Deep Learning with Differential Privacy

Martn Abadi[1] H. Brendan McMahan[1] Andy Chu[1] Ian Goodfellow[2] Ilya Mironov[1] Kunal Talwar[1] Li Zhang[1]

[1]Google

[2]OpenAI

CCS, 2016
Presenter: Bargav Jayaraman

Outline

Introduction

- Deep learning models are successfully used in many computer vision and NLP tasks
- These datasets may contain sensitive information about individuals that can be revealed by the learned models
- Private learning of deep learning models is hence required which does not destroy the utility of the models
- This work aims to achieve the best utility bound while guaranteeing a strong notion of privacy called *differential privacy*

Differential Privacy Background

Definition ((ϵ, δ)-Differential Privacy)

A randomized mechanism $M : D \rightarrow R$ with domain D and range R satisfies (ϵ, δ)-differential privacy if for any two adjacent inputs $d, d' \in D$ and for any subset of outputs $S \subseteq R$ it holds that

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

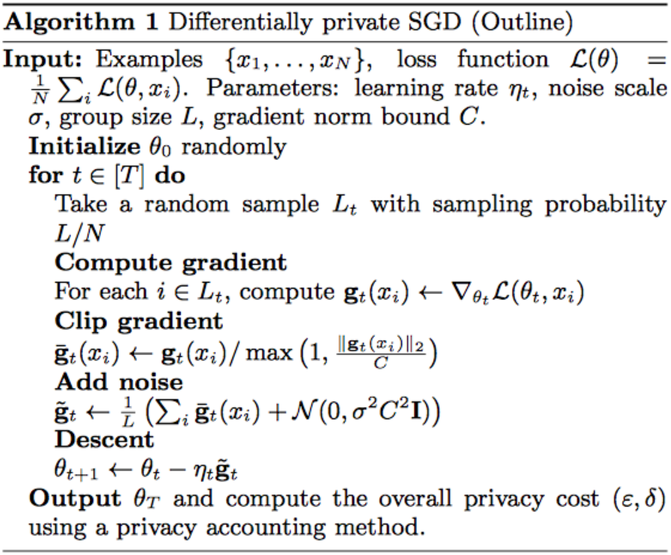
- Gaussian noise mechanism is defined by

$$\mathcal{M}(d) = \Delta f(d) + \mathcal{N}(0, S_f^2.\sigma^2)$$

  where $S_f = |f(d) - f(d')|$ is the sensitivity of f and $N(0, S_f^2.\sigma^2)$ is the Gaussian distribution with mean 0 and standard deviation $S_f\sigma$.
- Satisfies (ϵ, δ)-differential privacy if $\delta \geq \frac{4}{5}\exp(-(\sigma\epsilon)^2/2)$ and $\epsilon < 1$.

Outline

Differentially Private SGD Algorithm



Algorithm 1 Differentially private SGD (Outline)

```
Input: Examples $\{x_1, \ldots, x_N\}$, loss function $\mathcal{L}(\theta) = \frac{1}{N}\sum_i \mathcal{L}(\theta, x_i)$. Parameters: learning rate $\eta_t$, noise scale $\sigma$, group size $L$, gradient norm bound $C$.
  Initialize $\theta_0$ randomly
  for $t \in [T]$ do
    Take a random sample $L_t$ with sampling probability $L/N$
    Compute gradient
    For each $i \in L_t$, compute $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t}\mathcal{L}(\theta_t, x_i)$
    Clip gradient
    $\bar{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i) / \max\left(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C}\right)$
    Add noise
    $\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L}\left(\sum_i \bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$
    Descent
    $\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$
  Output $\theta_T$ and compute the overall privacy cost $(\varepsilon, \delta)$ using a privacy accounting method.
```

Differentially Private SGD Components

- **Norm clipping** - Gradient vector g is replaced by $g / \max(1, \frac{\|g\|_2}{C})$ for a clipping threshold C.
- **Lots** - Group of examples batched together for noise addition. Intuition - variance decreases as the group size increases. Size of a lot is L.
- **Privacy accounting** - Privacy loss is calculated at each step of the algorithm and accumulate to bound the overall privacy loss of the algorithm.
- **Moments accountant** - Achieves a lower bound on the privacy budget of the overall algorithm compared to the existing methods of bounding the overall budget.

