CSDS440 Final Report

Jiasen Zhang
jxz867@case.edu

12/10/2021

1 Overview

I implemented 2 models, ANN and K-means, and 2 extensions and there are totally four files. In the 2 models I implement some privacy preserving methods in which I need to modify the algorithms. While in the two extensions I test the effect of some more general privacy preserving methods that can be applied to many algorithms. They are:

	Filename	Algorithm and method
Model 1	ann_1.py	Fully connected network with DP-SGD
Extension 1	ann_2.py	Fully connected network with Laplacian mechanism
Model 2	kmeans_1.py	K-means with DPLloyd
Extension 2	kmeans_2.py	K-means with input perturbation

2 Model 1: ANN with DP-SGD

As we have learned in class, a fully connected neural network is an ANN consisting of a series of fully connected layers. The dimension of input and output layers depends on the dimension of dataset and the number of classes.

I build a simple fully connected network with two hidden layers and focus on differential privacy. The datasets are iris data and volcanoes data. I train the model with 8000 to 10000 epochs and it's really enough for iris and volcanoes datasets. The training is fast with GPU and is not too slow even with CPU. Because it's supervised learning so training and testing sets are needed. I use cross validation dividing the dataset into 5 folds.

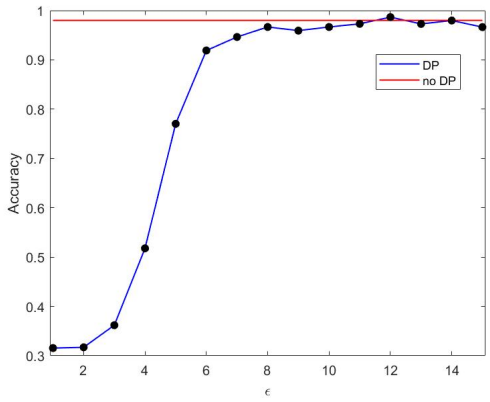
SGD (stochastic gradient descent) is learned in our class and it performs update separately for each example. The DP-SGD (differential privacy SGD) algorithm provide privacy preserving to neural network by adding noise to the gradients when performing SGD. In each step of SGD, the algorithm consists of the following steps [1]:

1. Take a subsample of the network parameters or weights.
2. Compute the gradients of loss w.r.t. each sample as original SGD.
3. Clip each gradient with a threshold C and their 2-norms: $g_i \leftarrow \max(1, ||g_i||_2/C)$
4. Aggregate the gradients, add Gaussian noise and get the average gradient:
$g \leftarrow mean(\sum g_i + N(0, \sigma^2 C^2 I)$
5. Update the weights with the gradient added noise.

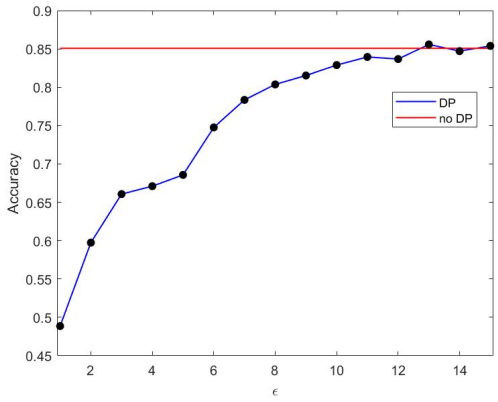
The mean and variance of Gaussian noise are 0 and $\sigma^2 C^2$, According to the paper [2], by choosing suitable privacy budget ϵ and δ which is related to σ, we can guarantee (ϵ, δ)-differentially private where means that ϵ-differentially is broken with probability δ [2]. Here I fix the δ as 1e-5 to guarantee ϵ-differentially and focusing on testing the effect of ϵ. In the code I modified the SGD optimizer of pytorch to a DP-SGD optimizer. To simplify, the group size of my subsample is one. So for each gradient I can just clip it and add noise.

Figure 1 (a) and (b) show the accuracy of two datasets with different values of ϵ. When ϵ is small, the accuracy is really low. The accuracy increases as ϵ increasing and converges to the accuracy without differential privacy. This tendency coincide with the results of the paper [2] and the definition of ϵ-differentially.

