Making Batch Normalization Great in Federated Deep Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Batch Normalization (BN) is commonly used in modern deep foundation models to improve stability and speed up convergence in centralized training. In federated learning (FL) with non-IID decentralized data, previous works observed that training with BN could hinder performance due to the mismatch of the BN statistics between training and testing. Group Normalization (GN) is thus more often used in FL as an alternative to BN. In this paper, we identify a more fundamental issue of BN in FL that makes BN inferior even with high-frequency communication between clients and servers. We then propose a frustratingly simple treatment, which significantly improves BN and makes it outperform GN across a wide range of FL settings. Along with this study, we also reveal an unreasonable behavior of BN in FL. We find it quite robust in the low-frequency communication regime where FL is commonly believed to degrade drastically. We hope that our study could serve as a valuable reference for future practical usage and theoretical analysis in FL.

1 Introduction

Foundation models [1] are normally very deep neural networks (DNNs) trained via stochastic gradient descent (SGD). FEDAVG [2] is arguably the most widely used training algorithm in an FL setting. FEDAVG iterates between two steps: parallel local SGD at the clients, and global model aggregation at the server. In the extreme case where global aggregation takes place after every local SGD step, FEDAVG is very much equivalent to centralized SGD for training simple DNN models like multi-layer perception [3, 4, 5, 6, 7]. Of course, due to the communication costs in practice, it is unlikely for clients to communicate at such a high frequency. Many existing works have thus focused on how to train DNNs at a lower communication frequency (e.g., once after local SGD for a few epochs), especially under the challenging condition where the data across clients are non-IID [8, 9, 10, 11].

In this paper, we specifically focus on DNN models that contain Batch Normalization (BN) layers [12]. In centralized training, especially for deep feed-forward models like ResNet [13], BN has been widely used to improve the stability of training and speed up convergence. In the literature on FL, however, many of the previous experiments have focused on shallow ConvNets (CNN) without BN; only a few works have particularly studied the usage of BN in FL [14, 15]. In [14], the authors pointed out the mismatch between the feature statistics (i.e., the means and variances in BN) estimated on non-IID local data (during training) and global data (during testing), and argued that this cannot be addressed by using larger mini-batch sizes or other sampling strategies. Hsieh et al. [14] thus proposed to replace BN with Group Normalization (GN) [16] and showed its superior performance in some extreme non-IID settings. Such a solution has since been followed by a long non-exhaustive list of later works [17, 18, 19, 20, 21, 22, 23, 24].

With that being said, replacing BN with GN in FL is more like an ad hoc solution rather than a cure-all. First, in centralized training, BN typically outperforms GN empirically. Replacing BN with

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

GN in FL thus seems like a compromise. Second, several recent works [25, 26, 27, 11] have reported that BN is still better than GN in their specific FL settings. Third, changing the normalization layer may create a barrier between the communities of centralized learning and FL. To illustrate, in centralized training, many publicly available pre-trained checkpoints [28, 29] are based on popular CNN architectures even recent transformers [30] with BN; most understanding [31, 32, 33], empirical studies [34], and theoretical analysis [35] about normalization in DNNs are built upon BN rather than GN. These prior results may become hard to be referred to in the FL community.

Last but not least, after a careful study of recent works that reported poor performance of BN [36, 37, 38], we found that the huge gap between centralized learning and FL cannot be closed even if clients communicate right after *every* local SGD step. Such a finding sharply contradicts what is observed on DNNs without BN. In other words, the issue with applying BN in FL seems to be more fundamental than previously believed.

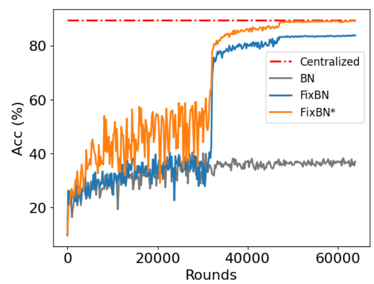


Figure 1: Our simple two-stage treatment FixBN largely bridges the gaps of using BN in FL and centralized learning. Please see section 3 and subsection C.4 for more details about FixBN (*: with SGD momentum) and this non-IID CIFAR-10 experiments.

Contributions. Building upon these aspects, we strive to answer the following questions towards a more holistic understanding of BN in FL, especially under non-IID settings.

1. Why does BN degrade so drastically in FL compared to centralized training or other normalizers? (section 3)
2. Is there a way to properly use BN in FL to bridge the performance gap w.r.t. centralized training? (section 4)
3. Is there a comfort zone and danger zone for BN (and other normalization methods) in FL? (Appendix B)

To begin with, we investigate several different perspectives to understand the issue of BN in FL, including BN statistic dynamics, the training/test mismatch of statistics, and the gradient w.r.t. the input of a BN layer under non-IID settings. Notably, we show that even if clients communicate after every local step, the *dependency of the gradient on the local mini-batch* prevents FedAvg from recovering the gradient computed in the centralized training setting. We note that this does not happen to DNNs with GN, as GN does not use mini-batch statistics to normalize features. Taking this insight into account, we propose a simple yet highly effective treatment named FixBN, which requires no architecture change, no additional training, and no extra communication costs.

