The Workshops of the Thirty-Second
AAAI Conference on Artificial Intelligence

Distributed Deep Learning under Differential Privacy with the Teacher-Student Paradigm

Jun Zhao
Carnegie Mellon University, and
Nanyang Technological University
junzhao@alumni.cmu.edu

Abstract

The goal of this *work in progress* is to address distributed deep learning under differential privacy using the teacher-student paradigm. In the setting, there are a number of distributed entities and one aggregator. Each distributed entity leverages deep learning to train a teacher network on sensitive and labeled training data. The knowledge of the teacher networks is transferred to the student network at the aggregator in a privacy-preserving manner that protects the sensitive data. This transfer results from training non-sensitive and unlabeled data. We also apply secure multi-party computation to securely combining the outputs of local machine learning, in order to update a global model.

Introduction

Recently, deep learning has become hugely popular for its ability to solve end-to-end learning systems (Goodfellow, Bengio, and Courville 2016). Its success results from a combination of algorithmic breakthroughs, increasingly powerful computers, and access to large amounts of data (Hitaj, Ateniese, and Perez-Cruz 2017).

Massive collection of training data required for deep learning presents obvious privacy issues. Among the efforts to quantify privacy, the recent notion of differential privacy by Dwork *et al.* (2006) has been successfully used in a range of data analysis tasks, since it provides a rigorous foundation for defining and preserving privacy. Differential privacy has received considerable attention in the literature (Abadi et al. 2016; Ghosh, Roughgarden, and Sundararajan 2012; Kairouz, Oh, and Viswanath 2014; Papadimitriou, Narayan, and Haeberlen 2017). Apple (2016) incorporated differential privacy into its mobile operating system iOS 10. Google implemented a differentially private tool called RAPPOR in the Chrome browser to collect information about clients (Erlingsson, Pihur, and Korolova 2014). A randomized algorithm Y satisfies ϵ-differential privacy if for any adjacent databases x and x' differing in one record, and for any event E, it holds that $\mathbb{P}\left[Y(x) \in E\right] \leq e^{\epsilon}\mathbb{P}\left[Y(x') \in E\right]$, where $\mathbb{P}\left[\cdot\right]$ denotes the probability throughout this paper. Intuitively, under differential privacy, an adversary given access to the output does not have much confidence to determine whether it was sampled from the probability distribution generated by the algorithm when the database is x or when the database is x'.

Papernot *et al.* (2017) utilize the student–teacher framework in deep learning. The framework has received considerable attention recently. In their setting, the teacher networks are learnt non-privately from disjoint subsets of the private training data. The parameters of the teacher networks are not accessible by the adversary. A student model is trained on the public data to mimic the output of the ensemble of teachers; i.e., the student aims to predict the same class that is predicted by the most number of teachers. The student model is published and accessible by the adversary.

To ensure privacy, whenever the student has to access the teachers, some noise is added to the output of the teacher ensemble. This is used to achieve differential privacy. In (Papernot et al. 2017), the total privacy cost depends on the number of queries made to the teacher ensemble by the student, and the noise level added to the output of the teacher ensemble. The total privacy cost is computed using the moment accountant technique proposed by Abadi *et al.* (2016).

In this paper, we consider distributed deep learning under differential privacy using the teacher-student framework. Each distributed entity utilizes deep learning to train a teacher network on private and labeled data. Then the knowledge of the teachers is transferred to the student network at the aggregator in a differentially private manner. This transfer uses non-sensitive and unlabeled data for training.

Contributions. The idea of this work in progress is largely motivated by (Papernot et al. 2017). We discuss this paper's contributions to distributed privacy-preserving deep learning via a comparison with (Papernot et al. 2017).

- Compared with (Papernot et al. 2017), we consider the more general distributed setting, where the private training data is distributed across different entities. In contrast, Papernot *et al.* (2017) assume that the private training data is at the same place, which limits the application scenarios. For instance, the medical data may be distributed at different hospitals.
- Papernot *et al.* (2017) use the Laplace noise to achieve ϵ-differential privacy (DP) for some ϵ to protect the private training data. Since ϵ-DP can be too strong to satisfy in certain applications (Abadi et al. 2016), we consider

(ϵ, δ)-DP, a privacy definition more relaxed than ϵ-DP, and then use the Gaussian noise to achieve (ϵ, δ)-DP.

