OPT2025: 17th Annual Workshop on Optimization for Machine Learning

DSGD-AC: controlled consensus errors improve generalization in decentralized training

Zesen Wang ZESEN@KTH.SE
Mikael Johansson MIKAELJ@KTH.SE
KTH Royal Institute of Technology, Stockholm, Sweden

Abstract

Decentralized SGD aggressively enforces consensus among workers by driving model disagreements to zero as learning rates decay. We argue that this vanishing consensus eliminates beneficial structured perturbations that promote better generalization, similar to Sharpness-Aware Minimization (SAM) but without additional gradient computations. We propose Decentralized SGD with Adaptive Consensus (DSGD-AC), which intentionally preserves non-vanishing consensus errors during late-stage training. Our key insight is that consensus errors are data-dependent and correlate with ascent directions on local datasets, providing implicit sharpness regularization over data distributions. Empirically, DSGD-AC improves the generalization over SGD with negligible computational overhead on classic deep learning tasks. By treating consensus as a tunable resource rather than a nuance to minimize, DSGD-AC offers a simple yet effective approach to improve generalization in decentralized training.

1. Introduction

In large-scale deep learning, decentralized optimization, where workers exchange model parameters only with neighbors, reduces the overhead of global synchronization and avoids costly all-reduce communication [1, 7]. Decentralized algorithms, therefore, offer significant runtime and scalability advantages in practice [2, 9, 13]. Standard analyses of decentralized SGD (DSGD) place strong emphasis on consensus: the per-worker deviations from the network average are driven to zero so that the decentralized average closely approximates a centralized iterate. While vanishing consensus simplifies convergence analysis, it also eliminates inter-worker model diversity that can act as a source of structured perturbations. Prior work shows that explicit perturbations (e.g., Sharpness-Aware Minimization [5, 8, 10]) or intrinsic DSGD noise can promote flatter minima and improved generalization [16]; from this perspective, aggressively enforcing consensus in decentralized training may discard a low-cost, data-dependent mechanism that benefits generalization.

Motivated by the observation in the training dynamics of DSGD, we propose to *treat consensus as a tunable resource*. We propose Decentralized SGD with Adaptive Consensus (DSGD-AC), a simple modification of DSGD in which the consensus correction is scaled by a time-dependent coefficient derived from the learning-rate schedule. DSGD-AC intentionally preserves non-vanishing inter-worker differences during the late-training phase. The modification requires no extra gradient evaluations and only standard decentralized communication.

Contributions: (1) We identify vanishing consensus in DSGD as a potential limiter of generalization and clarify the mechanism by which persisting consensus errors can act as implicit, low-cost perturbations. (2) We propose DSGD-AC, a lightweight adaptive-mixing rule that ties consensus

strength to the learning-rate schedule. (3) We provide analytic interpretation and empirical evidence showing improved test performance on representative deep learning tasks with negligible extra cost.

2. Notation and Background

Decentralized Optimization We consider the standard decentralized optimization setup with N workers. Each worker $i \in [N]$ holds a local objective determined by its local dataset $\mathcal{D}_i$: $F_i(x) = \mathbb{E}_{\xi \sim \mathcal{D}_i}[f(x;\xi)]$. The standard decentralized optimization problem is written as

$$\min_{x_1, x_2, \cdots, x_N} \frac{1}{N} \sum_{i=1}^{N} F_i(x_i), \quad \text{subject to } x_i = x_j, \forall i, j \in [N] \tag{1}$$

We let $\bar{x}^{(t)} = \frac{1}{N}\sum_{i=1}^{N} x_i^{(t)}$ denote the global average model. This is the model that would typically be deployed and evaluated in practice.

Decentralized SGD (DSGD) The update of DSGD [9] on worker i is:

$$x_i^{(t+1)} = x_i^{(t)} - \alpha^{(t)} \nabla f(x_i^{(t)}; \xi_i^{(t)}) + \sum_{j \in \mathcal{N}(i)} W_{ij}^{(t)} (x_j^{(t)} - x_i^{(t)}) \tag{2}$$

where $\mathcal{N}(i)$ is the set of neighbors of worker i (including itself), W_{ij} is a doubly stochastic matrix defining the weights of the edges ($W_{ij} = 0$ if worker i is a neighbor of worker j), and $\xi_i^{(t)}$ denotes the stochastic mini-batch sampled by worker i at iteration t. We denote the consensus error between worker i and the global average by $e_i = x_i - \bar{x}$.

