Deep learning with differential privacy on decentralized data

1st Yuanjia Wang
State Key Laboratory of Advanced Optical Communication Systems and Networks
Shanghai Jiao-tong University
Shanghai, China
yogawang@sjtu.edu.cn

***Abstract*—Deep learning networks have the ability to identify complex and subtle patterns from very large datasets. Ususally, the data are obtained by crowdsourcing technique and collecting users' data containing sensitive information. So the privacy must be taken into consideration in the deep learning models. Here, I consider the scenario where the data is distributed among multiple participants. Combining the deep learning techniques with the privacy-preserving mechanism, I study the stochastic gradient decent algorithm with differential privacy on the DistBelief model and analyze its performance.**

***Index Terms*—deep learning, privacy preserving, differential privacy**

I. Introduction

Machine learning equipped the computer with the ability to find patterns in data. New technique called deep learning is achieving remarkable results in many fields, like image classification, natural language processing and speech recognition [1]–[3].

Deeper networks and larger datasets can both dramatically improve the accuracy of the model. It is easy to understand since machine learning only take part of the whole data as the training set. So increasing the scale of deep learning, expanding training dataset and adding model parameters all can improve the performance. However, for a single participant, the computational capability is limited even he use a group of GPUs. The size of data or parameters is not enough for problems which needs lots of examples.

Let's consider a simple two-party scenario, which can help find the solution to the distributed deep network among many parties: two hospitals both want to learn the relationship between the lung cancer and the CT images. They have to share the data and the deep network because each hospital can't get good results by its own. However, the data must be kept secret from the other hospital, since the privacy of patients need to be protected. And the hospitals are both very mean. They don't want to reveal any extra information except the training model.

So how to solve it? I will introduce the concept of differential privacy first. Differential privacy will provide privacy by introduing randomness into the raw data. The randomness will not decrease the model accuracy. Instead, it can greatly increase the robustness of the model [4]. It's promising to combine differential privacy and deep network together.

In this paper, I describe a two-party distributed deep network scheme with differential privacy based on Google's deep neural network architecture DistBelief [4]. I show the model parallelism can be achieved as well as the robustness and privacy preservation. I will firstly introduce the basic concept of deep learning and differential privacy in Section II and describe the model in Section IV. Finally I discuss the future improvement of the model.

II. Preliminaries

In this sections, I briefly review deep learning and introduce basic concepts of differential privacy.

A. Deep learning

Deep learning is based on deep neural networks, which have remarkable effect on many machine learning applications. It take the mapping function from inputs to outputs as a combination of many layers of basic building blocks. Taking keras for example, the network layers includes dense layer, activation layer, dropout layer and more. Often, the most commonly used blocks are affine transformation, which transforms the coordinates of data and simple nonlinear function, which introduces the nonlinear factor into the model. By choosing appropriate kinds and orders of those blocks, I can form a network and train the mapping function to obtain the parameters which can fit any given finite size of examples.

To evaluate the parameters during training, I defince a loss function representing the penalty for the difference between the estimated result and the true output. The aim of training is to reduce the value of loss function to an acceptably small one and get close enough to the global minimum.

Complex networks usually have non-convex loss function, which is difficult to minimize. Previous studies proposed a minimization method called stochastic gradient descent (SGD) algorithm. In SGD algorithm, one run the loops where he forms a batch B of random examples and computes an estimation value $g_B = \frac{1}{|B| \sum_{x \in B} \Delta_\theta \mathcal{L}(\theta,x)}$ where $\mathcal{L}(\theta,x)$ is the loss function and
$theta$ is updated in the gradient direction $-g_B$ which points to the local minimum.

There are several frameworks of deep learning, such as Tensorflow [5], SparkNet [6], and Keras [7]. Here I choose the classical model DistBelief [4] as the basis because [4] designed a method called Downpour SGD which satisfy our assumption: the datas and models are distributed with a shared parameter server, and the SGD algorithm can be run on the parameter server.

B. *Differential Privacy*

Differential privacy is a strong standard for preserving privacy of large databases in algorithm. It is defined in terms of the application-specific concept of adjacent databases.