Outline

Moments Accountant

- Each lot is (ϵ, δ)-DP if $\sigma = \sqrt{2 \log \frac{1.5}{\delta}} / \epsilon$ for Gaussian noise.
- Thus, each step is $(O(q\epsilon), q\delta)$-DP over the dataset, where $q = \frac{L}{N}$ is the sampling probability of a lot over the dataset.
- Over T iterations, naive composition gives the bound of $(O(qT\epsilon), qT\delta)$-DP.
- Over T iterations, strong composition gives the bound of $(O(q\epsilon\sqrt{T \log \frac{1}{\delta}}), qT\delta)$-DP.
- Over T iterations, moments accountant gives a tighter bound of $(O(q\epsilon\sqrt{T}), \delta)$-DP.

Moments Accountant

Theorem (1)

There exist constants c_1 and c_2 so that given the sampling probability $q = L/N$ and the number of steps T, for any $\epsilon < c_1 q^2 T$, algorithm is (ϵ, δ)-differentially private for any $\delta > 0$ if we choose

$$\sigma \geq c_2 \frac{q\sqrt{T \log \frac{1}{\delta}}}{\epsilon}$$

If we use the strong composition theorem, we will then need to choose $\sigma = \Omega(q\sqrt{T log(1/\delta) log(T/\delta)}/\epsilon)$.
For $L = 0.01N$, $\sigma = 4$, $\delta = 10^{-5}$, and $T = 10000$, we have $\epsilon \approx 1.26$ using the moments accountant, and $\epsilon \approx 9.34$ using the strong composition theorem.

Outline

Moments Accountant Details

Privacy Loss

For neighboring databases $d, d' \in \mathcal{D}^n$, a mechanism $\mathcal{M}$, auxiliary input aux, and an outcome $o \in \mathcal{R}$, privacy loss is defined at o as

$$c(o; \mathcal{M}, \mathrm{aux}, d, d') = \log \frac{\Pr[\mathcal{M}(\mathrm{aux}, d) = o]}{\Pr[\mathcal{M}(\mathrm{aux}, d') = o]}$$

Moment Generating Function

For a given mechanism $\mathcal{M}$, we define the λ^{th} moment $\alpha_{\mathcal{M}}(\lambda; \mathrm{aux}, d, d')$ as the log of the moment generating function evaluated at the value λ:

$$\alpha_{\mathcal{M}}(\lambda; \mathrm{aux}, d, d') = \log \mathbb{E}_{o \sim \mathcal{M}(\mathrm{aux}, d)}[\exp(\lambda c(o; M, \mathrm{aux}, d, d'))]$$

Overall moment is given as:

$$\alpha_{\mathcal{M}}(\lambda) = \max_{\texttt{aux, d, d'}} \alpha_{\mathcal{M}}(\lambda; \mathrm{aux}, d, d')$$

Moments Accountant Details

Theorem (2)

*[**Composability**] Suppose that a mechanism $\mathcal{M}$ consists of a sequence of adaptive mechanisms $\mathcal{M}_1, \ldots, \mathcal{M}_k$ where $\mathcal{M}_i : \prod_{j=1}^{i-1} \mathcal{R}_j \times \mathcal{D} \to \mathcal{R}_i$. Then, for any λ*

$$\alpha_{\mathcal{M}}(\lambda) \leq \sum_{i=1}^{k} \alpha_{\mathcal{M}_i}(\lambda)$$

*[**Tail bound**] For any $\epsilon > 0$, the mechanism $\mathcal{M}$ is (ϵ, δ)-differentially private for*

$$\delta = \min_{\lambda} \exp(\alpha_{\mathcal{M}}(\lambda) - \lambda\epsilon)$$

Deriving Moments bound for Gaussian

Let μ_0 denote the pdf of $\mathcal{N}(0, \sigma^2)$, and μ_1 denote the pdf of $\mathcal{N}(1, \sigma^2)$. Let μ be the mixture of two Gaussians $\mu = (1-q)\mu_0 + q\mu_1$. Then $\alpha(\lambda) = \log \max(E_1, E_2)$ where

$$E_1 = E_{z\sim\mu_0}[(\mu_0(z)/\mu(z))^\lambda]$$

$$E_2 = E_{z\sim\mu}[(\mu(z)/\mu_0(z))^\lambda]$$

In implementation, $\alpha(\lambda)$ is computed via numerical integration.
In addition, the asymptotic bound is given as:

$$\alpha(\lambda) \leq q^2\lambda(\lambda+1)/(1-q)\sigma^2 + O(q^3/\sigma^3)$$

Together with this bound and Theorem 2, we get the proof of Theorem 1.