Sharpness-aware minimization (SAM) and connection to DSGD Sharpness-aware minimization (SAM) seeks solutions that are robust to worst-case or random perturbations of the weights [4, 5, 14]. A recent observation [16, Theorem 1] suggests that, in DSGD, the expected update of $\bar{x}^{(t)}$ aligns with the gradient of an average-direction SAM objective whose perturbation covariance is given by the weight diversity. Our experiments, however, demonstrate that disagreements are data-dependent even with i.i.d. reshuffling; we therefore *control*—rather than eliminate—consensus errors via an adaptive consensus strength.

Practical remarks about distributed data sampler We let $D_i^{(e)}$ denote the local dataset assigned to worker i in epoch e. The common practice for the distributed data sampler (also in our experiments) is to reshuffle the full dataset at the start of each epoch and re-assign it to workers. This makes $F_i^{(e)}$ (the epoch-dependent local objective) vary across epochs while preserving overall data coverage. The empirical experiment in this work measures local losses using this convention.

3. Method

3.1. Key finding: vanishing consensus error in DSGD

We start by empirically investigating the behavior of consensus errors when training a Wide ResNet (WRN28-10) [15] on CIFAR-10 [6] with DSGD. In this experiment, we employ a cosine annealing learning rate schedule [11] with a linear warm-up during the first 10 epochs (Figure 1, left). We track the average norm of the consensus errors and observe, as shown in blue in Figure 1 (right), that under DSGD the consensus errors gradually vanish as the learning rate decreases.

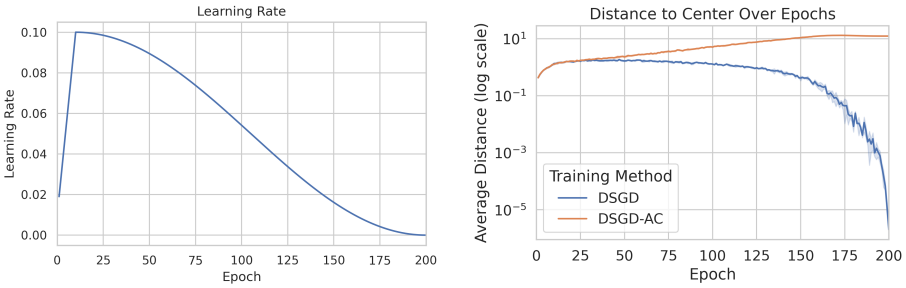

Figure 1: Decentralized training of WRN28-10 on CIFAR-10 (3 random runs for each algorithm). There are 8 workers, and the communication topology is the one-peer ring topology (time-varying). **Left**: Learning rate schedule (same for both algorithms). **Right**: Average norm of consensus errors evaluated at the end of every epoch ($\frac{1}{N}\sum_{i=1}^{N}\|x_i^{(eT)} - \bar{x}^{(eT)}\|$). p is set to 3 for DSGD-AC.

Claim 1 *With diminishing step sizes, DSGD loses the effect of the average-direction SAM.*

Claim 1 can be explained theoretically. By interpreting the mixing step as a gradient step on a quadratic consensus penalty, one obtains the per-step surrogate

$$J^{(t)}(x_1, \cdots, x_n) = \underbrace{\frac{1}{N}\sum_{i=1}^{N} F_i(\bar{x}^{(t)})}_{\text{objective on deployed model}} + \underbrace{\frac{1}{N}\sum_{i=1}^{N}[F_i(x_i^{(t)}) - F_i(\bar{x}^{(t)})]}_{\text{sharpness}} + \underbrace{\frac{1}{2N\alpha^{(t)}}\sum_{i,j\in[N]} W_{ij}\|x_i^{(t)} - x_j^{(t)}\|^2}_{\text{consensus regularizer}} \quad (3)$$

With symmetric mixing weights and no momentum or adaptivity, each DSGD step is exactly a (stochastic) gradient on J. Thus, when $\alpha^{(t)}$ goes to 0, the consensus regularizer dominates the objective function, which minimizes the consensus errors but also eliminates the sharpness term because $f_i(x_i^{(t)}) \approx f_i(\bar{x}^{(t)})$ as $x_i^{(t)} \approx \bar{x}$.

