LEASGD: an Efficient and Privacy-Preserving Decentralized Algorithm for Distributed Learning

Hsin-Pai Cheng[*]
ECE Department
Duke University
Durham, NC 27708
hc218@duke.edu

Patrick Yu[*]
Monta Vista High School
Cupertino, CA 95014
pyu592@student.fuhsd.org

Haojing Hu[*]
Beihang University of
Aeronautics and Astronautics
Beijing, China
haojinghu@buaa.edu.cn

Feng Yan
Computer Science
and Engineering
University of Nevada, Reno
Reno, NV 89557
fyan@unr.edu

Shiyu Li
Tsinhua University
Beijing, China
shiyu.li@duke.edu

Hai Li
ECE Department
Duke University
Durham, NC 27708
hai.li@duke.edu

Yiran Chen
ECE Department
Duke University
Durham, NC 27708
yiran.chen@duke.edu

Abstract

Distributed learning systems have enabled training large-scale models over large amount of data in significantly shorter time. In this paper, we focus on decentralized distributed deep learning systems and aim to achieve differential privacy with good convergence rate and low communication cost. To achieve this goal, we propose a new learning algorithm LEASGD (Leader-Follower Elastic Averaging Stochastic Gradient Descent), which is driven by a novel Leader-Follower topology and a differential privacy model. We provide a theoretical analysis of the convergence rate and the trade-off between the performance and privacy in the private setting. The experimental results show that LEASGD outperforms state-of-the-art decentralized learning algorithm DPSGD by achieving steadily lower loss within the same iterations and by reducing the communication cost by 30%. In addition, LEASGD spends less differential privacy budget and has higher final accuracy result than DPSGD under private setting.

1 Introduction

With data explosion and ever-deeper neural network structures, distributed learning systems play an increasingly important role in training large-scale models with big training data sources [1][2][3]. Most distributed learning systems have centralized parameter server(s) to maintain a single global copy of the model and coordinate information among workers/clients. However, such system topology is vulnerable to privacy leakage because once the central server(s) is eavesdropped, information of the entire system can be exposed [4]. Decentralized distributed learning systems are potentially more

[*] Equal Contribution

Accepted by NIPS 2018 Workshop on Privacy Preserving Machine Learning (PPML), Montréal, Canada.

robust to the privacy as critical information such as training data, model weights, and the states of all workers can no longer be observed or controlled through a single point of the system [5].

However, decentralized systems usually perform worse in convergence rate and are known to have higher communication cost. In addition, most of the decentralized systems do not provide guarantees on differential privacy [6, 7]. Some recent works are proposed to solve the above problems. For example, D-PSGD [6] focuses on improving communication efficiency and convergence rate of decentralized learning systems, but it is not differentially private. Bellet *et al.* considers both decentralized design and differential privacy in their recent work [5]. However, it is based on a simple linear classification task, not a good representation of the modern neural networks, which have much more complex and deeper structures. Inspired by the above work, we aim at developing a generalized decentralized learning approach that has better applicability and can achieve differential privacy with good convergence rate and low communication cost.

To this end, we propose LEASGD (*Leader-Follower Elastic Averaging Stochastic Gradient Descent*) that provides differential privacy with improved convergence rate and communication efficiency. To improve both the communication and training efficiency while also facilitate differential privacy, we first design a novel communication protocol that is driven by a dynamic leader-follower approach. The parameters are only transferred between the leader-follower pair, which significantly reduces the overall communication cost. LEASGD adopts the insight of the *Elastic Averaging Stochastic Gradient Descent* (EASGD) [8] by exerting the elastic force between the leader and follower at each update. Inspired by [9], we use *momentum account* to quantize the privacy budget which provides a tighter bound of privacy budget ϵ than the classical *Strong Composition Theorem* [10]. In addition, the convergence rate of LEASGD is mathematically proved.

We evaluate LEASGD against the state-of-the-art approach D-PSGD [6] on three main aspects: the convergence rate, the communication cost, and the privacy level. The theoretical analysis shows LEASGD converge faster than D-PSGD. Our real testbed experiments show LEASGD achieves higher accuracy than D-PSGD in the non-private setting with the same communication cost and under the private setting with less privacy budget.