Before I talk about differential privacy, some definitons about probability need to be introduced first:

Definition 1 (Probability Simplex). Given a discrete set B, the probability simplex over B, denoted $\triangle(B)$ is defined to be:

$$\triangle(B) = \{x \in \mathbb{R}^{|B|} : x_i \leq 0 \ for \ all \ i \ and \ \sum_{i=1}^{|B|} x_i = 1\}$$

III. Model

Definition 2 (Randomized Algorithm). A randomized algorithm $\mathcal{M}$ with domain A and discrete range B is associated with a mapping $M : A \rightarrow \Delta(B)$. On input $a \in A$, the algorithm $\mathcal{M}$ outputs $\mathcal{M}(a) = b$ with probability $(M(a))_b$ for each $b \in B$. The probability space is over the coin flips of the algorithm $\mathcal{M}$.

Definition 3 (Distance Between Database) Suppose I have databases $x \in \mathbb{N}^{|\mathcal{X}|}$ which are collections of records from the whole space $\mathcal{X}$. The l_1 norm of x is denoted $||x||_1$ and has the following form:

$$||x||_1 = \sum_{i=1}^{|\mathcal{X}|} |x_i|$$

The l_1 distance between two databases x and y is $||x-y||_1$.

The definition of differential privacy is like following:

Definition 4 (Differential Privacy) A randomized algorithm $\mathcal{M}$ with domain $\mathbb{N}^{|\mathcal{X}|}$ is (ϵ,δ)-differentially private if for all $\mathcal{S} \subseteq Range(\mathcal{M})$ and for all $x,y \in \mathbb{N}^{|\mathcal{X}|}$ such that the l_1 distance between x and y satisfies $||x-y||_1 \leq 1$:

$$Pr[\mathcal{M}(x) \in \mathcal{S}] \leq e^{\epsilon} Pr[\mathcal{M}(y) \in \mathcal{S}] + \delta$$

When $\delta = 0$, $\mathcal{M}$ is ϵ-differentially private.

IV. Model

This section describes our model under the two-party scenario. I combine the differentially private stochastic gradient descent (SGD) algorithm with the Downpour SGD in DistBelief together.

A. *Assumption*

Fig. 1 shows our model structure. The model consists of two parties (called Alice and Bob in the next section of paper) and a parameter server. Each pary own a model replica and a database. The deep network model are known to both parties and they have a same copy of it, while the database are different. I can take the combination of the datasets as a whole virtual database. Then the data is horizontally partitioned. Each object in the virtual database is either completely owned by Alice or completely owned by Bob. And the model are trained on Alice and Bob's joint dataset.

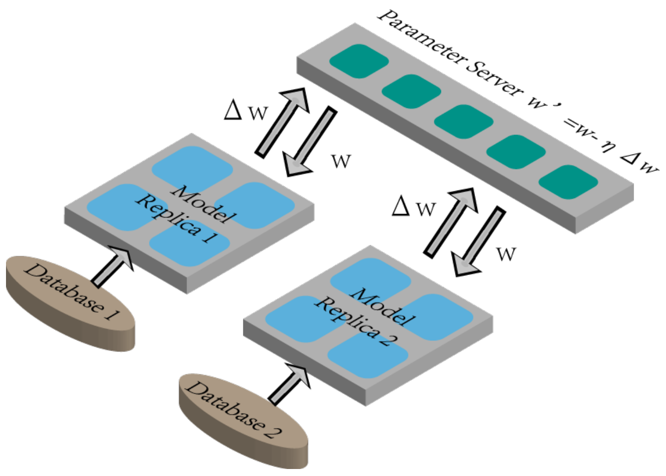

Fig. 1. The model of two-party distributed deep network schemes. Each party own a model replica and a database. They share a parameter server.

B. *Differentially Private SGD Algorithm on distributed deep network*

Stochastic gradient descent (SGD) is the most commonly used optimization procedure for minimizing the complex loss function. [4] proposed Downpour SGD, a variant of asynchronous stochastic gradient descent that uses many replicas of a single model. Downpour SGD divide a training data into a number of subsets and run a copy of the model on each of these subsets. Similar to the algorithm, I have a divided virtual dataset and a shared model. So I can train the network asynchronously with two remote party and a parameter server.

The algorithm is shown in Table I:

The differential privacy has been proved in [8]. Here I give a simple explaination. Taking appreciate value of σ in algorithm,the algorithm can be (ϵ,δ)-differentially private. Since the two party train the model independently, each θ they get is a little outdated. Meanwhile, if the parameter server runs well, even one party fails during the training, the other parties can still get a trraining result. These both increases the model's robustness.