3.2. Algorithm: Decentralized SGD with adaptive consensus (DSGD-AC)

The algorithm of the proposed method is shown in Algorithm 1, and the main difference from DSGD is highlighted. Compared with DSGD, the proposed variant includes an adaptive factor to maintain non-diminishing consensus errors intentionally.

Note that $\alpha^{(t)}$ is determined by the employed learning rate schedule (cosine annealing, for example), and $\alpha_{\min}$ and $\alpha_{\max}$ are the minimal and maximal learning rates throughout the training. The algorithm takes the global average of local models as the deployed model.

3.3. Main characteristics of DSGD-AC

Claim 2 *DSGD-AC maintains non-vanishing and useful consensus errors.*

The original motivation is to maintain non-diminishing consensus errors. Therefore, we multiply the weight of the consensus regularizer by an adaptive γ, which directly leads to the DSGD-AC algorithm. The objective function of DSGD-AC is mostly the same as DSGD. Only the weight of

Algorithm 1: Decentralized SGD with adaptive consensus (DSGD-AC) on worker i

Data: Dataset (D), the number of workers (N), the number of epoch (E), the number of batches per epoch (T), initial parameter ($x^{(0)}$), and a hyperparameter ($p \in \mathbb{R}^+$).

Result: Deployed model $\bar{x} = \frac{1}{n}\sum_{j=1}^{n} x_j^{(TE)}$

$x_1^{(0)} = x_2^{(0)} = \cdots = x_n^{(0)} = x^{(0)}$

for $t = 1$ *to* TE **do**

$g_i^{(t)} = \nabla f(x_i^{(t-1)}; \xi_i^{(t)})$

$\gamma^{(t)} = \left[(\alpha^{(t)} - \alpha_{\min})/(\alpha_{\max} - \alpha_{\min})\right]^p$

$x_i^{(t)} = x_i^{(t-1)} - \alpha^{(t)} g_i^{(t)} + \gamma^{(t)} \sum_{j \in \mathcal{N}(i)} W_{ij}(x_j^{(t-1)} - x_i^{(t-1)})$

end

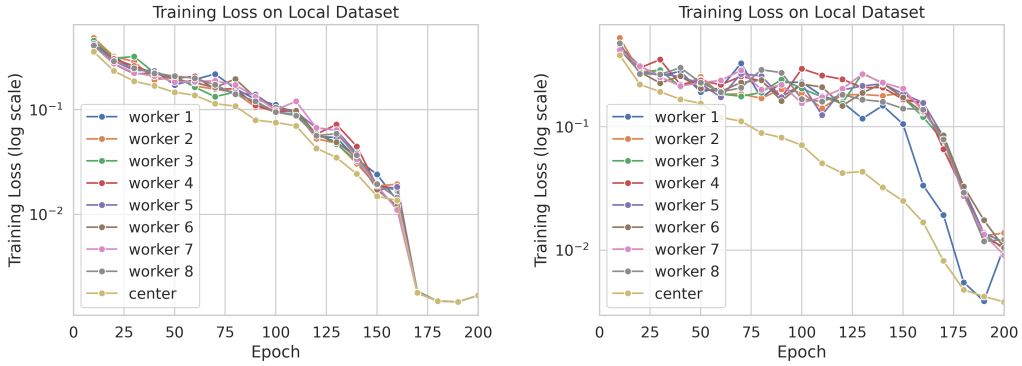

Figure 2: The losses on the models of workers and the global center on the local dataset of worker 1 used in the last epoch, which are $L(x_i^{(eT)}; D_1^{(eT)})$ and $L(\bar{x}^{(eT)}; D_1^{(eT)})$ for $i \in \{1, \cdots, N\}$ and $e \in \{10, 20, \cdots, 200\}$. **Left**: DSGD. **Right**: DSGD-AC.

the consensus regularizer becomes $\gamma^{(t)}/(2N\alpha^{(t)})$. As shown in Figure 1, the norm of the consensus errors keeps increasing and does not vanish. Moreover, as shown by the empirical experiments in Section 4, DSGD-AC outperforms DSGD and SGD in generalization by a clear margin.