The results of two datasets are somewhat different. As ϵ increasing, the accuracy of iris converges to that of volcanoes with faster rate. It's reasonable because volcanoes dataset has much more features than iris and in my code the hidden units of volcanoes are more than iris. As a result, the amount of noise added to volcanoes in each DP-SGD step is much more than iris.



(a) DP-SGD Iris



(b) DP-SGD volcanoes

Figure 1: Fully connected network: DP-SGD.

3 Extension 1: ANN with Laplacian Mechanism

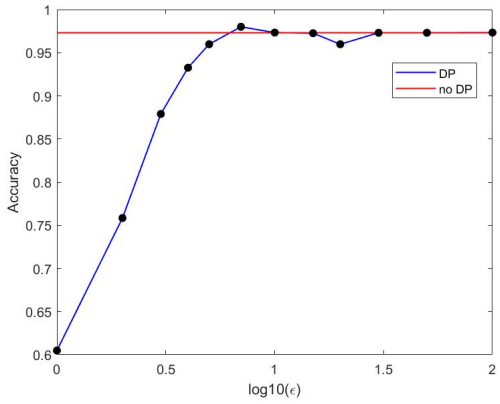
Laplacian mechanism and Gaussian mechanism are two classical and popular ϵ-differentially private mechanisms [3]. They can be used in various machine learning algorithms. In this part I use Laplacian mechanism in my fully connected network and observe the results.

Laplacian mechanism guarantee ϵ-differentially by adding Laplace noise to the output. The coefficient of the Laplacian distribution is defined by ϵ and sensitivity of the algorithm. The Laplacian mechanism consists of the following steps [4]:

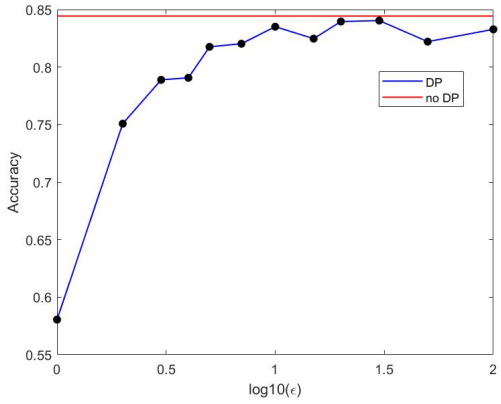
1. Get the output of the learning mechanism.
2. Get the sensitivity S with the output.
3. Add Laplace noise to the output: $Laplacian(0, S/\epsilon)$.

Because the output of my fully connected network comes from the SoftMax layer and their values are all between zero and one. In this case the sensitivity is simply no more than one. So I set sensitivity as one. The results in figure 3 and 4 are pretty good.

In figure 2 (a) and (b), I use semi log plot because the scale of ϵ variates largely. The accuracy increases as ϵ increasing and converges to the accuracy without differential privacy. The results show that this Laplacian mechanism satisfies differential privacy and works on my fully connected network and the datasets.



(a) Laplacian mechanism: Iris



(b) Laplacian mechanism: volcanoes

Figure 2: Fully connected network: Laplacian mechanism.

4 Model 2: K-means with DPLloyd

The k-means algorithm finds an optimal clustering so that the sum of distances between data points and the centroids of their clusters is the smallest. It's a clustering algorithm so no training and testing datasets.

The k-means algorithm written by me, using classical Lloyd's algorithm, has been compared with that of sklearn. It first randomly select initial centroids and then update them iteratively. In my initialization part, I randomly choose centroids for 20 times and use the best centroids as initial value. My results of iris data is as good as that of sklearn.

The Differentially Private K-Means I use is called DPLloyd [5]. The DPLloyd adds Laplace noise to the centroids in each update. It first adds noise to the count of one cluster, then add noise to the sum of points in the cluster and finally clamp the sum divided by the count with the data range.

The accuracy with different ϵ are shown in figure 3. As we expect, the overall accuracy of k-means is not as good as ANN. But the trend is similar, as the privacy budget ϵ increasing, the accuracy will finally converge to the accuracy without differential privacy.