- We consider deep learning, while Papernot *et al.* (2017) consider general learning problems. Hence, we need to address specific challenges in deep learning.
- Although differential privacy restricts the information leaked from publishing the final result, it does not handle the collaboration part for computation over data distributed between different parties. To solve this, we adopt the advanced technique of secure multi-party computation, which can be used to securely combine the outputs of local machine learning, in order to update a global model.
- We consider local differential privacy, which allows personalized choice of privacy for each data distributed at different entities. In contrast, Papernot *et al.* (2017) use the weaker notion of (global) differential privacy.

We note that in both (Papernot et al. 2017) and this paper, a necessary assumption is that non-private unlabeled data is available to train the student network. This may not be much of a problem because public datasets exist for text, images, and medical data.

Future Work. We outline two future directions.

- We address (local) differential privacy in this paper. It will be of interest to address other recently proposed privacy definitions; e.g., Rényi differential privacy (Mironov 2017), membership privacy (Li et al. 2013), and semantic privacy (Kasiviswanathan and Smith 2014).
- We will consider dynamic training data; e.g., the private training data may change over time.

Roadmap. The rest of the paper is organized as follows. The next section presents some preliminaries. Afterwards, we present our framework of privacy-preserving distributed deep learning with the student–teacher architecture. Then we discuss related work and conclude the paper.

Preliminaries

Deep Learning

We review deep learning below based on the discussion by Shokri and Shmatikov (2015). Deep learning extracts complex features from high-dimensional data and utilize them to build a model that relates inputs to outputs (e.g., classes). A common architecture for deep learning is a multi-layer network, where lower-level features are input to nonlinear functions for computing more abstract features. Figure 1 illustrates a neural network with two hidden layers. Each node represents a neuron, which receives the output of the neurons in the previous layer plus a bias signal from a special neuron emitting one. A weighted average of the inputs is used to derive the total input. Then applying a nonlinear activation function to the total input value results in the output. For neurons in layer k, the output vector is $\boldsymbol{o}_k = f(W_k \boldsymbol{o}_{k-1})$ for an activation function f, where the weight matrix W_k determines the contribution of each input signal. When the neural network is used to classify input data into several classes, the activation function in the last layer is often a softmax function $f(z_j) = e^{z_j}/(\sum_k e^{z_k}),\ \forall j$. In the last layer, the output

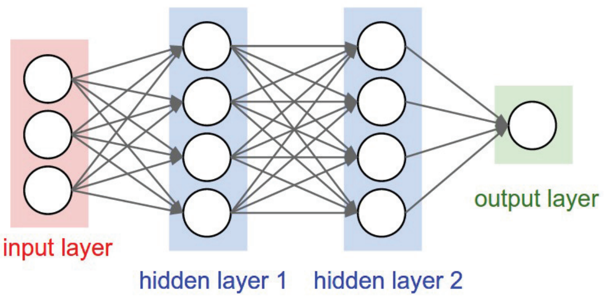

Figure 1: A neural network with two hidden layers.

of each neuron j is often the probability that the input belongs to class j. The goal of deep learning is to learn the weight parameters from the training data in order to maximize the learning accuracy.

Learning network parameters using gradient descent. Due to the nonlinear activation functions, learning a neural network's parameters is a nonlinear optimization problem. The algorithms to solve this problem are often variants of gradient descent. After beginning at a random set of parameters, gradient descent at each step computes the gradient of the nonlinear objective function being optimized and updates the parameters in order to decrease the gradient. This process continues iteratively to the point where a local optimum is found.

The gradient of each weight parameter is computed via feed-forward and back-propagation operations. First, feed-forward computes the output of the network and calculates the error defined as the difference between this output and the true value of the function. Second, back-propagation propagates the error back through the layers to compute the contribution of each neuron to the total error. For each parameter, its gradient is computed according to its contribution to the error and the neuron's activation value.

Stochastic gradient descent. The gradients of the parameters can be naively averaged over all available data. However, this approach called batch gradient descent is inefficient. To improve efficiency, stochastic gradient descent (SGD) computes the gradient over a small subset (called mini-batch) of the whole training dataset.