Claim 3 *Consensus errors are not independent of the epoch-dependent local data distributions.*

In the proof of [16, Theorem 1], it assumes the consensus error e_i is independent of its corresponding local function F_i. However, we demonstrate empirically that this independence assumption is violated in practice.

As described in the practical remarks in Section 2, reshuffling and partitioning are performed on the whole dataset at every epoch. Therefore, in the experiments, $F_i = F_j$ should hold for all $i, j \in [N]$. In expectation, all local models should perform similarly on all the local datasets.

We evaluate the local models on the local dataset of worker i used in the last epoch. To be specific, we evaluate the loss $L(x_i^{eT}; D_1^{(e)})$, where T is the number of local batches in one epoch (as described in Algorithm 1). The results are shown in Figure 2. For DSGD, all workers achieve comparable performance on the dataset of worker 1. In contrast, under DSGD-AC, worker 1's model

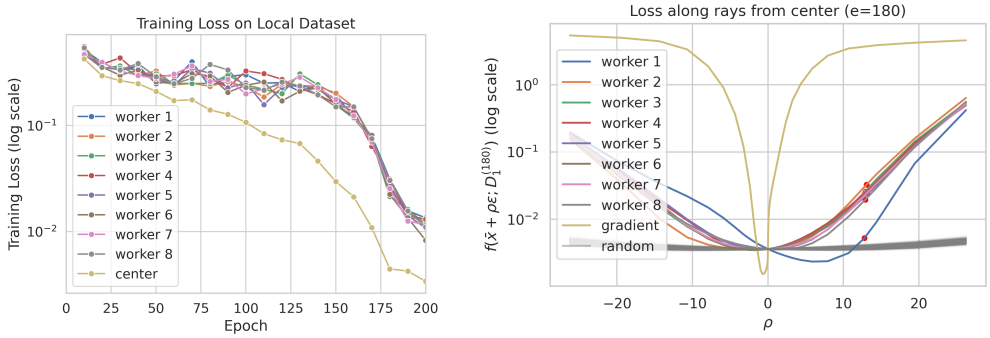

Figure 3: **Left**: Same metrics as in Fig. 2 except using the local dataset of worker 1 in the next epoch. **Right**: Losses $F(\bar{x}+\rho\cdot\epsilon/\|\epsilon\|; D_1^{(180)})$ along segments crossing the global center. worker i: along e_i, gradient: along $\nabla F(\bar{x}; D_1^{(180)})$, and random: 1500 random directions ϵ generated from normal distributions (as in [4]). The red dots correspond to $\rho = \|e_i\|$.

consistently attains significantly lower losses than the other workers, especially for $e \geq 150$. Similar patterns are observed when evaluating on the local datasets of other workers (see Appendix 6.4).

Claim 3 is particularly noteworthy, as it highlights a clear link between consensus errors and the underlying data distributions in DSGD-AC.

Remark 1 *DSGD-AC can be considered as the sharpness-aware minimization over data distributions.*

Given the findings in Claim 3, $F_i^{(e)}$ should not be considered as all the same even under i.i.d. data sampling. Therefore, for epoch e, DSGD-AC should have an epoch-dependent objective function, which can be written as

$$J^{(e)}(x_1, \cdots, x_n) = \frac{1}{N}\sum_{i=1}^{N} F_i^{(e)}(\bar{x}) + \frac{1}{N}\sum_{i=1}^{N}[F_i^{(e)}(x_i) - F_i^{(e)}(\bar{x})] + \frac{\gamma^{(t)}}{2N\alpha^{(t)}}\sum_{i,j\in[N]} W_{ij}\|x_i - x_j\|^2 \tag{4}$$

We first focus on the objective of the deployed model. Since $F_i^{(e)}$ is determined by its corresponding local dataset, and the union of all local datasets in every epoch e is always the whole dataset. Therefore, the following statement holds for all $e \in \mathbb{N}^+$.

$$\sum_{i=1}^{N}\sum_{s=1}^{T} f(x; \xi_i^{(e)}) = \sum_{i=1}^{N}\sum_{s=1}^{T} f(x; \xi_i^{(1)}) \Rightarrow \sum_{i=1}^{N} F_i^{(e)}(x) = \sum_{i=1}^{N} F_i^{(1)}(x) \tag{5}$$

which concludes that the first term in Eq. 4 remains the same for the same $\bar{x}$.