Implementation

- DPSGD[1] consists of: `Sanitizer` and `Privacy Accountant`.
- `Sanitizer` performs two operations: (1) limit the sensitivity of each individual example by clipping the norm of its gradient; and (2) add noise to the gradient of a batch before updating the network parameters.
- `Privacy Accountant` keeps track of privacy spending over the course of training.
- Differentially Private PCA: Gaussian noise is added to normalized covariance matrix and then the principal components are obtained.
- Convolutional Layers: Learned on public data, and based on GoogLeNet or AlexNet features for image models.

[1] `https://github.com/tensorflow/models/`

Experimental Results

- Experiments done on:
  - MNIST : 60,000 train and 10,000 test
  - CIFAR-10 : 50,000 train and 10,000 test
- Model uses a 60-dimensional PCA projection layer and a single hidden layer with 1,000 ReLu hidden units.
- Parameters are set as $q = 0.01$, $\sigma = 4$, and $\delta = 10^{-5}$, $L = 600$.
- The learning rate is set at 0.1 initially and linearly decreased to 0.052 over 10 epochs and fixed thereafter.

Applying Moments Accountant

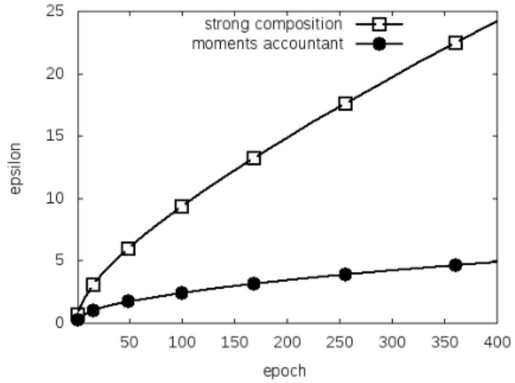
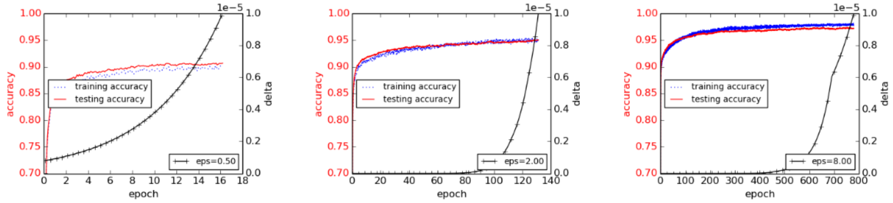


Figure 2: The ε value as a function of epoch E for $q = 0.01, \sigma = 4, \delta = 10^{-5}$, using the strong composition theorem and the moments accountant respectively.

MNIST Results



(1) Large noise (2) Medium noise (3) Small noise

Figure 3: Results on the accuracy for different noise levels on the MNIST dataset. In all the experiments, the network uses 60 dimension PCA projection, 1,000 hidden units, and is trained using lot size 600 and clipping threshold 4. The noise levels (σ, σ_p) for training the neural network and for PCA projection are set at (8, 16), (4, 7), and (2, 4), respectively, for the three experiments.

MNIST Results

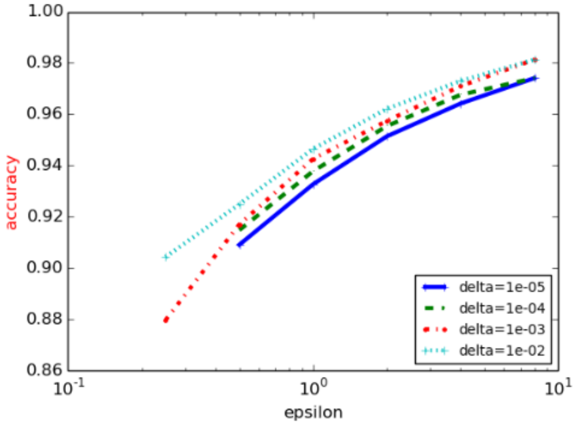
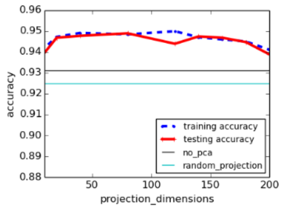
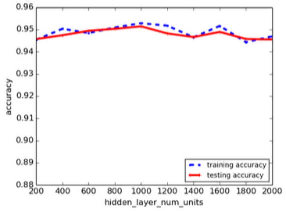


