OVERLAP LOCAL-SGD: AN ALGORITHMIC APPROACH TO HIDE COMMUNICATION DELAYS IN DISTRIBUTED SGD

Jianyu Wang *Hao Liang* *Gauri Joshi*

Carnegie Mellon University, Pittsburgh, USA
{jianyuw1, hliang2, gaurij}@andrew.cmu.edu

ABSTRACT

Distributed stochastic gradient descent (SGD) is essential for scaling the machine learning algorithms to a large number of computing nodes. However, the infrastructures variability such as high communication delay or random node slowdown greatly impedes the performance of distributed SGD algorithm, especially in a wireless system or sensor networks. In this paper, we propose an algorithmic approach named Overlap-Local-SGD (and its momentum variant) to overlap the communication and computation so as to speedup the distributed training procedure. The approach can help to mitigate the straggler effects as well. We achieve this by adding an anchor model on each node. After multiple local updates, locally trained models will be pulled back towards the synchronized anchor model rather than communicating with others. Experimental results of training a deep neural network on CIFAR-10 dataset demonstrate the effectiveness of Overlap-Local-SGD. We also provide a convergence guarantee for the proposed algorithm under non-convex objective functions.

***Index Terms*—** Local SGD, communication efficient training, federated learning

1. INTRODUCTION

Distributed optimization with stochastic gradient descent (SGD) is the backbone of the state-of-the-art supervised learning algorithms, especially when training large neural network models on massive datasets [1, 2]. The widely adopted approach now is to let worker nodes compute stochastic gradients in parallel, and average these using a parameter server [3] or a blocking communication protocol ALLREDUCE [4]. Then, the model parameters are updated using the averaged gradient. This classical parallel implementation is referred as *fully synchronous SGD*. However, in a wireless system where the computing nodes typically have low bandwidth and poor connectivity, the high communication delay and unpredictable nodes slowdown may greatly hinder the benefits of parallel computation [5, 6, 7]. It is imperative to make distributed SGD to be fast as well as robust to the system variabilities.

A promising approach to reduce the communication overhead in distributed SGD is to reduce the synchronization frequency among worker nodes. Each node maintains a local copy of the model parameters and performs τ local updates (only using local data) before synchronizing with others. Thus, in average, the communication time per iteration is directly reduced by τ times. This method is called *Local SGD* or *periodic averaging SGD* in recent literature [8, 9, 10] and its variant *federated averaging* has been shown to work well even when worker nodes have non-IID data partitions [11]. However, the significant communication reduction of Local SGD comes with a cost. As observed in experiments [12], a larger number of local updates τ requires less communication but typically leads to a higher error at convergence. There is an interesting trade-off between the error-convergence and communication efficiency.

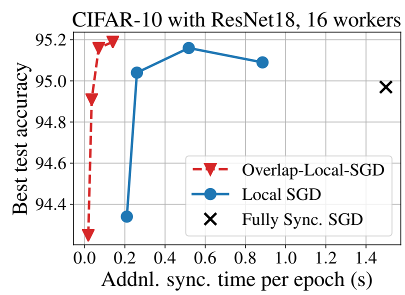


Fig. 1: Error-runtime trade-off. The proposed algorithm Overlap-Local-SGD significantly improves the Pareto efficiency of Local SGD. Each point in the plot corresponds to a specific value of τ. Note that the computation time per epoch is about 4.6 seconds.

In this paper, we propose a novel algorithm named *Overlap-Local-SGD* that further improves the communication efficiency of Local SGD and achieves a better balance in the error-runtime trade-off. The key idea in Overlap-Local-SGD is introducing an anchor model on each node. While workers are performing local updates, the anchor model use another thread/process to synchronize. Thus, the communication and computation are decoupled and happen in parallel. The locally trained models achieve consensus via averaging with the synchronized anchor model instead of communicating with others. The benefits of using Overlap-Local-SGD is shown in Figure 1. One can observe that the additional synchronization latency per epoch is nearly negligible compared to fully synchronous SGD. By setting a proper number of updates ($\tau = 1, 2$), Overlap-Local-SGD can even achieve a higher accuracy. Extensive experiments in Section 4 further validate the effectiveness of Overlap-Local-SGD under both IID and non-IID data settings. We provide a convergence analysis in Section 5 and show that the proposed algorithm can converge to a stationary point of non-convex objectives and matches the same rate of fully synchronous SGD.

2. PROPOSED ALGORITHM

Preliminaries. Consider a network of m worker nodes, each of which only has access to its local data distribution $\mathcal{D}_i$, for all $i \in \{1, \ldots, m\}$. Our goal is to use these m nodes to jointly minimize an objective function $F(\boldsymbol{x})$, defined as follows:

$$F(\boldsymbol{x}) := \frac{1}{m} \sum_{i=1}^{m} \mathbb{E}_{\boldsymbol{s} \sim \mathcal{D}_i} \left[\ell(\boldsymbol{x}; \boldsymbol{s}) \right] \tag{1}$$

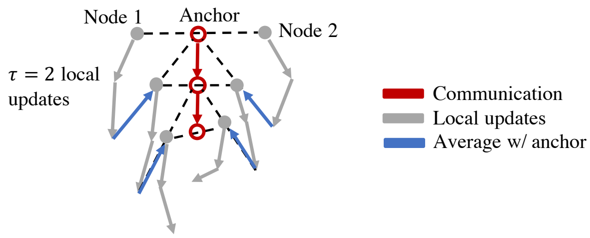


Fig. 2: Example on model trajectories in the model parameter space. It is worth noting that the update of anchor is performed in parallel to the local updates of worker nodes.