2 Leader-Follower Elastic Averaging Stochastic Gradient Descent Algorithm and Privacy-preserving Scheme

2.1 Problem Setting

We assume there are m workers each with a set of local data S_i and $i \in \{1, 2......m\}$, which can only be accessed locally by worker i. Along the training process, each worker i computes a parameter vector w_t^i at each iteration t to represent the learning outcomes and then computes the corresponding loss function $f_t^i(w^i) = l(w_t^i, x_t^i, y_t^i)$ with the input x_t^i and given labels y_t^i. After learning from the data, each worker has two ways to contribute to the global learning progress: 1) Update its model parameters by local gradient descent; 2) Communicate with other workers to update each other's model parameters. We define a communication interval τ to represent how many iterations between each update in our learning algorithm. When the training process is done, each worker has its own variation of the same model (i.e., performing the same task but with different trained model parameters w^i). Given each worker has its own local version of the model, it is necessity to assemble all local models unbiasedly by averaging the loss function. We formulate it as an optimization problem as follows:

$$w^* = \{w^1, ..., w^m\} \tag{1}$$

$$\underset{w^i \in \Omega}{argmin}\, \overline{F}(w, T) = \frac{1}{m}\sum_{i=1}^{m} f_T^i(w^i), s.t. \Omega \subseteq \mathbb{R}^n \, and \, T \in \mathbb{R} \tag{2}$$

where T presents the predefined number of iterations in τ.

2.2 Decentralized Leader-Follower Topology

To support the decentralized design, we categorize all workers into two worker pools: leader pool with workers of lower loss function values and follower pool with workers of higher loss function

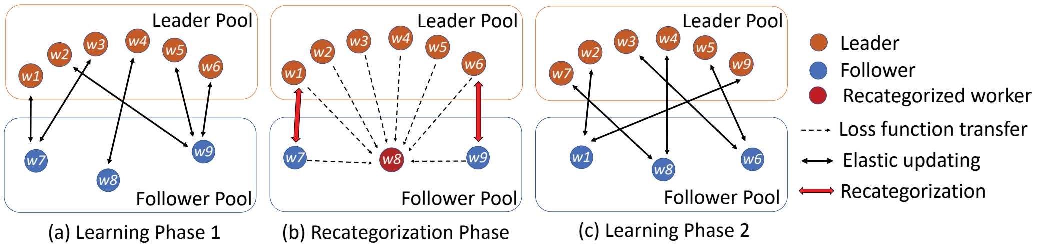


Figure 1: The dynamic Leader-Follower topology. (a) shows the structure of leader pool and follower pool. (b) shows a recategorization phase where one of the workers is elected as recategorized worker and gathers the latest loss function values from all other workers. (c) shows the new structure of leader pool and follower pool after recategorization (note the randomization used for avoiding over-fitting).

values, see Figure 1(a). The leader pools are always larger than the follower to guarantee the sufficient pull power. The core idea is to let leaders to pull follower so that better performing workers (leaders) can guide the followers in the right direction to improve the learning. Specifically, we use an elastic updating rule to regulate the learning updates in each leader-follower pair as follows:

$$w^i_{t+1} = w^i_t - \eta g^i_t + \eta\rho(w^f_t - w^i_t) \quad and \quad w^f_{t+1} = w^f_t - \eta g^f_t + \eta\rho(w^i_t - w^f_t) \tag{3}$$

We use i to denote a leader; f is a follower; k is the categorization interval; ρ is elastic factor; g is gradient; and η is learning rate. Given learning is a dynamic process, the two worker pools are dynamically updated based on the learning progress. The pools are recategorized each $k\tau$ time interval. This protocol enables the convergence rate of our algorithm to have a limited upper bound. To avoid over-fitting to one worker's model during the training process, we add the L2-normalization on the training loss function. We also randomly pair the leaders and followers after each learning update to avoid one follower's model having excessive influence on others. This randomization mechanism also benefits the privacy-preserving as randomized communication can confuse the attacker and make it more difficult to trace the information source. An asynchronous version can be derived by setting the number of wake up iterations for different workers according to a Poisson Stochastic Process with different arrival rate based on local clock time of each worker, see Supplementary for more details.

2.3 Privacy-preserving Scheme

The general idea to preserve differential privacy is to add noise on the output of the algorithm and the noise scale is based on the sensitivity of the output function as defined in [11]. Note that for different input data, equation 3 only differs in the gradient g^i_t part. In other words, the sensitivity of the updating rule of LEASGD is the same as the gradient g^i_t. Thus we use the similar scheme as the DP-SGD algorithm [9]. To limit the sensitivity of gradient, we clip the gradient into a constant C as $\overline{g}^i_t = g^i_t/max(1, \frac{\|g^i_t\|_2}{C})$. Then, we add Gaussian noise on the clipped gradient

$$\tilde{g}^i_t = \overline{g}^i_t + \mathcal{N}(0, \sigma_2^2 C^2) \tag{4}$$

By using $\tilde{g}^i_t$ to replace g^i_t in equation 3, we obtain the differential-privacy preserving scheme of LEASGD as:

$$\tilde{w}^i_{t+1} = \tilde{w}^i_t - \eta\tilde{g}^i_t + \eta\rho(\tilde{w}^f_t - \tilde{w}^i_t) \quad and \quad \tilde{w}^f_{t+1} = \tilde{w}^f_t - \eta\tilde{g}^f_t + \eta\rho(\tilde{w}^i_t - \tilde{w}^f_t) \tag{5}$$