Figure 4: Accuracy of various (ε, δ) privacy values on the MNIST dataset. Each curve corresponds to a different δ value.

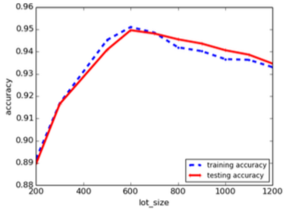
MNIST Results



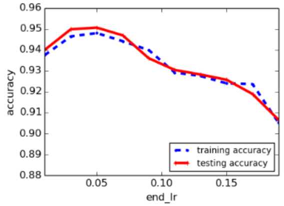
(1) variable projection dimensions



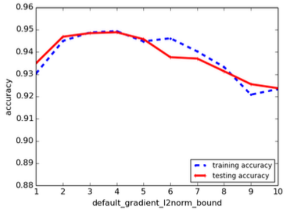
(2) variable hidden units



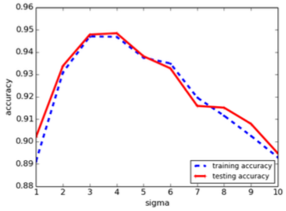
(3) variable lot size



(4) variable learning rate



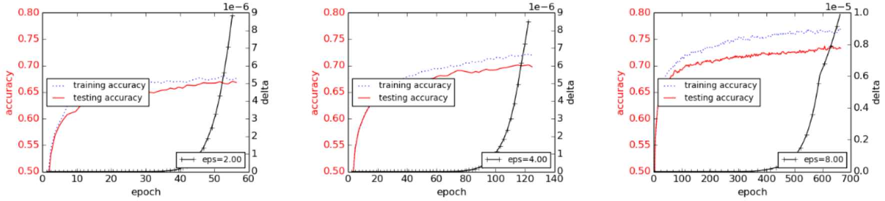
(5) variable gradient clipping norm



(6) variable noise level

Figure 5: MNIST accuracy when one parameter varies, and the others are fixed at reference values.

CIFAR-10 Results



(1) $\varepsilon = 2$ (2) $\varepsilon = 4$ (3) $\varepsilon = 8$

Figure 6: Results on accuracy for different noise levels on CIFAR-10. With δ set to 10^{-5}, we achieve accuracy 67%, 70%, and 73%, with ε being 2, 4, and 8, respectively. The first graph uses a lot size of 2,000, (2) and (3) use a lot size of 4,000. In all cases, σ is set to 6, and clipping is set to 3.

Related Works

- Convex optimization methods have been extensively used for differentially private learning. For example, Wu et al.[2] achieve 83% accuracy on MNIST via private convex ERM.
- Shokri and Shmatikov[3] designed and evaluated a system for distributed training of a deep neural network, where privacy budget is per model parameter, which is several thousand times more.
- Differentially private deep learning by Phan et al.[4] focuses on learning autoencoders, where privacy is based on perturbing the objective function.

[2] X. Wu, A. Kumar, K. Chaudhuri, S. Jha, and J. F. Naughton. Differentially private stochastic gradient descent for in-RDBMS analytics. CoRR, abs/1606.04722, 2016.

[3] R. Shokri and V. Shmatikov. Privacy-preserving deep learning. In CCS. ACM, 2015.

[4] N. Phan, Y. Wang, X. Wu, and D. Dou. Differential privacy preservation for deep auto-encoders: an application of human behavior prediction. In AAAI, 2016.

Conclusion

- Model achieves 97% training accuracy for MNIST and 73% accuracy for CIFAR-10, both with $(8, 10^{-5})$-differential privacy
- Differentially private SGD algorithm proposed and implemented in TensorFlow
- Moments accountant proposed for tighter bounds on privacy loss