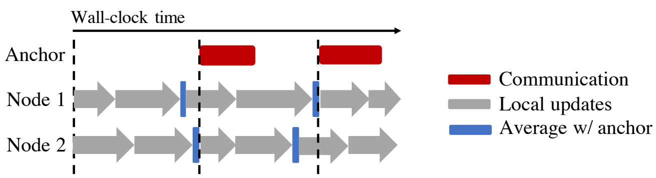


Fig. 3: The corresponding execution pipeline of the example in Figure 2. There is an extra communication thread on each worker node to perform communication and update anchor models.

where $\ell(\boldsymbol{x}; \boldsymbol{s})$ denotes the loss function for data sample $\boldsymbol{s}$, and $\boldsymbol{x}$ denotes the parameters in the learning model. In Local SGD algorithm, each node performs mini-batch SGD updates in parallel and periodically synchronize model parameters. For the model at i-th worker $\boldsymbol{x}^{(i)}$, we have

$$\boldsymbol{x}_{k+1}^{(i)} = \begin{cases} \frac{1}{m}\sum_{i=1}^{m}[\boldsymbol{x}_k^{(i)} - \gamma g_i(\boldsymbol{x}_k^{(i)}; \xi_k^{(i)})] & (k+1) \bmod \tau = 0 \\ \boldsymbol{x}_k^{(i)} - \gamma g_i(\boldsymbol{x}_k^{(i)}; \xi_k^{(i)}) & \text{otherwise} \end{cases} \tag{2}$$

where $g_i(\boldsymbol{x}_k^{(i)}; \xi_k^{(i)})$ represents the stochastic gradient evaluated on a random sampled mini-batch $\xi_k^{(i)} \sim \mathcal{D}_i$, and γ is the learning rate.

Overlap-Local-SGD. In Overlap-Local-SGD, each node maintains two set of model parameters: the locally trained model $\boldsymbol{x}^{(i)}$ and an additional anchor model $\boldsymbol{z}$, which can be considered as a stale version of the averaged local model. We omit the node index of $\boldsymbol{z}$ since it is always synchronized and the same across all nodes.

In Figures 2 and 3, we present a brief illustration of Overlap-Local-SGD. Specifically, after every τ local updates, the updated local model $\boldsymbol{x}^{(i)}$ will be pulled towards the anchor model, *i.e.*, $\boldsymbol{x}^{(i)} \leftarrow \boldsymbol{x}^{(i)} - \alpha(\boldsymbol{x}^{(i)} - \boldsymbol{z})$, where α is a tunable parameter. This step is critical to make local models become similar so that they can reach consensus and converge to a common point at the end of training. Intuitively, there should exist a best value of α that yields the fastest convergence. Besides, it is worth noting that the pullback step does not require any communication, since each node has one local copy of the anchor model. Right after pulling back, nodes will start next round of local updates immediately. Meanwhile, another thread on each node will synchronize the current local models in parallel and store the average value into the anchor model. As long as the parallel communication time is smaller than τ steps computation time, one can completely hide the communication latency. This can be achieved via setting a larger number of local updates τ.

Update Rule. We define matrices $\boldsymbol{X}_k, \boldsymbol{G}_k \in \mathbb{R}^{d\times(m+1)}$ to stack all local copies of model parameters and stochastic gradients:

$$\boldsymbol{X}_k = [\boldsymbol{x}_k^{(1)}, \dots, \boldsymbol{x}_k^{(m)}, \boldsymbol{z}_k], \tag{3}$$

$$\boldsymbol{G}_k = [g_1(\boldsymbol{x}_k^{(1)}; \xi_k^{(1)}), \dots, g_m(\boldsymbol{x}_k^{(m)}; \xi_k^{(m)}), \boldsymbol{0}]. \tag{4}$$

Then, the update rule of Overlap-Local-SGD can be written as

$$\boldsymbol{X}_{k+1} = [\boldsymbol{X}_k - \gamma \boldsymbol{G}_k]\boldsymbol{W}_k, \tag{5}$$

where $\boldsymbol{W}_k$ represents the mixing pattern between local models and the anchor model, which is defined as follows:

$$\boldsymbol{W}_k = \begin{cases} \begin{bmatrix} (1-\alpha)\boldsymbol{I} & (1-\alpha)\boldsymbol{1}_m/m \\ \alpha\boldsymbol{1}_m & \alpha \end{bmatrix} & (k+1) \bmod \tau = 0 \\ \boldsymbol{I} & \text{Otherwise.} \end{cases} \tag{6}$$

Note that $\boldsymbol{W}_k$ is a column-stochastic matrix, and hence previous analyses in distributed optimization literature [13, 10, 14], which require $\boldsymbol{W}_k$ to be doubly- or row-stochastic, cannot directly apply.

Momentum Variant. Momentum has been widely used to improve the optimization and generalization performance of SGD, especially when training deep neural networks [15]. Inspired by the distributed momentum scheme proposed in [16], Overlap-Local-SGD adopts a two-layer momentum structure. To be specific, the local updates on each node use common Nesterov momentum and the momentum buffer is updated only using the local gradients. Moreover, the anchor model also updates in a momentum style. When $(k+1) \mod \tau = 0$, we have

$$\boldsymbol{v}_{k+1} = \beta\boldsymbol{v}_k + \left(\frac{1}{m}\sum_{i=1}^{m}\boldsymbol{x}_{k+1}^{(i)} - \boldsymbol{z}_k\right), \tag{7}$$

$$\boldsymbol{z}_{k+1} = \boldsymbol{z}_k + \boldsymbol{v}_{k+1} \tag{8}$$

where $\boldsymbol{v}_k$ is the momentum buffer for anchor model and β denotes the momentum factor.