Figure 2 demonstrates that, except for worker 1, all other workers perform significantly worse on the local dataset of worker 1 in epoch e compared with the global average. In the next epoch, the samples in $D_1^{(e)}$ are expected to be distributed evenly to all local datasets. Even though it is not intentionally controlled, the default data shuffle strategy makes the direction of the consensus error e_i positively correlate with the ascent direction on the local dataset of other workers.

To provide empirical evidence for the statement, we again evaluate the local models on the local dataset of worker 1 but in the next epoch, which is $L(x_i^{(eT)}; D_1^{(e+1)})$. The left of Figure 3 validates the statement, and the global center always outperforms other workers. We also evaluate the loss on the line connecting $\bar{x}$ and x_i on $D_1^{(e)}$. The right of Figure 3 validates the correlation between consensus errors and the losses on the local datasets. Moreover, to demonstrate that the sharpness term in Eq. 4 is not equivalent to the average-direction SAM, we also sample a sufficient number of random directions as in [4], and the results demonstrate that the consensus errors in DSGD-AC have a significant correlation with the ascent direction, which captures sharpness information over data distributions. Even though the consensus errors in DSGD-AC do not align perfectly with the gradient (roughly the sharpest direction), it should be noted that the consensus errors do not incur extra cost as in the ascent-direction SAM.

4. Numerical Experiments

In this section, we present the results of the numerical experiments on image classification with wide ResNet [15] and on machine translation with transformers [12]. The results are shown in Table 1. We defer the detailed hyperparameters to Appendix 6.1 and the training curves to Appendix 6.2.

Table 1: Results on CIFAR-10 with WRN28-10 and WMT14 En→De with Transformer-big. Values are mean ± std over 3 runs.

Table 2: CIFAR-10 (WRN28-10)

Method	Acc. (%)	Comp. Cost
SGD	96.20 ± 0.14	1×
DSGD	96.13 ± 0.13	1×
DSGD-AC	<u>96.86 ± 0.24</u>	1×
SAM	**97.30 ± 0.08**	2×

Table 3: WMT14 En→De (Transformer-big)

Method	BLEU (%)
Adam	28.68 ± 0.07
DAdam	28.38 ± 0.22
DAdam-AC	**28.85 ± 0.18**

5. Conclusion & Discussion

We presented DSGD-AC, a lightweight modification to decentralized SGD that treats consensus as a tunable resource rather than a constraint to eliminate. By adaptively scaling consensus errors according to the learning-rate schedule, DSGD-AC preserves inter-worker diversity that serves as implicit sharpness-aware regularization, improving generalization with negligible cost.

Our empirical results suggest that the consensus errors that we keep throughout training are not just random noise, but are shaped by the loss. In our experiments, directions with higher Hessian curvature tend to amplify and retain more disagreement, and the consensus error naturally leans toward the ”sharp” directions of the loss landscape. A theoretical analysis (not included here) suggests that these disagreements act like a curvature-weighted regularizer on the deployed model, where perturbations in sharper directions hurt more, nudging the optimization trajectory toward flatter and more robust regions. In this sense, decentralized training implicitly injects structured, curvature-aware noise, offering a simple way to obtain SAM-like benefits without extra computation.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. {TensorFlow}: a system for {Large-Scale} machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

[2] Mahmoud Assran, Nicolas Loizou, Nicolas Ballas, and Mike Rabbat. Stochastic gradient push for distributed deep learning. In *International Conference on Machine Learning*, pages 344–353. PMLR, 2019.

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[4] Devansh Bisla, Jing Wang, and Anna Choromanska. Low-pass filtering sgd for recovering flat optima in the deep learning optimization landscape. In *International Conference on Artificial Intelligence and Statistics*, pages 8299–8339. PMLR, 2022.

[5] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

[6] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[7] Shen Li, Yanli Zhao, Rohan Varma, Omkar Salpekar, Pieter Noordhuis, Teng Li, Adam Paszke, Jeff Smith, Brian Vaughan, Pritam Damania, et al. Pytorch distributed: Experiences on accelerating data parallel training. *arXiv preprint arXiv:2006.15704*, 2020.

