Shanghai Jiao Tong University

Assignment4 Concept Questions

Student name: *520030910342 Jiyu Liu*

Course: *Data Mining* – Professor: *Liyao Xiang*

Date: *May 28, 2023*

1 Concepts Questions

1.1 Question 1

Question: Calculate b from the Laplace distribution $Lap(x|b)$ that satisfies ε-differential privacy with an ℓ_1-sensitivity of 1.

Laplace mechanism is defined as :

$$M(x, f(\cdot), \epsilon) = f(x) + Y$$

,where Y is a random variable drawn from $Lap(\Delta f/\epsilon)$.

Cause the Laplace distribution $Lap(x|b)$ that satisfies ϵ-differential privacy with an l1-sensitivity of 1,

$$\max_{\|x-y\|_1 \leq 1} \|f(x) - f(y)\|_1 = 1$$

Let p_x denote the PDF of $M(x)$ and p_y denote the PDF of M(y). At some arbitrary point z:

$$\begin{aligned}
\frac{p_x(z)}{p_y(z)} &= \frac{\exp(-\frac{|f(x)-z|}{b})}{\exp(-\frac{|f(y)-z|}{b})} \\
&= \exp((|f(x)-z| - |f(y)-z|)/b) \leq \exp(\|f(x)-f(y)\|_1/b) \\
&\leq \exp(\frac{1}{b}) = \exp(\epsilon)
\end{aligned}$$

So that we have $\epsilon = \frac{1}{b}$, i.e. $b = \frac{1}{\epsilon}$.

1.2 Question 2

Question: Describe the algorithm for Differentially Private Stochastic Gradient Descent.

Part one: Input

The algorithm takes examples $x_1, \cdots, x_N$ as input. The loss function is defined as $L(\theta) = \frac{1}{N}\sum_i L(\theta, x_i)$. The parameters are defined as below: learning rate η_i, noise scale σ, gradient norm bound C.

Part two: Iteration

- First, we randomly initialize the model parameters.
- Second, for each $t \in [T]$, we take a random sample L_t with sampling probability L/N.

  Next, we compute the gradient of the loss function with respect to the model parameters, i.e $g_t(x_i) \leftarrow \nabla_{\theta_t} L(\theta_t, x_i)$ and clip the gradient in order to make all gradients satisfy $\|g\|_2 \leq C$.

  Then, we add random noise $N(0, \sigma^2 C^2 I)$ clipped gradients to ensure privacy.

  Finally, we use this clipped gradient with gradient added to do gradient descent and update parameters.

Part three: Output

The algorithm outputs network parameters after T updating iterations and the overall privacy cost (ϵ, δ) using a privacy accounting method.

1.3 Question 3

Question: Design an algorithm to enhance differentially private SGD based on the following requirements. Let $g(x_i) \in R^p$ be the gradient of the example x_i, and $G \in R^{n \times p} = [g(x_1)g(x_2)...g(x_n)]$ be the gradient matrix. Create an algorithm to compress the gradient matrix such that $\hat{G} = GB$, where $\hat{G} \in R^{n \times k}, k < p$, and $B \in R^{P \times k}$ is a direction matrix, related to the direction of G, and it needs to be guaranteed to be orthogonal. Utilize $\hat{G}$ to perform per-example clipping and add Gaussian noise in DPSGD. Finally, project the noise gradient back to R^p using B^T and update the model's parameters. Provide the algorithm for the entire process.

Part one: Input

The algorithm takes examples $x_1, \cdots, x_N$ as input. The loss function is defined as $L(\theta) = \frac{1}{N}\sum_i L(\theta, x_i)$. The parameters are defined as below: learning rate η_i, noise scale σ, gradient norm bound C.

Part two: Iteration

- First, we randomly initialize the model parameters.

- Second, for each $t \in [T]$, we take a random sample L_t with sampling probability L/N.

  Next, we compute the gradient of the loss function with respect to the model parameters, i.e $g_t(x_i) \leftarrow \nabla_{\theta_t} L(\theta_t, x_i)$ and clip the gradient in order to make all gradients satisfy $\|g\|_2 \leq C$.

  Let $G_t = [g_t(x_1)g_t(x_2)...g_t(x_n)]$. We use Singular Value Decomposition (SVD) method to decompose the gradient matrix: $G_t = U\Sigma V^T$, where Σ is a $p \times p$ rectangular diagonal matrix, U, V are $n \times p$ real orthogonal matrices.

  Select the top-k columns of V and transpose it to obtain a direction matrix: $B \in R^{p \times k}$, then the compressed gradient matrix $\hat{G} = GB$.

  Add Gaussian noise to the clipped gradient: Let $n_i \in R^k$ be a vector drawn from a Gaussian distribution with zero mean and a variance matrix of $\sigma^2 C^2 I$. Add n_i to the i-th row of $\hat{G}$ to obtain the noisy compressed gradient: $\hat{G}_i = \hat{G}_i + n_i$. Uncompress the gradient matrix: $\widehat{G} = \hat{G}B^T$. Then we can use it to do gradient descent and update parameters.

Part three: Output

The algorithm outputs network parameters after T updating iterations and the overall privacy cost (ϵ, σ) using a privacy accounting method.