Mitigating the Effect of Stragglers. The overlap technique not only can hide the communication latency but also eliminates the straggler effect. This is because the communication operations are non-blocking, and hence workers do not need to wait for others in some cases. As shown in Figure 3, when the communication time satisfies certain conditions, worker nodes will run independently and there is no idle time in waiting for the slowest one. In order to analyze the delay model, we denote by $Y_{i,k} \sim F_Y$ the time taken by the i-th worker node to finish the k-th local step. For simplicity, $Y_{i,k}$ are assumed to be IID across nodes and iterations. In the ideal case, the c-th anchor update (*i.e.*, c-th communication step) happens at

$$T_c = \max_{i\in\{1,\dots,m\}} \left[\sum_{k=0}^{c\tau-1} Y_{i,k}\right]. \tag{9}$$

Similarly, one can get the expression of T_{c-1}. Note that when the communication time T_{comm} is smaller than $T_c - T_{c-1}$, then there is no waiting time. This yields a threshold on the number of local updates τ:

$$T_{\text{comm}} \leq \mathbb{E}\left[\max_{i\in\{1,\dots,m\}}\left[\sum_{k=0}^{c\tau-1} Y_{i,k}\right] - \max_{i\in\{1,\dots,m\}}\left[\sum_{k=0}^{(c-1)\tau-1} Y_{i,k}\right]\right]. \tag{10}$$

The exact expression of RHS in (10) depends on the distribution F_Y. In the simplest case where $Y_{i,k}$ is a constant, we have $\tau \geq T_{\text{comm}}/Y$.

3. RELATED WORKS

The idea of pulling back locally trained models towards an anchor model is inspired by elastic averaging SGD (EASGD) [17], which allows some slack between local models by adding a proximal term to the objective function. The convergence guarantee of EASGD under non-convex objectives has not been established until our recent work [10]. However, in EASGD, the anchor and local models are updated in a symmetric manner (*i.e.*, mixing matrix $\boldsymbol{W}_k$ in (5) should be symmetric and doubly-stochastic). EASGD naturally allows overlap of communication and computation, but the original paper [17] did not observe and utilize this advantage to reduce communication delays.

There also exist other techniques that can decouple communication and computation in Local SGD. In [18], the authors propose to apply the local updates to an averaged model which is τ-iterations before. Their proposed algorithm CoCoD-SGD can achieve the same runtime benefits as Overlap-Local-SGD. Nonetheless, later in Section 4, we will show that, Overlap-Local-SGD consistently reaches comparable or even higher test accuracy than CoCoD-SGD given the same τ.

Algorithm	$\tau = 1$	$\tau = 2$	$\tau = 8$	$\tau = 24$
CoCoD-SGD	94.98%	94.99%	94.05%	92.54%
EAMSGD	94.51%	93.89%	92.43%	89.93%
Ours	$\mathbf{95.19\%}$	$\mathbf{95.16\%}$	$\mathbf{94.25\%}$	$\mathbf{92.92\%}$

Table 1: Comparison of Local SGD variants in **IID** data partition setting. As a reference, fully synchronous SGD achieves a test accuracy of 94.97%. The corresponding training loss curves can be found in Appendix B.

Algorithm	$\tau = 1$	$\tau = 2$	$\tau = 8$	$\tau = 24$
CoCoD-SGD	91.50%	$\mathbf{91.67\%}$	Diverges	Diverges
EAMSGD	91.38%	91.12%	88.88%	85.59%
Ours	$\mathbf{91.56\%}$	91.61%	$\mathbf{91.45\%}$	$\mathbf{88.73\%}$

Table 2: Comparison of Local SGD variants in **Non-IID** data partition setting. As a reference, fully synchronous SGD achieves a test accuracy of 85.88%. The hyper-parameter choices are identical to the IID case.

4. EXPERIMENTAL RESULTS

Experimental setting. The experimental analysis is performed on CIFAR-10 image classification task [19]. We train a ResNet-18 [20] for 300 epochs following the exactly same training schedule as [5]. That is, the mini-batch size on each node is 128 and the base learning rate is 0.1, decayed by 10 after epoch 150 and 250. The first 5 epoch uses the learning rate warmup schedule as described in [4]. There are total 16 computing nodes connected via 40 Gbps Ethernet, each of which is equipped with one NVIDIA Titan X GPU. The training data is evenly partitioned across all nodes and *not shuffled* during training. The algorithms are implemented in PyTorch [21] and NCCL communication backend.

In Overlap-Local-SGD, the momentum factor of the anchor model is set to $\beta = 0.7$, following the convention in [16]. For different number of local updates τ, we tune the value of pullback parameter α. It turns out that in the considered training task, for $\tau \geq 2$, $\alpha = 0.6$ consistently yields the best test accuracy at convergence. In intuition, a larger value of α may enable a larger base learning rate. We believe that if one further tune the base learning rate and the momentum factor, the performance of Overlap-Local-SGD will be further improved. For example, in our setting, when $\tau = 1$, then $\alpha = 0.5$ and base learning rate 0.15 gives the highest accuracy.

Negligible Communication Cost. We first examine the effectiveness of the overlap technique. As shown in Figure 4(a), Overlap-Local-SGD significantly outperforms all other methods. Given a target final accuracy, Overlap-Local-SGD incurs nearly negligible additional latency compared to fully synchronous SGD (0.1s versus 1.5s per epoch). When $\tau = 2$, Overlap-Local-SGD reduces the communication-to-computation ratio from 34.6% to 1.5%, while maintaining roughly the same loss-versus-iterations convergence as fully synchronous SGD (see Figure 4(c)). The superiority of Overlap-Local-SGD will be further magnified when using a slow inter-connection (*e.g.*, 10 Gbps) or a larger neural network (*e.g.*, transformer [22]).

Compressing or quantizing the exchanged gradients among worker nodes is another communication-efficient training method, which is extensively studied in recent literature. Here, we choose PowerSGD [5], which is the state-of-the-art gradient compression algorithm, as another baseline to compare with. In Figure 4, the rank of PowerSGD ranges from $\{1, 2, 4, 8\}$ (lower means higher compression ratio). When the rank is 1 (the lowest), PowerSGD can compress the transferred gradient by $243\times$. However, even in this extreme case, the additional synchronization latency of PowerSGD is still much higher than Local SGD methods. The reason is that the nodes cost some time to establish the handshakes. Compression techniques cannot reduce this part of communication overhead, and also introduce non-negligible encoding and decoding latency.

