensitivity from C to $2C$, and thus bigger variance is needed at each step.

We run the algorithms on mnist and a new dataset *covertype*, which contains 581012 examples. Each example has 54 features and 7 labels. The model and hyperparameters for mnist are the same as before(except the strategy for varying clip). For covertype, we use a fully connected neural network consisting of two hidden layers(100 neurons in each layer) with tanh activation and a output fully connected layer followed by a softmax layer. Learning rate is 0.1 and sampling probability is 0.001. Clip bound is initialized as 8. Momentum is 0.6. Clip bound shrinks $1/1000$ at each step and remains unchanged when reach $1/2$ of the initial value. Figure 4 shows the results for two algorithms.

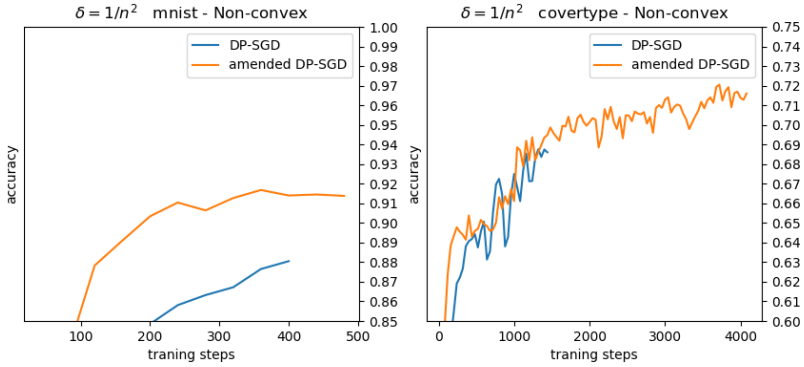

Figure 4: Comparison between our amended DP-SGD and DP-SGD in a more restricted privacy setting. Our algorithm achieves $\sim$91.5% accuracy on mnist with $(0.5, 1/n^2)$-DP.