Let $\boldsymbol{w}$ be the vector of all parameters. Let E be the error function which denotes the difference between the true value and the computed output. E is often based on ℓ_2 norm or cross entropy. The back-propagation procedure calculates the partial derivative of E with respect to each parameter in $\boldsymbol{w}$ and updates the parameter in order to reduce its gradient. In SGD, a parameter w_j is updated according to $w_j := w_j - \alpha \frac{\partial E_i}{\partial w_j}$, where α denotes the learning rate and E_i is computed over the ith minibatch.

Differential Privacy

Differential privacy (DP) (Dwork et al. 2006) has received considerable interest recently. Intuitively, a randomized mechanism achieving (ϵ, δ)-DP means that except with a (typically small) probability δ, altering a record in a database cannot change the probability that the same output is seen by more than a multiplicative factor e^ϵ. Formally, for x and x' iterating through all pairs of neighboring databases (i.e., databases that differ by one record), and for $\mathcal{Y}$ iterating through all subsets of the output range of some mechanism Y, the mechanism Y achieves (ϵ, δ)-DP if $\mathbb{P}[Y(x) \in \mathcal{Y}] \leq e^\epsilon \mathbb{P}[Y(x') \in \mathcal{Y}] + \delta$, where $\mathbb{P}[\cdot]$ denotes the probability, and the probability space is over the coin flips of the mechanism Y. If $\delta = 0$, the notion of (ϵ, δ)-DP becomes ϵ-DP. In the literature, (ϵ, δ)-DP and ϵ-DP (Steinke and Ullman 2016; Zhao, Zhang, and Poor 2017; Zhu et al. 2015) are also referred to as approximate DP and pure DP, respectively.

Local Differential Privacy

In local differential privacy (Kairouz, Oh, and Viswanath 2014; Qin et al. 2016; Zhao and Zhang 2017; Kairouz, Oh, and Viswanath 2014), the private data is regarded as a single tuple, so any different pairs of data are neighboring. Local differential privacy hides the complete data, not just each individual record, from the adversary.

Secure Multi-Party Computation

Secure multi-party computation refers to a set of cryptographic technologies designed to enable computation over data distributed between different parties so that only the result of the computation is revealed to the participants, but no other information is shared (Yao 1982; Ben-David, Nisan, and Pinkas 2008; Pettai and Laud 2015). For example, two companies may be interested in computing how large is the intersection of their customer bases without revealing their customer lists to each other.

Assume that there are k entities such that each entity i holds a private value x_i. The aim of secure multi-party computation is to evaluate a function $f(x_1, \ldots, x_k)$, so that each party learns the result, but no information about each other's inputs x_i beyond what can be inferred from the result.

Distributed Deep Learning under Differential Privacy

We consider distributed deep learning under differential privacy. There are N distributed entities and one aggregator. Each distributed entity leverages deep learning to train a teacher network on private and labeled training data. The knowledge of the teacher networks is transferred to the student network at the aggregator in a privacy-preserving manner that protects the sensitive data. This transfer results from training non-sensitive and unlabeled data. For convenience, let x be one of such samples. In this work in progress, we present only the basic idea, which is largely motivated by (Papernot et al. 2017). Further details will be provided in the full version.

Let $m_n(x)$ be the true predicted class of the nth teacher network for sample x. To protect privacy, the nth teacher network adds a truncated geometric random variable $G(\epsilon, m_n(x), M)$ to $m_n(x)$ while enforcing the result after perturbation (denoted by $\widetilde{m_n}(x)$) is still a valid class number (i.e., it still belongs to $\{1, 2, \ldots, M\}$. We have

$$\widetilde{m_n}(x) := m_n(x) + G(\epsilon, m_n(x)), \tag{1}$$

where $G(\epsilon, m_n(x), M)$ follows a truncated geometric distribution specified as follows (Ghosh, Roughgarden, and Sundararajan 2012):

$$\mathbb{P}[G(\epsilon, m_n(x), M) = g] = \begin{cases} 0, & \text{if } g < -m_n(x) \text{ or } g > M - m_n(x), \\ \frac{e^{-\epsilon \cdot m_n(x)/M}}{1+e^{-\epsilon/M}}, & \text{if } g = -m_n(x), \\ \frac{e^{-\epsilon \cdot [M - m_n(x)]/M}}{1+e^{-\epsilon/M}}, & \text{if } g = M - m_n(x), \\ \frac{e^{\epsilon/M}-1}{e^{\epsilon/M}+1} e^{-\epsilon \cdot |g|/M}, & \text{if } -m_n(x) < g < M - m_n(x), \end{cases} \tag{2}$$