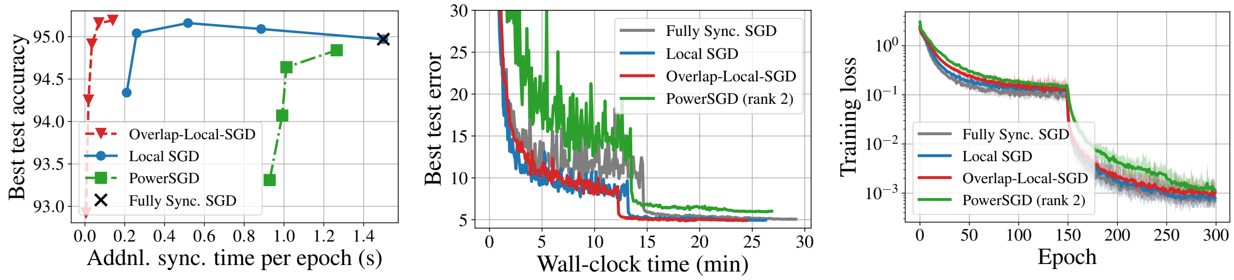
Higher Accuracy than Other Local SGD Variants. As discussed in Section 3, EASGD (and its momentum version EAMSGD [17]) also involve(s) a similar 'pullback' mechanism as Overlap-Local-SGD. And CoCoD-SGD proposed in [18] can decouple communication and computation as well. In Table 1, we empirically compare the performance of these Local SGD variants. The results show that given a fixed number of τ, Overlap-Local-SGD always achieves the best test accuracy among all methods, and EAMSGD has significant worse performance than others.

Non-IID Data Partitions Setting. We further validate the effectiveness of Overlap-Local-SGD in a non-IID data partitions setting. In particular, each node is assigned with 3125 training samples, 2000 of which are belong to one class. Thus, the training data on each node is highly skewed. In Figure 5, observe that both fully synchronous SGD and Local SGD are pretty unstable in this case. Overlap-Local-SGD not only reduces the total training time but also yields better convergence in terms of error-versus-iterations (see Figure 5(c)). Compared to CoCoD-SGD (see Table 2), Overlap-Local-SGD still can achieve comparable test accuracy and overcome the divergence issue when τ is large.

5. CONVERGENCE ANALYSIS

In this section, we will provide a convergence guarantee for Overlap-Local-SGD under non-convex objectives, which is common for deep neural networks. The analysis is based on the following assumptions:

1. Each local objective function $F_i(\boldsymbol{x}) := \mathbb{E}_{\boldsymbol{s} \sim \mathcal{D}_i}[\ell(\boldsymbol{x}; \boldsymbol{s})]$ is L-smooth: $\|\nabla F_i(\boldsymbol{x}) - \nabla F(\boldsymbol{y})\| \leq L \|\boldsymbol{x} - \boldsymbol{y}\|, \forall i \in [1, m]$.

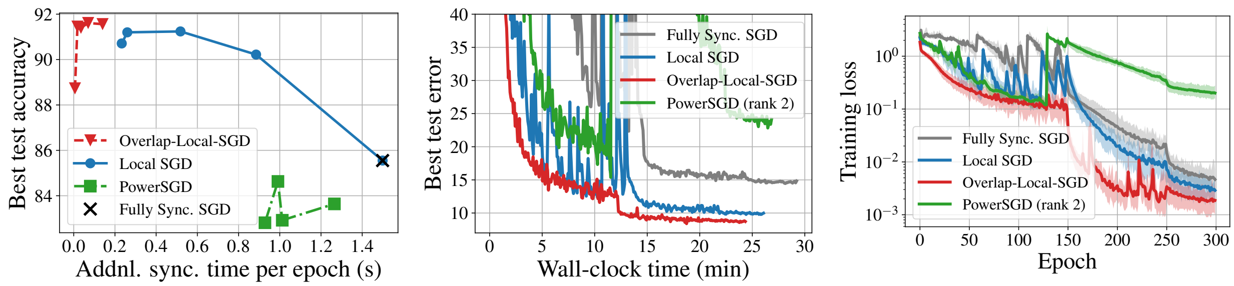


(a) Error-runtime trade-off.

(b) Validation curves.

(c) Training curves.

Fig. 4: Comparison of communication-efficient SGD methods in **IID** data partition setting. In (a), the number of local updates of Local SGD method is taken from $\{1, 2, 4, 8, 24\}$. In (b) and (c), we fix $\tau = 2$.



(a) Error-runtime trade-off.

(b) Validation curves.

(c) Training curves.

Fig. 5: Comparison of communication-efficient SGD methods in **non-IID** data partition setting. In (a), the number of local updates of Local SGD method is taken from $\{1, 2, 4, 8, 24\}$. In (b) and (c), we fix $\tau = 2$. Overlap-Local-SGD is much more stable than other methods.

2. The stochastic gradients are unbiased estimators of local objectives' gradients, *i.e.*, $\mathbb{E}_{\xi\sim\mathcal{D}_i}[g_i(\boldsymbol{x};\xi)] = \nabla F_i(\boldsymbol{x})$.
3. The variance of stochastic gradients is bounded by a non-negative constant: $\mathbb{E}_{\xi\sim\mathcal{D}_i}[\|g_i(\boldsymbol{x};\xi) - \nabla F_i(\boldsymbol{x})\|^2] \leq \sigma^2$.
4. The average deviation of local gradients is bounded by a non-negative constant: $\frac{1}{m}\sum_{i=1}^{m}\|\nabla F_i(\boldsymbol{x}) - \nabla F(\boldsymbol{x})\| \leq \kappa^2$.