V. Improvement

A main problem in the previous model is that the parameter server needs to be trustful and the communication between the parties and the model must be secure. Improvement can be made if I remove the parameter server from the model but let the two party store the parameters instead. Then it becomes a total multi-party computation problem. A new model without the parameter server is shown in Fig. 2.

I need to introduce the concept of homomorphic encryption and random shares into the model. Homomorphiic encryption is a form of encryption that allows computation on ciphertexts and by decrypt the real result from the "encrypted" result.

TABLE I
DIFFERENTIALLY PRIVATE SGD ALGORITHM ON DISTRIBUTED DEEP NETWORK

```
PARTICIPANTS Alice, Bob, and parameter server.
INPUT Examples $\{x_{a_1},x_{a_2},...,x_{a_n}\}$ owned by Alice and $\{x_{b_1},x_{b_2},...,x_{b_m}\}$
    owned by Bob, 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 $C$.
STEPS
The parameter server Initialize $\theta_0$ randomly
Each party A/B does:
for $t \in [T]$ do
    Take a random sample $L_t$ with sampling probability $L/N$
    Each party asks the current version of $\theta$ from the parameter server.
    it's denoted as $\theta_t$.
    Compute gradient
    For each $i \in L_t$, compute $g_t(x_{A/B_i}) \leftarrow \Delta_{\theta_i}\mathcal{L}(\theta_t,x_{A/B_i})$
    Clip gradient
    $\bar{g}_t(x_{A/B_i}) \leftarrow \frac{g_t(x_{A/B_i})}{max(1,\frac{||g_t(x_{A/B_i})||2}{C}}$
    Add noise
    $\tilde{g}_t \leftarrow \frac{1}{L}\sum_i(\bar{g}_t(x_{A/B_i}) + \mathcal{N}(0,\sigma^2C^2I)$
    Descent
    submit $\Delta = \eta_t\tilde{g}_t$ to the server
The server updates the $\theta$ each time it receives a submission:
    $\theta' = \theta - \Delta$.
OUTPUT $\theta_T$ stored in the server and compute the overall privacy
    cost $(\epsilon,\delta)$ using a privacy accounting method.
```

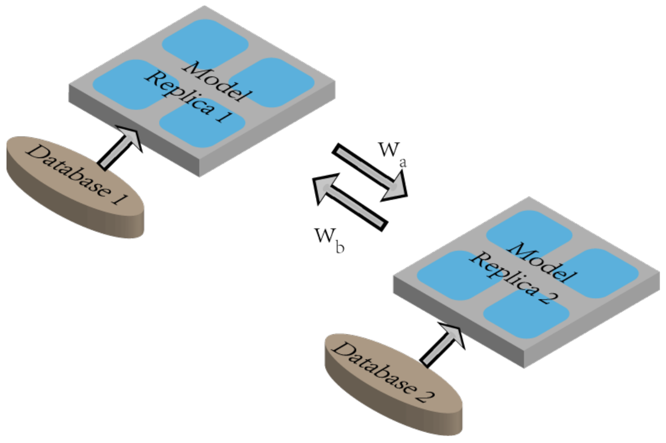

Fig. 2. The model of two-party distributed deep network schemes. Each party own a model replica and shares the parameters withous any third party.

Random shares means the values are shared as uniformly distributed random values between the two parties. The sum of the values owned by the parties is the actual value. [9] More formally, I have the following definition:

Definition 5 (Random Shares) Alice and Bob have random shares of a value x drawn from a field F of size N. The abbreviation is that Alice and Bob have random shares of x. That means Alice knows a value $a \in F$ and Bob knows a value $b \in F$ such that

$$(a + b)\ mod\ N = x$$

where a and b are uniformly random in field F.

In this model, Alice and Bob have random shares of θ. Firstly they use homomorphic encrypton to own a random share of θ_0 securely. Suppose Alice owns θ_a and Bob owns θ_b. When one party like Alice wants to use θ, she will ask Bob to get a outdated $\tilde{\theta}_b$. Then she computes $\tilde{\theta} = \theta_a + \tilde{\theta}_b$ and generates a new θ_a randomly. She then sends $\tilde{\theta} - \theta_a$ to Bob and tells Bob to replace θ_b with it. When Bob wants to use θ, he does the same steps.