[8] Tao Li, Pan Zhou, Zhengbao He, Xinwen Cheng, and Xiaolin Huang. Friendly sharpness-aware minimization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5631–5640, 2024.

[9] 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. *Advances in neural information processing systems*, 30, 2017.

[10] Yong Liu, Siqi Mai, Minhao Cheng, Xiangning Chen, Cho-Jui Hsieh, and Yang You. Random sharpness-aware minimization. *Advances in neural information processing systems*, 35: 24543–24556, 2022.

[11] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[13] Zesen Wang, Zhang Jiaojiao, Wu Xuyang, and Mikael Johansson. From promise to practice: realizing high-performance decentralized training. In *The Thirteenth International Conference on Learning Representations*. ICLR, 2025.

[14] Kaiyue Wen, Tengyu Ma, and Zhiyuan Li. How does sharpness-aware minimization minimize sharpness? *arXiv preprint arXiv:2211.05729*, 2022.

[15] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[16] Tongtian Zhu, Fengxiang He, Kaixuan Chen, Mingli Song, and Dacheng Tao. Decentralized SGD and average-direction SAM are asymptotically equivalent. In *International Conference on Machine Learning*, pages 43005–43036. PMLR, 2023.

6. Appendix

6.1. Experiment Details

6.1.1. WIDERESNET ON CIFAR10

General:

- Number of epochs: 200
- Global batch size: 128
- Learning rate scheduler: Linear warm-up to 0.1 in the first 10 epochs, then use the cosine annealing learning rate scheduler until the end.
- Base optimizer: SGD + momentum (0.9). Weight decay is set to 0.0005.
- Data shuffle: Randomly shuffle and divide into N local datasets in every epoch.

Decentralized training:

- Number of workers: 8
- Communication topology: one-peer ring (switch between $i-1$ and $i+1$ neighbors in consecutive iterations)
- Exponent p is fixed to 3 in DSGD-AC (tuned by experiments), and γ is fixed to 1 in the warm-up stage.
- To avoid the influence of the mismatched batch normalization statistics (exponentially averaged mean and variance of the input) on the global average center, we run a full scan on all samples in the training set before the evaluation on the validation set to calibrate the statistics. While it is costly, it only needs one pass throughout the training if not logging the performance of the deployed model before the last epoch.

6.1.2. TRANSFORMER ON WMT14

General:

- Number of epochs: 20
- Global batch size: $\sim$25000 tokens
- Learning rate scheduler: Linear warm-up to 0.0005 in the first 4000 steps, then use $0.0005 \cdot (4000/t)^{0.5}$ until the end (t is the number of iterations).
- Base optimizer: Adam ($\beta_1 = 0.9, \beta_2 = 0.98$).
- Data shuffle: Randomly shuffle and divide into N local datasets in every epoch.

Decentralized training:

- Number of workers: 8
- Communication topology: one-peer ring (switch between left and right neighbors in consecutive iterations)
- Exponent p is fixed to 3 in DSGD-AC (tuned by experiments), and γ is fixed to 1 in the warm-up stage.
- Since only layer norm [3] is used in transformers, there is no need to calibrate the statistics for this experiment.