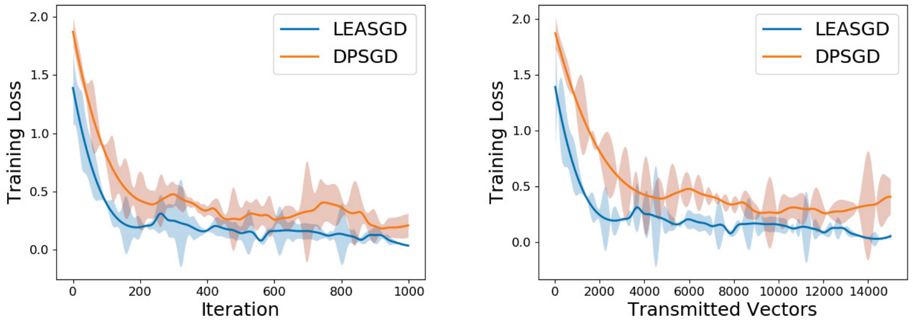
When we choose the variance of Gaussian noise $\sigma_2 = \frac{\sqrt{2ln(1.25/\delta)}}{\epsilon}$, we ensure that each communication step of LEAGSD is (ϵ, δ)-DP. Using the property of DP-mechanism in [10], the composition of a series of DP-mechanisms remains DP, which guarantees that for each worker i, its training algorithm $\mathcal{M}_i$ at each iteration is DP.

3 Analysis

Convergence Rate Analysis. In this section, we provide a convergence rate analysis for synchronous LEASGD in a strongly-convex case and also compare it with the D-PSGD [6] theoretically.

We define that

$$d_t = \frac{E\sum_{i=1}^{p} \| w^i_t - w^* \|^2 + E \| w^f_t - w^* \|^2}{p+1} \tag{6}$$



(a) (b)

Figure 2: (a). Training Loss vs. iteration on MNIST (b). Training Loss vs. number of transmitted vectors on MNIST

and the result of convergence rate of d_t is as follow.

Proposition 1 *(Convergence rate of d_t) If* $0 \leq \eta \leq \frac{2(1-\beta)}{\mu+L}$,$0 \leq \alpha = \eta\rho < 1$,$0 \leq \beta = p\alpha < 1$, *then we obtain the convergence of* d_t

$$d_t \leq h^t d_0 + (c_0 - \frac{\eta^2\sigma_1^2}{\gamma})(1-\gamma)^t(1-(\frac{p}{p+1})^t) + \eta^2\sigma_1^2\frac{1-h^t}{\gamma},$$
$$where\ 0 < h = \frac{p(1-\gamma)}{p+1} < 1, k = \frac{1-\gamma}{p+1}, \gamma = 2\eta\frac{\mu L}{\mu + L},\ c_0 = \underset{i=1,\ldots,p,f}{max} \parallel w_0^i - w^* \parallel^2 \tag{7}$$

This proposition implies that the average gap between all workers and optimum in a subsystem includes three parts, which could also be applied to the whole system. If we simply ignore the influence of the inherent noise on the gradient and extend the $t \to \infty$, we can easily obtain an purely exponential decline of the gap, that is $E[d_{t+1}] \leq hE[d_t]$. Note that the shrink factor h is negatively correlated to the p, which implies that, when our system is operating in a strongly convex setting, the larger worker scale can correspondlingly result in a faster convergence rate of the system. The convergence rate of D-PSGD [6] is $O(1/[(p+1)t])$ with our denotation in the strongly-convex setting. Compared with our $O(h^t)$ rate, the convergence rate of D-PSGD is relatively slower when we extend the $t \to \infty$. Detailed proof can be found in Supplementary material.

Privacy trade-off analysis. Following the convergence rate analysis above, we obtain the modified convergence rate of d_t by adding the extra noise. We assume the DP noise is independent of the inherent noise. Thus, the variance of the composed noise is the sum of the two independent noise variances and it satisfies $\sigma^2 < \sigma_1^2 + C^2\sigma_2^2$. Finally, by replacing the σ_1^2 with $\sigma_1^2 + C^2\sigma_2^2$ in Proposition 1, we obtain the convergence rate in the private setting. The extra trade-off can be formulated as $\frac{\eta^2 C^2 \sigma_2^2}{\gamma}$ when $t \to \infty$. Note that this trade-off remains the same when p grows. It implies our algorithm has a stable scalability when applied in the private setting.