Formally, we have the following theorem which can guarantee Overlap-Local-SGD converges to a stationary point of the non-convex objective function.

Theorem 1. *Suppose all local models and anchor model are initialized at the same point $\boldsymbol{x}_0^{(i)} = \boldsymbol{z}_0$ for all $i \in \{1, \ldots, m\}$. Under Assumptions 1 to 4, if the learning rate is set as $\gamma = \frac{1}{L}\sqrt{\frac{m}{K}}$, and the total iterations K satisfies $K \geq 60m\tau^2/\alpha^2$, then we have*

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\left[\|\nabla F(\boldsymbol{y}_k)\|^2\right] \leq \frac{4L[F(\boldsymbol{y}_0) - F_{inf}]}{(1-\alpha)\sqrt{mK}} + \frac{2(1-\alpha)\sigma^2}{\sqrt{mK}} + \frac{2m\sigma^2}{K}\left[\frac{2}{(2-\alpha)\alpha}\tau - 1\right] + \frac{2m\tau^2\kappa^2}{\alpha^2 K} \tag{11}$$

$$= \mathcal{O}\left(\frac{1}{\sqrt{mK}}\right) + \mathcal{O}\left(\frac{1}{K}\right). \tag{12}$$

where $\boldsymbol{y}_k = (1-\alpha)\sum_{i=1}^{m}\boldsymbol{x}_k^{(i)} + \alpha\boldsymbol{z}_k$ and F_{inf} is the lower bound of the objective value.

Due to space limitation, please refer to Appendix A to check the proof details. Briefly, the proof technique is inspired by [10]. The key challenge is that the mixing matrix of Overlap-Local-SGD is column-stochastic instead of doubly- or row-stochastic [13]. It is worth highlighting that the analysis can be generalized to other column stochastic matrices rather than the specific form given in (6). Theorem 1 also shows that when the learning rate is configured properly and the total iterations K is sufficiently large, the error bound of Overlap-Local-SGD will be dominated by $1/\sqrt{mK}$, matching the same rate as fully synchronous SGD.

6. CONCLUSIONS

In this paper, we propose a novel distributed training algorithm named Overlap-Local-SGD. It allows workers to perform local updates and overlaps the local computation and communication. Experimental results on CIFAR-10 show that Overlap-Local-SGD can achieve the best error-runtime trade-off among multiple popular communication-efficient traing methods, such as Local SGD and PowerSGD. Moreover, when worker nodes have non-IID data partitions, Overlap-Local-SGD not only reduces the total runtime but also converges faster than other methods. We further prove that Overlap-Local-SGD can converge to a stationary point of a smooth and non-convex objective function. While the experiments only focus on image classification, the key idea of Overlap-Local-SGD can be easily extended to other training task and first-order optimization algorithms, such as Adam [23] for neural machine translation [22].

7. REFERENCES

[1] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[2] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever, "Language models are unsupervised multi-task learners," Open AI tech. report, Feb. 2019.

[3] Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su, "Scaling distributed machine learning with the parameter server.," in *OSDI*, 2014, vol. 14, pp. 583–598.

[4] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He, "Accurate, large minibatch SGD: Training ImageNet in 1 hour," *arXiv preprint arXiv:1706.02677*, 2017.

[5] Thijs Vogels, Sai Praneeth Karimireddy, and Martin Jaggi, "PowerSGD: Practical low-rank gradient compression for distributed optimization," in *Advances in Neural Information Processing Systems*, 2019.

[6] Sanghamitra Dutta, Gauri Joshi, Soumyadip Ghosh, Parijat Dube, and Priya Nagpurkar, "Slow and stale gradients can win the race: Error-runtime trade-offs in distributed SGD," in *International Conference on Artificial Intelligence and Statistics*, 2018, pp. 803–812.

[7] Nuwan Ferdinand, Haider Al-Lawati, Stark Draper, and Matthew Nokelby, "Anytime minibatch: Exploiting stragglers in online distributed optimization," in *International Conference on Learning Representations*, 2019.

[8] Sebastian U Stich, "Local SGD converges fast and communicates little," in *International Conference on Learning Representations*, 2019.

[9] Hao Yu, Sen Yang, and Shenghuo Zhu, "Parallel restarted SGD with faster convergence and less communication: Demystifying why model averaging works for deep learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019, vol. 33, pp. 5693–5700.

[10] Jianyu Wang and Gauri Joshi, "Cooperative SGD: A unified framework for the design and analysis of communication-efficient SGD algorithms," *arXiv preprint arXiv:1808.07576*, 2018.

[11] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, 2017, pp. 1273–1282.

[12] Jianyu Wang and Gauri Joshi, "Adaptive communication strategies to achieve the best error-runtime trade-off in local-update SGD," *CoRR*, vol. abs/1810.08313, 2018.

[13] Angelia Nedić, Alex Olshevsky, and Michael G. Rabbat, "Network topology and communication-computation tradeoffs in decentralized optimization," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 953–976, 2018.

[14] Mahmoud Assran, Nicolas Loizou, Nicolas Ballas, and Michael Rabbat, "Stochastic gradient push for distributed deep learning," in *International Conference on Machine Learning*, 2019.

[15] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton, "On the importance of initialization and momentum in deep learning," in *International Conference on Machine Learning*, 2013, pp. 1139–1147.

[16] Jianyu Wang, Vinayak Tantia, Nicolas Ballas, and Michael Rabbat, "SlowMo: Improving communication-efficient distributed SGD with slow momentum," *arXiv preprint arXiv:1910.00643*, Oct. 2019.

[17] S. Zhang, A. Choromanska, and Y. LeCun, "Deep learning with elastic averaged SGD," in *Advances in Neural Information Processing Systems*, 2015, pp. 685–693.