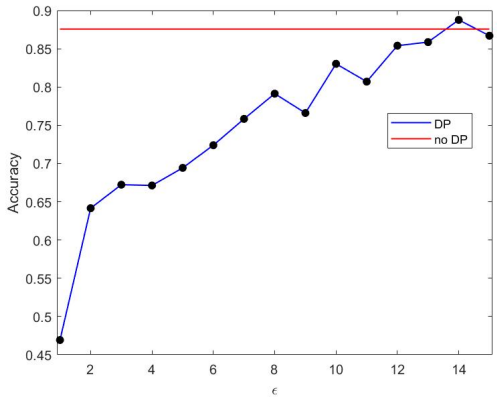


Figure 3: K-means: DPLloyd.

5 Extension 2: K-means with input perturbation

A method of private input perturbation for recommender system is introduced [6] and is proved to guarantee ϵ-differentially privacy. It's a very practical and effective method and obviously can be used in many other learning algorithms. In this part I use this input perturbation in my k-means algorithm with some modification and get some good results.

The algorithm supposes the input is a set of rating data, then the sensitivity is the difference of the maximum and the minimum rates. After getting sensitivity, Laplace noise is added to each of the data. Here I represent the rating with the average value of each sample and get very good result.

As shown in figure 4, As the privacy budget ϵ increasing, the accuracy will finally converge to the accuracy without differential privacy.

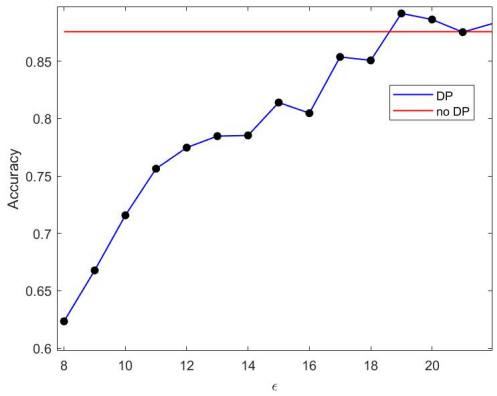


Figure 4: K-means: input perturbation.

6 Discussion

6.1 Comparison of model 1 and extension 1

Compared with DP-SGD, the accuracy of Laplacian mechanism increases slower as ϵ increasing (in figure 2 ϵ changes from 1 to 100). That means for DP-SGD we can use smaller ϵ (larger noise) to get close to the original accuracy. So DP-SGD is more stable than Laplacian mechanism for ANN when adding noise and therefore it has better effect of privacy preserving than Laplacian mechanism. However, Laplacian mechanism still has some advantages such as universality and simplicity.

6.2 Comparison of model 2 and extension 2

Obviously, the method of input perturbation needs larger value of ϵ to converge than DPLloyd, which means DPLloyd has better effect of privacy preserving than input perturbation. That's reasonable because input perturbation is not designed for k-means and I just try it in the extension, while DPLloyd is specific to k-means.

P.S. two of my references are unique: [1] and [5].

References

[1] Bu, Zhiqi, et al. ”Deep learning with Gaussian differential privacy.” Harvard data science review 2020.23 (2020).

[2] Abadi, Martin, et al. ”Deep learning with differential privacy.” Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. 2016.

[3] Ji, Zhanglong, Zachary C. Lipton, and Charles Elkan. ”Differential privacy and machine learning: a survey and review.” arXiv preprint arXiv:1412.7584 (2014).

[4] Dwork, Cynthia, et al. ”Calibrating noise to sensitivity in private data analysis.” Theory of cryptography conference. Springer, Berlin, Heidelberg, 2006.

[5] Su, Dong, et al. ”Differentially private k-means clustering and a hybrid approach to private optimization.” ACM Transactions on Privacy and Security (TOPS) 20.4 (2017): 1-33.

[6] Ren, Jiahui, et al. ”Recommender systems based on autoencoder and differential privacy.” 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC). Vol. 1. IEEE, 2019.