Since the sensitivity of $m_n(x)$ with respect to the private training data of the nth teacher network is M, it is straightforward to show that $\widetilde{m_n}(x)$ achieves ϵ-local differential privacy with respect to the private training data of the nth teacher network.

Among the N teacher networks, some may participate in helping the student train when the sample is x, while some may not. Let $\mathcal{N}(x)$ be the set of participating teachers. In the case where secure multi-party computation protocol is not enforced, for $n \in \mathcal{N}(x)$, the nth teacher network just sends $\widetilde{m_n}(x)$ directly to the student. Among $\{\widetilde{m_n}(x) : n \in \mathcal{N}(x)\}$ that the student collects, let $c_j(x)$ be the number of times that j appears, where $j \in \{1, 2, \ldots, M\}$; i.e., $c_j(x)$ denotes the number of teachers reporting class j as the predicted class after adding perturbation to protect privacy. Formally,

$$c_j(x) = |\{n : n \in \mathcal{N}(x) \text{ and } \widetilde{m_n}(x) = j\}|. \tag{3}$$

The student computes the joint noisy class predicted by the N teacher networks as follows:

$$C(x) := \arg\max_j c_j(x). \tag{4}$$

The training goal of the student is to mimic $C(x)$ as the output class for sample x.

When a secure multi-party computation protocol is not enforced, the goal is to allow the student compute $C(x)$ from the data $\{\widetilde{m_n}(x) : n \in \mathcal{N}(x)\}$ distributed at the teachers. Designing such secure multi-party computation protocol is one of our ongoing work.

Let T_n be the number of times that the nth teacher network helps train the student; i.e., the nth teacher network computes $\widetilde{m_n}(x)$ for T_n number of x. Then the total privacy cost for the nth teacher network's training data is $T_n\epsilon$.

Related Work

Abadi *et al.* (2016) demonstrate that deep neural networks with differential privacy can be trained with a modest total privacy loss. Shokri and Shmatikov (2015) as well as Zhang *et al.* (2017) and Chase *et al.* (2017) present systems for privacy-preserving multiparty deep learning, wherein autonomous data owners jointly train accurate deep neural network models without sharing their private data. Among these work, Shokri and Shmatikov (2015) use differential privacy and does not utilize secure multi-party computation. Although Zhang *et al.* (2017) leverage both techniques, there is no quantifiable analysis on the privacy cost. Hitaj *et al.* (2017) demonstrate that certain deep learning systems are susceptible to a powerful reconstruction attack.

Pathak *et al.* (2010) propose a privacy-preserving protocol for composing a differentially private aggregate classifier using local classifiers from different parties. Papadimitriou *et al.* (2017) present a system that efficiently performs computations on graphs containing confidential data.

Lindell and Pinkas (2009) survey the basic paradigms and notions of secure multiparty computation and discuss their relevance to the field of privacy-preserving data mining.

Conclusion

We have studied distributed deep learning under differential privacy. The system consists of a number of distributed entities and one aggregator. Each distributed entity applies deep learning to the training of a teacher network on sensitive and labeled data. The knowledge of the teachers is transferred to the student at the aggregator in a privacy-preserving way that protects privacy of the sensitive data. This transfer uses non-sensitive and unlabeled data for training.

References

Abadi, M.; Chu, A.; Goodfellow, I.; McMahan, H. B.; Mironov, I.; Talwar, K.; and Zhang, L. 2016. Deep learning with differential privacy. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 308–318.

Apple Incorporated. 2016. iPhone user guide for iOS 10.

Ben-David, A.; Nisan, N.; and Pinkas, B. 2008. FairplayMP: A system for secure multi-party computation. In *ACM Conference on Computer and Communications Security (CCS)*, 257–266.