[18] Shuheng Shen, Linli Xu, Jingchang Liu, Xianfeng Liang, and Yifei Cheng, "Faster distributed deep net training: Computation and communication decoupled stochastic gradient descent," in *IJCAI*, 2019.

[19] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton, "Learning multiple layers of features from tiny images," *CIFAR-10 (Canadian Institute for Advanced Research)*, 2009.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[21] Adam Paszke, Soumith Chintala, Ronan Collobert, Koray Kavukcuoglu, Clement Farabet, Samy Bengio, Iain Melvin, Jason Weston, and Johnny Mariethoz, "Pytorch: Tensors and dynamic neural networks in python with strong gpu acceleration," 2017.

[22] Myle Ott, Grangier David Edunov, Sergey, and Michael Auli, "Scaling neural machine translation," in *Conference on Machine Translation (WMT)*, 2018.

[23] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.

[24] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd, "The pagerank citation ranking: Bringing order to the web.," Tech. Rep., Stanford InfoLab, 1999.

[25] Taher Haveliwala and Sepandar Kamvar, "The second eigenvalue of the google matrix," Tech. Rep., Stanford, 2003.

A. PROOF OF THEOREM 1

Recall the update rule of Overlap-Local-SGD:

$$\boldsymbol{X}_{k+1} = [\boldsymbol{X}_k - \gamma \boldsymbol{G}_k]\boldsymbol{W}_k. \tag{13}$$

Matrix $\boldsymbol{W}_k$ is column-stochastic as defined in (6). To be specific,

$$\boldsymbol{W}_k = \begin{cases} \boldsymbol{P} & (k+1) \bmod \tau = 0 \\ \boldsymbol{I} & \text{Otherwise.} \end{cases} \tag{14}$$

where matrix $\boldsymbol{P} \in \mathbb{R}^{(m+1)\times(m+1)}$ is defined as

$$\boldsymbol{P} = \begin{bmatrix} (1-\alpha)\boldsymbol{I} & (1-\alpha)\boldsymbol{1}_m/m \\ \alpha \boldsymbol{1}_m & \alpha \end{bmatrix}. \tag{15}$$

There must be a vector $\boldsymbol{v} \in \mathbb{R}^{m+1}$ such that $\boldsymbol{P}\boldsymbol{v} = \boldsymbol{v}$ and hence $\boldsymbol{W}_k\boldsymbol{v} = \boldsymbol{v}$. In particular, for the matrix given in (6), $\boldsymbol{v} = [(1-\alpha)\boldsymbol{1}/m, \alpha]$. Multiplying $\boldsymbol{v}$ on both sides of (13), we have

$$\boldsymbol{X}_{k+1}\boldsymbol{v} = \boldsymbol{X}_k\boldsymbol{v} - \gamma\boldsymbol{G}_k\boldsymbol{v} \tag{16}$$

$$= \boldsymbol{X}_k\boldsymbol{v} - \frac{(1-\alpha)\gamma}{m}\sum_{i=1}^{m} g_i(\boldsymbol{x}_k^{(i)}; \xi_k^{(i)}). \tag{17}$$

For the ease of writing, we introduce a virtual sequence $\boldsymbol{y}_k := \boldsymbol{X}_k\boldsymbol{v} = (1-\alpha)\sum_{i=1}^{m}\boldsymbol{x}_k^{(i)}/m + \alpha\boldsymbol{z}_k$, and define effective learning rate as $\gamma_{\text{eff}} := (1-\alpha)\gamma$. Consequently, we get an equivalent vector-form update rule for Overlap-Local-SGD as follows:

$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k - \gamma_{\text{eff}}\frac{1}{m}\sum_{i=1}^{m} g_i(\boldsymbol{x}_k^{(i)}; \xi_k^{(i)}). \tag{18}$$

Then, we can directly apply Lemma 3 in [10] and obtain the following (when $\gamma_{\text{eff}}L \leq 1$)

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\left[\|\nabla F(\boldsymbol{y}_k)\|^2\right] \leq \frac{2[F(\boldsymbol{y}_0) - F_{\text{inf}}]}{\gamma_{\text{eff}}K} + \frac{\gamma_{\text{eff}}L\sigma^2}{m} + \frac{L^2}{Km}\sum_{k=0}^{K-1}\sum_{i=1}^{m}\mathbb{E}\left[\left\|\boldsymbol{y}_k - \boldsymbol{x}_k^{(i)}\right\|^2\right]. \tag{19}$$

Note that

$$\sum_{i=1}^{m}\left\|\boldsymbol{y}_k - \boldsymbol{x}_k^{(i)}\right\|^2 \leq \sum_{i=1}^{m}\left\|\boldsymbol{y}_k - \boldsymbol{x}_k^{(i)}\right\|^2 + \|\boldsymbol{y}_k - \boldsymbol{z}_k\|^2 \tag{20}$$

$$= \left\|\boldsymbol{X}_k\left(\boldsymbol{I} - \boldsymbol{v}\boldsymbol{1}^\top\right)\right\|_{\text{F}}^2. \tag{21}$$