The new algorithm is shown in Table II:

TABLE II
DIFFERENTIALLY PRIVATE SGD ALGORITHM ON DISTRIBUTED DEEP NETWORK WITHOUT SERVER

```
PARTICIPANTS Alice, Bob.
INPUT Examples $\{x_{a_1},x_{a_2},...,x_{a_n}\}$ owned by Alice and $\{x_{b_1},x_{b_2},...,x_{b_m}\}$
    owned by Bob, 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 $C$.
STEPS
The two parties Initialize $\theta_0$ randomly and establish a random shares of $\theta_0$
    by homomorphic encryption.
Each party A/B does:
for $t \in [T]$ do
    Take a random sample $L_t$ with sampling probability $L/N$
    Each party asks the other to calculate $\theta$.
    the parameter is denoted as $\theta_t$.
    Compute gradient
    For each $i \in L_t$, compute $g_t(x_{A/B_i}) \leftarrow \Delta_{\theta_i}\mathcal{L}(\theta_t,x_{A/B_i})$
    Clip gradient
    $\bar{g}_t(x_{A/B_i}) \leftarrow \frac{g_t(x_{A/B_i})}{max(1,\frac{||g_t(x_{A/B_i})||2}{C}}$
    Add noise
    $\tilde{g}_t \leftarrow \frac{1}{L}\sum_i(\bar{g}_t(x_{A/B_i}) + \mathcal{N}(0,\sigma^2C^2I)$
    Descent
    Calculate $\theta' = \theta_t - \eta_t\tilde{g}_t$
    Randomly choose a value $\theta'_{A/B}$. Send $\theta' - \theta'_{A/B}$ to the other party. The
    parties updates the $\theta_{A/B}$ each time when they receive a new value from
    the other:
OUTPUT $\theta_T = \theta_{A_T} + \theta_{B_T}$ and compute the overall privacy
    cost $(\epsilon,\delta)$ using a privacy accounting method.
```

The parameter exchanging process ensures that the communication complexity is almost the same as the previous model with a parameter server. But the model tights up the constraints: it require both parties work well during the training, since if one fails, the true value of θ is lost. And the robustness is only kept by the randomness introduced to hold differential privacy.

VI. DISCUSSION

If there are three or more parties, the requirements can be relaxed: each party can store $k(k \geq 2)$ values about θ, if one party fails, the remaining parties can still obtain an approximate θ. However, the communication scheme will be more complex. The orders of sending random values and the randomness of the subgroups which can obtain an approximate θ must be taken into consideration.

Here I only discuss the horizontally partitioned data. Actually, there can be cases that the data is vertically partitioned that each attribute is completely stored by one party. Arbitrary partitioned data is also possible [10]. In such cases, not only the parameters needs to be randomly shared, but also the datasets because one party can't get well trained result when the atrributes is not complete.

VII. Acknowledgment

I thank Prof. Xinbing Wang and Prof. Luoyi Fu for the mobile internet class and helpful advices about this project. I especially appreciate Prof. Fu's help for the reference materials she kindly provided. I also thank Shanghai Jiao Tong University to provide the access to Google Scholar.

References

[1] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich *et al.*, "Going deeper with convolutions." Cvpr, 2015.

[2] O. Vinyals, Ł. Kaiser, T. Koo, S. Petrov, I. Sutskever, and G. Hinton, "Grammar as a foreign language," in *Advances in Neural Information Processing Systems*, 2015, pp. 2773–2781.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[4] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, A. Senior, P. Tucker, K. Yang, Q. V. Le *et al.*, "Large scale distributed deep networks," in *Advances in neural information processing systems*, 2012, pp. 1223–1231.

[5] "Tensorflow," https://www.tensorflow.org/.

[6] "Sparknet," https://sparknet.net/.

[7] "Keras," https://github.com/keras-team.

[8] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016, pp. 308–318.

[9] Y. Lindell and B. Pinkas, "Privacy preserving data mining," in *Annual International Cryptology Conference*. Springer, 2000, pp. 36–54.

[10] G. Jagannathan and R. N. Wright, "Privacy-preserving distributed k-means clustering over arbitrarily partitioned data," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 2005, pp. 593–599.