Chase, M.; Gilad-Bachrach, R.; Laine, K.; Lauter, K.; and Rindal, P. 2017. Private collaborative neural network learning. *Cryptology ePrint Archive, Report 2017/762*.

Dwork, C.; McSherry, F.; Nissim, K.; and Smith, A. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference (TCC)*, 265–284.

Erlingsson, Ú.; Pihur, V.; and Korolova, A. 2014. RAPPOR: Randomized aggregatable privacy-preserving ordinal response. In *ACM Conference on Computer and Communications Security (CCS)*, 1054–1067.

Ghosh, A.; Roughgarden, T.; and Sundararajan, M. 2012. Universally utility-maximizing privacy mechanisms. *SIAM Journal on Computing* 41(6):1673–1693.

Goodfellow, I.; Bengio, Y.; and Courville, A. 2016. *Deep learning*. MIT press.

Hitaj, B.; Ateniese, G.; and Perez-Cruz, F. 2017. Deep models under the GAN: Information leakage from collaborative deep learning. In *ACM Conference on Computer and Communications Security (CCS)*.

Kairouz, P.; Oh, S.; and Viswanath, P. 2014. Extremal mechanisms for local differential privacy. In *Conference on Neural Information Processing Systems (NIPS)*, 2879–2887.

Kasiviswanathan, S., and Smith, A. 2014. On the 'semantics' of differential privacy: A Bayesian formulation. *Journal of Privacy and Confidentiality* 6(1):1–16.

Li, N.; Qardaji, W.; Su, D.; Wu, Y.; and Yang, W. 2013. Membership privacy: A unifying framework for privacy definitions. In *ACM Conference on Computer and Communications Security (CCS)*, 889–900.

Lindell, Y., and Pinkas, B. 2009. Secure multiparty computation for privacy-preserving data mining. *Journal of Privacy and Confidentiality* 1(1):59–98.

Mironov, I. 2017. Rényi differential privacy. In *IEEE Computer Security Foundations Symposium (CSF)*.

Papadimitriou, A.; Narayan, A.; and Haeberlen, A. 2017. DStress: Efficient differentially private computations on distributed data. In *European Conference on Computer Systems (EuroSys)*, 560–574.

Papernot, N.; Abadi, M.; Erlingsson, Ú.; Goodfellow, I.; and Talwar, K. 2017. Semi-supervised knowledge transfer for deep learning from private training data. In *International Conference on Learning Representations (ICLR)*.

Pathak, M.; Rane, S.; and Raj, B. 2010. Multiparty differential privacy via aggregation of locally trained classifiers. In *Conference on Neural Information Processing Systems (NIPS)*, 1876–1884.

Pettai, M., and Laud, P. 2015. Automatic proofs of privacy of secure multi-party computation protocols against active adversaries. In *IEEE Computer Security Foundations Symposium (CSF)*, 75–89.

Qin, Z.; Yang, Y.; Yu, T.; Khalil, I.; Xiao, X.; and Ren, K. 2016. Heavy hitter estimation over set-valued data with local differential privacy. In *ACM Conference on Computer and Communications Security (CCS)*, 192–203.

Shokri, R., and Shmatikov, V. 2015. Privacy-preserving deep learning. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 1310–1321.

Steinke, T., and Ullman, J. 2016. Between pure and approximate differential privacy. *Journal of Privacy and Confidentiality* 7(2):3–22.

Yao, A. C. 1982. Protocols for secure computations. In *Symposium on Foundations of Computer Science*, 160–164.

Zhang, X.; Ji, S.; Wang, H.; and Wang, T. 2017. Private, yet practical, multiparty deep learning. In *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 1442–1452.

Zhao, J., and Zhang, J. 2017. Preserving privacy enables "coexistence equilibrium" of competitive diffusion in social networks. *IEEE Transactions on Signal and Information Processing over Networks* 3(2):282–297.

Zhao, J.; Zhang, J.; and Poor, H. V. 2017. Dependent differential privacy for correlated data. In *IEEE GLOBECOM 5th Annual Workshop on Trusted Communications with Physical Layer Security*.

Zhu, T.; Xiong, P.; Li, G.; and Zhou, W. 2015. Correlated differential privacy: Hiding information in non-IID data set. *IEEE Transactions on Information Forensics and Security* 10(2):229–242.