6.2. Training Curves of Wide ResNet Experiments

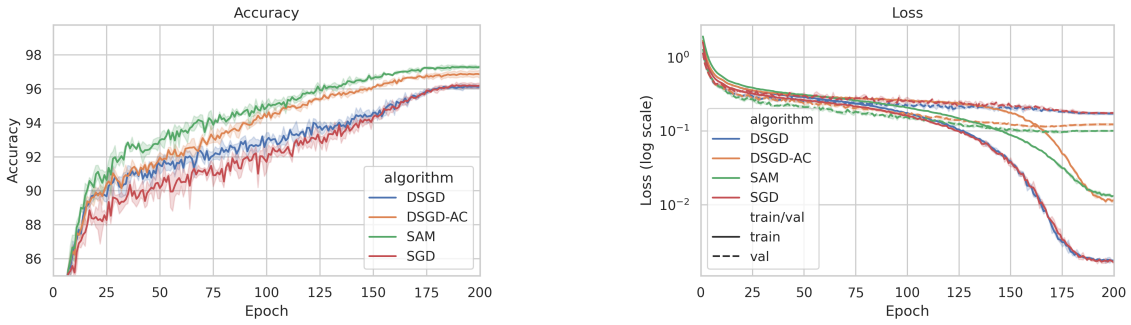

Figure 4: WRN28-10 on CIFAR-10. **Left**: Test accuracy on validation set. For decentralized training, the accuracy is evaluated on the global average model. **Right**: Training losses (evaluated on the workers for decentralized training, and evaluated on perturbed points for SAM) and validation losses (evaluated on the global average model for decentralized training). The curves for each algorithm are based on 3 runs with the same set of random seeds.

6.3. Training Curves of Transformer Experiments

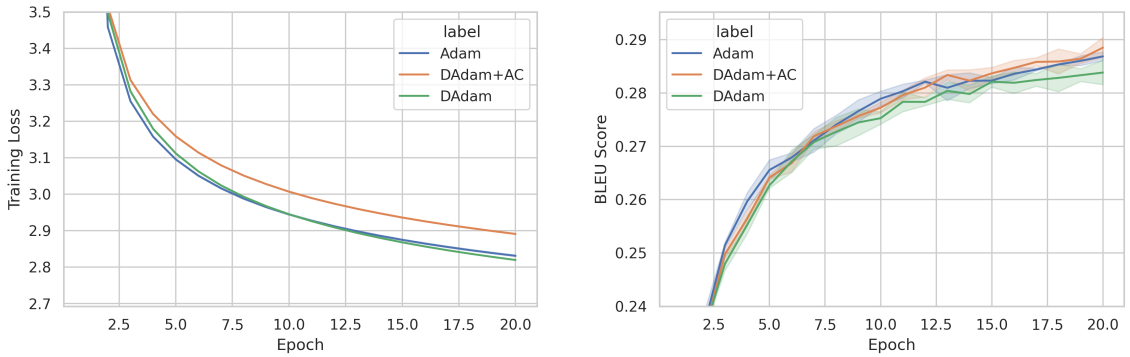

Figure 5: Transformer (big) on WMT14 English-to-German. **Left**: Losses on training set. Right: BLEU scores on the test set.

6.4. Losses on Local Datasets

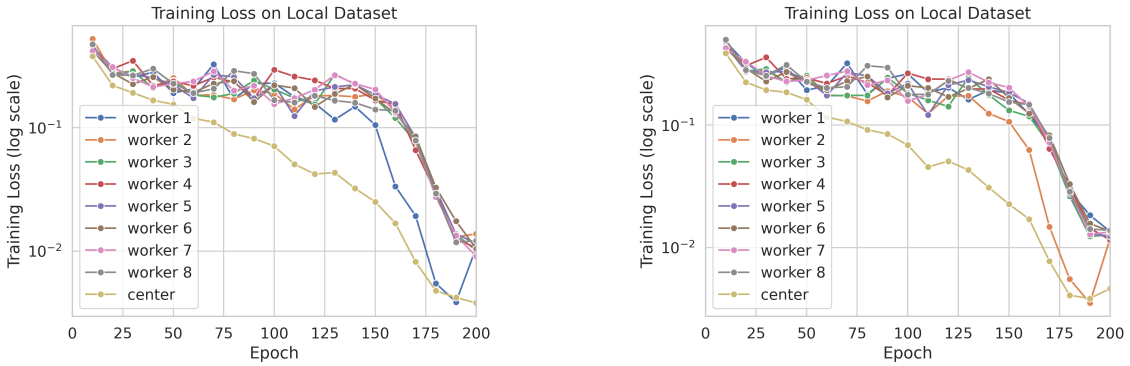

Figure 6: **Left**: Loss on the local dataset of worker 1. **Right**: Loss on the local dataset of worker 2.