4 Experimental Evaluation

We perform experimental evaluation using a 3-layer Multi-layer Perceptron (MLP) and 3-layer CNN with MNIST and CIFAR-10, respectively, running on a cluster of 15 servers each equipped with 4 NVIDIA Tesla P100 GPUs and the servers are connected with 100Gb/s Intel Omni-Path fabric.

Non-private setting comparison. To quantitatively evaluate the communication cost, we track the average training loss of all workers by comparing between the proposed LEASGD and D-PSGD in terms of the number of iterations and transmitted vectors, see Figure 2. Theoretically, in each iteration, LEASGD has less transmitted vectors than D-PSGD. Figure 2(a) shows that LEASGD converges faster than D-PSGD at the beginning of the training process and also achieves a lower loss function at the end. Figure 2(b) shows LEASGD outperforms D-PSGD in terms of the communication efficiency, i.e., with the same transmitted vectors, LEASGD achieves better training loss.

Differential private comparison. In the private setting, we use the *momentum account* to compute the totally spent ϵ and the adding noise scales are the same for two algorithms. As shown in Table 1, LEASGD achieves better accuracy with less ϵ than D-PSGD. More importantly, the final accuracy of our algorithm does not vary greatly when the worker scale m increases. We believe this result benefits from two attributes of LEASGD: 1) the DP noise helps improve the accuracy by encouraging space exploration and helping workers trapped in local optimum to get out [12]; 2) the great scalability that prevents DP noise from accumulating when the worker scale expands.

	m=5				m=15			
	ours		D-PSGD		ours		D-PSGD	
	Accu.	Total ϵ	Accu.	Total ϵ	Accu.	Total ϵ	Accu.	Total ϵ
MNIST	**0.97**	**4.183**	0.97	4.505	**0.97**	**4.651**	0.95	4.843
CIFAR-10	**0.74**	**4.651**	0.71	4.925	**0.72**	**4.116**	0.68	4.56

Table 1: Private setting result of ϵ and accuracy

5 Acknowledgement

This work is supported in part by the following grants: National Science Foundation CCF-1756013, IIS-1838024, 1717657 and Air Force Research Laboratory FA8750-18-2-0057.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Davis, et al. Tensorflow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, OSDI'16, pages 265–283, Berkeley, CA, USA, 2016. USENIX Association.

[2] Jeffrey Dean, Greg S. Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Quoc V. Le, Mark Z. Mao, Marc'Aurelio Ranzato, Andrew Senior, and Paul Tucker. Large scale distributed deep networks. In *International Conference on Neural Information Processing Systems*, pages 1223–1231, 2013.

[3] Eric P. Xing, Qirong Ho, Wei Dai, Kyu Kim Jin, Jinliang Wei, Seunghak Lee, Xun Zheng, Pengtao Xie, Abhimanu Kumar, and Yaoliang Yu. Petuum: A new platform for distributed machine learning on big data. *IEEE Transactions on Big Data*, 1(2):1335–1344, 2015.

[4] Naman Agarwal, Ananda Theertha Suresh, Felix Yu, Sanjiv Kumar, and H Brendan Mcmahan. cpsgd: Communication-efficient and differentially-private distributed sgd. *arXiv preprint arXiv:1805.10559*, 2018.

[5] Aurelien Bellet, Rachid Guerraoui, Mahsa Taziki, and Marc Tommasi. Personalized and private peer-to-peer machine learning. *AISTATS*, 2018.

[6] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pages 5330–5340, 2017.

[7] Feng Yan, Shreyas Sundaram, SVN Vishwanathan, and Yuan Qi. Distributed autonomous online learning: Regrets and intrinsic privacy-preserving properties. *IEEE Transactions on Knowledge and Data Engineering*, 25(11):2483–2493, 2013.

[8] Sixin Zhang, Anna E Choromanska, and Yann LeCun. Deep learning with elastic averaging sgd. In *Advances in Neural Information Processing Systems*, pages 685–693, 2015.

[9] Martin Abadi, Andy Chu, Ian J Goodfellow, H Brendan Mcmahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. *computer and communications security*, pages 308–318, 2016.

[10] Cynthia Dwork, Guy N. Rothblum, and Salil Vadhan. Boosting and differential privacy. *Foundations of Computer Science Annual Symposium on*, 26(2):51–60, 2010.

[11] Cynthia Dwork. Differential privacy. *33rd International Colloquium on Automata, Languages and Programming, part II (ICALP 2006)*, 2006.

[12] Arvind Neelakantan, Luke Vilnis, Quoc V Le, Ilya Sutskever, Lukasz Kaiser, Karol Kurach, and James Martens. Adding gradient noise improves learning for very deep networks. *arXiv preprint arXiv:1511.06807*, 2015.