According to the update rule (13) and repeatedly using the fact $\boldsymbol{W}_k\boldsymbol{v} = \boldsymbol{v}, \boldsymbol{1}^\top\boldsymbol{W}_k = \boldsymbol{1}^\top$ and $\boldsymbol{v}^\top\boldsymbol{1} = 1$, we have

$$\boldsymbol{X}_k\left(\boldsymbol{I} - \boldsymbol{v}\boldsymbol{1}^\top\right)$$

$$= (\boldsymbol{X}_{k-1} - \gamma_{\text{eff}}\boldsymbol{G}_{k-1})\boldsymbol{W}_k\left(\boldsymbol{I} - \boldsymbol{v}\boldsymbol{1}^\top\right) \tag{22}$$

$$= \boldsymbol{X}_{k-1}\left(\boldsymbol{W}_{k-1} - \boldsymbol{v}\boldsymbol{1}^\top\right) - \gamma_{\text{eff}}\boldsymbol{G}_{k-1}\left(\boldsymbol{W}_{k-1} - \boldsymbol{v}\boldsymbol{1}^\top\right) \tag{23}$$

$$= \boldsymbol{X}_0\left(\prod_{j=0}^{k-1}\boldsymbol{W}_j - \boldsymbol{v}\boldsymbol{1}^\top\right) - \gamma_{\text{eff}}\sum_{j=0}^{k-1}\boldsymbol{G}_j\left(\prod_{s=j}^{k-1}\boldsymbol{W}_s - \boldsymbol{v}\boldsymbol{1}^\top\right) \tag{24}$$

$$= \boldsymbol{x}_0\boldsymbol{1}^\top\left(\prod_{j=0}^{k-1}\boldsymbol{W}_j - \boldsymbol{v}\boldsymbol{1}^\top\right) - \gamma_{\text{eff}}\sum_{j=0}^{k-1}\boldsymbol{G}_j\left(\prod_{s=j}^{k-1}\boldsymbol{W}_s - \boldsymbol{v}\boldsymbol{1}^\top\right) \tag{25}$$

$$= -\gamma_{\text{eff}}\sum_{j=0}^{k-1}\boldsymbol{G}_j\left(\prod_{s=j}^{k-1}\boldsymbol{W}_s - \boldsymbol{v}\boldsymbol{1}^\top\right). \tag{26}$$

Therefore,

$$\sum_{i=1}^{m}\left\|\boldsymbol{y}_k - \boldsymbol{x}_k^{(i)}\right\|^2 \leq \gamma_{\text{eff}}^2\left\|\sum_{j=0}^{k-1}\boldsymbol{G}_j\left(\prod_{s=j}^{k-1}\boldsymbol{W}_s - \boldsymbol{v}\boldsymbol{1}^\top\right)\right\|_{\text{F}}^2. \tag{27}$$

Here we observe that the analysis of Overlap-Local-SGD is very similar to the general analysis in [10]. The difference is that we require $\boldsymbol{W}_k$ to be column-stochastic instead of symmetric and doubly-stochastic. As a result, $\prod_{s=0}^{\infty}\boldsymbol{W}_s$ converges to $\boldsymbol{v}\boldsymbol{1}^\top$ rather than $\boldsymbol{1}\boldsymbol{1}^\top/m$. Then, one can directly re-use the intermediate results in [10] and get that

$$\frac{1}{Km}\sum_{k=0}^{K-1}\sum_{i=1}^{m}\mathbb{E}\left[\left\|\boldsymbol{y}_k - \boldsymbol{x}_k^{(i)}\right\|^2\right] \leq \gamma^2\sigma^2\left(\frac{1+\zeta^2}{1-\zeta^2}\tau - 1\right) + \frac{\gamma^2\tau^2}{1-\zeta}\left(\frac{2\zeta^2}{1+\zeta} + \frac{2\zeta}{1-\zeta} + \frac{\tau-1}{\tau}\right)\frac{1}{Km}\sum_{k=0}^{K-1}\sum_{i=1}^{m}\mathbb{E}\left[\left\|\nabla F_i(\boldsymbol{x}_k^{(i)})\right\|^2\right] \tag{28}$$

where $\zeta := \left\|\boldsymbol{P} - \boldsymbol{v}\boldsymbol{1}^\top\right\|_2$. In order to guarantee the upper bound (28) make sense, ζ should be strictly smaller than 1. Now, we are going to provide an analytical expression of ζ for the specific $\boldsymbol{P}$ chosen in Overlap-Local-SGD. One can also design other forms of $\boldsymbol{P}$ as long as $\zeta < 1$.

Observe that the matrix $\boldsymbol{P}$ can be decomposed into two parts:

$$\boldsymbol{P} = (1-\alpha)\boldsymbol{A} + \alpha\boldsymbol{b}\boldsymbol{1}^\top \tag{29}$$

where $\boldsymbol{b} = [0, \ldots, 0, 1] \in \mathbb{R}^{m+1}$ and

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{1}_m/m \\ \boldsymbol{0} & 0 \end{bmatrix}. \tag{30}$$

Both $\boldsymbol{A}$ and $\boldsymbol{b}\boldsymbol{1}^\top$ are column-stochastic matrix. Actually, the formulation (29) is widely used in the PageRank algorithm [24]. It is proved that [25]: $\zeta = \left\|\boldsymbol{P} - \boldsymbol{v}\boldsymbol{1}^\top\right\|_2 \leq (1-\alpha)$. Plugging the expression of ζ into (28) and further relaxing the upper bound, we obtain:

$$\frac{1}{Km}\sum_{k=0}^{K-1}\sum_{i=1}^{m}\mathbb{E}\left[\left\|\boldsymbol{y}_k - \boldsymbol{x}_k^{(i)}\right\|^2\right] \leq \gamma^2\sigma^2\left[\frac{2}{(2-\alpha)\alpha}\tau - 1\right] + \frac{5\gamma^2\tau^2}{\alpha^2}\frac{1}{Km}\sum_{k=0}^{K-1}\sum_{i=1}^{m}\mathbb{E}\left[\left\|\nabla F_i(\boldsymbol{x}_k^{(i)})\right\|^2\right]. \tag{31}$$