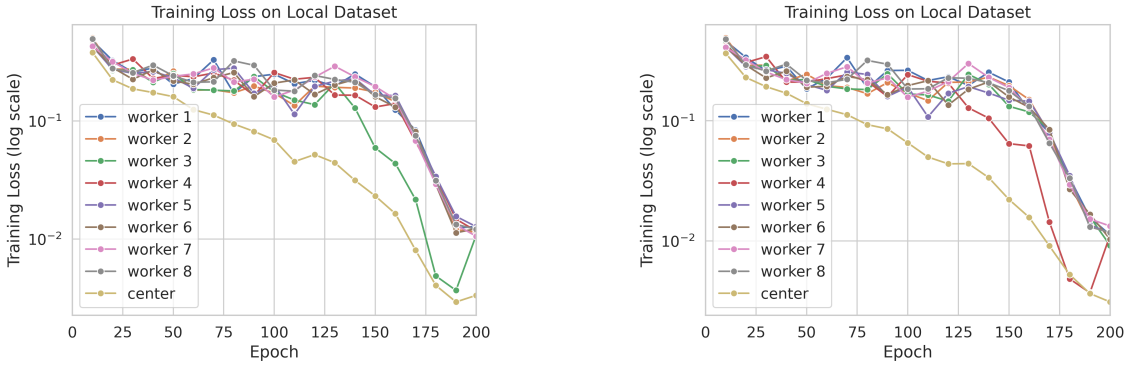

Figure 7: **Left**: Loss on the local dataset of worker 3. **Right**: Loss on the local dataset of worker 4.

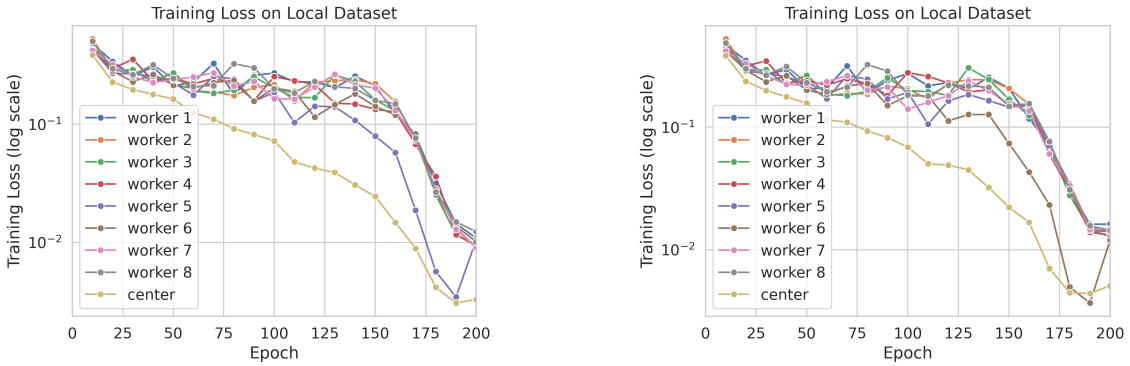

Figure 8: **Left**: Loss on the local dataset of worker 5. **Right**: Loss on the local dataset of worker 6.

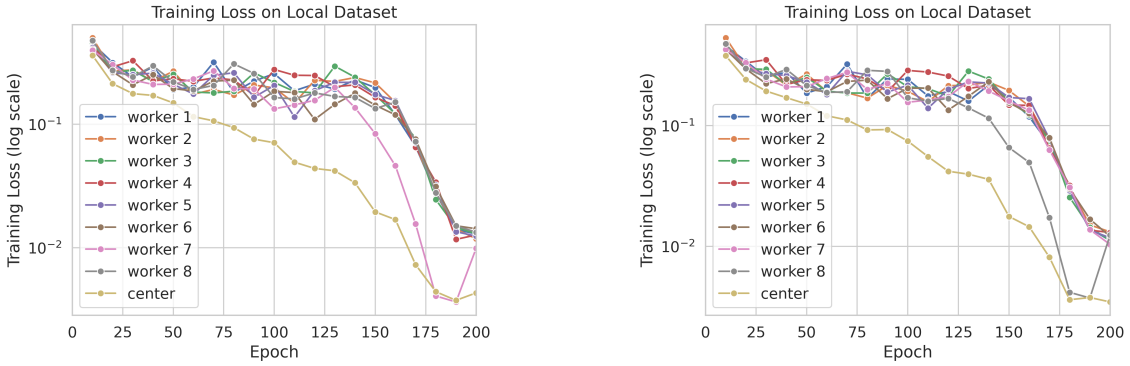

Figure 9: **Left**: Loss on the local dataset of worker 7. **Right**: Loss on the local dataset of worker 8.