Furthermore, note that

$$\frac{1}{m}\sum_{i=1}^{m}\left\|\nabla F_i(\boldsymbol{x}_k^{(i)})\right\|^2 \tag{32}$$

$$\leq \frac{3}{m}\sum_{i=1}^{m}\left\|\nabla F_i(\boldsymbol{x}_k^{(i)}) - \nabla F_i(\boldsymbol{y}_k)\right\|^2 + \frac{3}{m}\sum_{i=1}^{m}\|\nabla F_i(\boldsymbol{y}_k) - \nabla F(\boldsymbol{y}_k)\|^2 + 3\|\nabla F(\boldsymbol{y}_k)\|^2 \tag{33}$$

$$\leq \frac{3L^2}{m}\sum_{i=1}^{m}\mathbb{E}\left[\left\|\boldsymbol{y}_k - \boldsymbol{x}_k^{(i)}\right\|^2\right] + 3\kappa^2 + 3\|\nabla F(\boldsymbol{y}_k)\|^2. \tag{34}$$

Combing (31) and (34), we have

$$\left(1-\frac{15\gamma^2L^2\tau^2}{\alpha^2}\right)\frac{1}{Km}\sum_{k=0}^{K-1}\sum_{i=1}^{m}\mathbb{E}\left[\left\|\boldsymbol{y}_k-\boldsymbol{x}_k^{(i)}\right\|^2\right]$$
$$\leq\gamma^2\sigma^2\left[\frac{2}{(2-\alpha)\alpha}\tau-1\right]+\frac{15\gamma^2\tau^2\kappa^2}{\alpha^2}+$$
$$\frac{15\gamma^2\tau^2}{\alpha^2K}\sum_{k=0}^{K-1}\mathbb{E}\left[\|\nabla F(\boldsymbol{y}_k)\|^2\right]. \tag{35}$$

For the ease of writing, define $D=15\gamma^2L^2\tau^2/\alpha^2$. Then,

$$\frac{L^2}{Km}\sum_{k=0}^{K-1}\sum_{i=1}^{m}\mathbb{E}\left[\left\|\boldsymbol{y}_k-\boldsymbol{x}_k^{(i)}\right\|^2\right]$$
$$\leq\frac{\gamma^2L^2\sigma^2}{1-D}\left[\frac{2}{(2-\alpha)\alpha}\tau-1\right]+\frac{\gamma^2L^2\tau^2\kappa^2}{\alpha^2(1-D)}+$$
$$\frac{D}{1-D}\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\left[\|\nabla F(\boldsymbol{y}_k)\|^2\right]. \tag{36}$$

Substituting (36) into (19), one can get

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\left[\|\nabla F(\boldsymbol{y}_k)\|^2\right]$$
$$\leq\frac{2[F(\boldsymbol{y}_0)-F_{\text{inf}}]}{\gamma_{\text{eff}}K}+\frac{\gamma_{\text{eff}}L\sigma^2}{m}+$$
$$\frac{\gamma^2L^2\sigma^2}{1-D}\left[\frac{2}{(2-\alpha)\alpha}\tau-1\right]+\frac{\gamma^2L^2\tau^2\kappa^2}{\alpha^2(1-D)}+$$
$$\frac{D}{1-D}\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\left[\|\nabla F(\boldsymbol{y}_k)\|^2\right]. \tag{37}$$

After minor rearranging, it follows that

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\left[\|\nabla F(\boldsymbol{y}_k)\|^2\right]$$
$$\leq\left[\frac{2[F(\boldsymbol{y}_0)-F_{\text{inf}}]}{\gamma_{\text{eff}}K}+\frac{\gamma_{\text{eff}}L\sigma^2}{m}\right]\frac{1-D}{1-2D}+$$
$$\frac{\gamma^2L^2\sigma^2}{1-2D}\left[\frac{2}{(2-\alpha)\alpha}\tau-1\right]+\frac{\gamma^2L^2\tau^2\kappa^2}{\alpha^2(1-2D)}. \tag{38}$$

When the learning rate is set to $\gamma=\frac{1}{L}\sqrt{\frac{m}{K}}$, $D=15m\tau^2/(\alpha^2K)$. If $K\geq 60m\tau^2/\alpha^2$, then $1-2D\geq 1/2$ and hence,

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\left[\|\nabla F(\boldsymbol{y}_k)\|^2\right]$$
$$\leq\frac{4[F(\boldsymbol{y}_0)-F_{\text{inf}}]}{\gamma_{\text{eff}}K}+\frac{2\gamma_{\text{eff}}L\sigma^2}{m}+$$
$$2\gamma^2L^2\sigma^2\left[\frac{2}{(2-\alpha)\alpha}\tau-1\right]+\frac{2\gamma^2L^2\tau^2\kappa^2}{\alpha^2} \tag{39}$$
$$=\frac{4L[F(\boldsymbol{y}_0)-F_{\text{inf}}]}{(1-\alpha)\sqrt{mK}}+\frac{2(1-\alpha)\sigma^2}{\sqrt{mK}}+$$
$$\frac{2m\sigma^2}{K}\left[\frac{2}{(2-\alpha)\alpha}\tau-1\right]+\frac{2m\tau^2\kappa^2}{\alpha^2K} \tag{40}$$
$$=\mathcal{O}\left(\frac{1}{\sqrt{mK}}\right)+\mathcal{O}\left(\frac{1}{K}\right). \tag{41}$$

Here we complete the proof of Theorem 1.

B. ADDITIONAL EXPERIMENTAL RESULTS

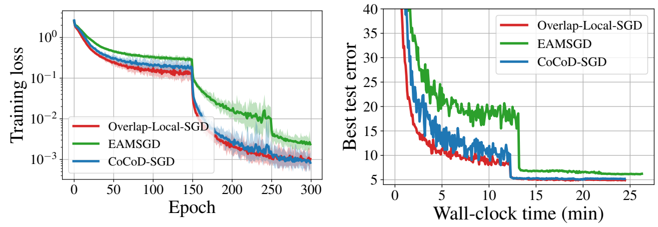


Fig. 6: Comparison to CoCoD-SGD [18] and EAMSGD [17]. In all algorithms, the number of local updates τ is fixed as 2. Overlap-Local-SGD slightly improves the loss-versus-iterations convergence of CoCoD-SGD.