2 Related Work

Normalization layers in centralized training. The benefits of BN [12] have been extensively studied in centralized training such as less internal covariate shift [12], smoother optimization landscape [32], robustness to hyperparameters [31] and initialization [35], accelerating convergence [12], etc. The noise of the estimated statistics of BN in mini-batch training is considered a regularizer [33] that improves generalization [12]. A recent study [39] shows that BN is still the irreplaceable normalizer vs. a wide range of alternative choices in general settings. Note that, unlike in FL, BN *often outperforms* GN *in standard centralized training.*

Existing use of normalizers in FL. In the context of FL, [14] is the first to suggest replacing BN with GN for non-IID decentralized learning. Several works [15, 40] report that LN can be competitive to GN. [41] enhances adversarial robustness by using statistics from reliable clients but not for improving performance. HeteroFL [42] proposed to simply normalize batch activations instead of tracking running statistics for the scenario that the clients have heterogeneous model architectures. These works aim to *replace* BN while *we analyze* BN *and reclaim its superiority*.

Several works [43, 44] propose dedicated server aggregation methods for BN statistics (separated from other model parameters) for specific tasks. For multi-modal learning, [45] proposes to maintain each modality as a different BN layer instead of sharing a single one. In personalized FL, [46, 47, 48] propose to maintain each client's independent BN layer, inspired by the practice of domain adaptation in centralized training [49]. [50] leverages BN statistics to guide aggregation for personalization. The goals of these works are orthogonal to ours.

3 Rethinking Batch Normalization in FL

3.1 Background

Batch Normalization (BN). The BN layer is widely used as a building block in feed-forward DNNs. Given an input feature vector $\boldsymbol{h}$, the BN layer normalizes the feature (via the mean $\boldsymbol{\mu}_{\mathcal{B}}$ and variance $\boldsymbol{\sigma}^2_{\mathcal{B}}$ computed on a batch of features $\mathcal{B}$), followed by a learnable affine transformation (via $\gamma, \boldsymbol{\beta}$): $\hat{\boldsymbol{h}} = f_{\text{BN}}(\boldsymbol{h}; (\boldsymbol{\gamma}, \boldsymbol{\beta}), (\boldsymbol{\mu}_{\mathcal{B}}, \boldsymbol{\sigma}^2_{\mathcal{B}})) = \gamma \frac{\boldsymbol{h} - \boldsymbol{\mu}_{\mathcal{B}}}{\sqrt{\boldsymbol{\sigma}^2_{\mathcal{B}} + \epsilon}} + \boldsymbol{\beta}$. ϵ is a small constant. In standard training, the statistics $\boldsymbol{\mu}_{\mathcal{B}}$ and $\boldsymbol{\sigma}^2_{\mathcal{B}}$ are computed on each training mini-batch during the forward passes. These mini-batch statistics are accumulated during training by the following exponential moving average (controlled by α) to replace $\boldsymbol{\mu}_{\mathcal{B}}$ and $\boldsymbol{\sigma}^2_{\mathcal{B}}$ in the equation above for testing:

$$\boldsymbol{\mu} := \alpha \boldsymbol{\mu} + (1 - \alpha) \boldsymbol{\mu}_{\mathcal{B}}, \quad \boldsymbol{\sigma}^2 := \alpha \boldsymbol{\sigma}^2 + (1 - \alpha) \boldsymbol{\sigma}^2_{\mathcal{B}}. \tag{1}$$

Federated averaging (FEDAVG). Federated learning (FL) learns a model parameterized by $\boldsymbol{\theta}$ on the decentralized data $\mathcal{D}_m, \forall m \in [M]$ of M clients. For DNNs with BN layers $\boldsymbol{\theta}$ includes learnable weights and the statistics $\{(\boldsymbol{\gamma}, \boldsymbol{\beta}), \boldsymbol{S}\}$ of all BN layers, where $\boldsymbol{S} = (\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$ are the BN means and variances. The fundamental FL algorithm FEDAVG [2] solves it by multiple rounds of parallel local updates at the clients and global model aggregation at the server. Given an initial model $\bar{\boldsymbol{\theta}}^{(0)}$, for round $t = 1, ..., T$, FEDAVG performs:

$$\textbf{Local: } \boldsymbol{\theta}_m^{(t)} = \texttt{ClientUpdate}(\mathcal{L}_m, \bar{\boldsymbol{\theta}}^{(t-1)}); \quad \textbf{Global: } \bar{\boldsymbol{\theta}}^{(t)} \leftarrow \sum_{m=1}^{M} \frac{|\mathcal{D}_m|}{|\mathcal{D}|} \boldsymbol{\theta}_m^{(t)}. \tag{2}$$

During local training, the clients update the model parameters received from the server, typically by minimizing each client's empirical risk $\mathcal{L}_m$ with several steps (denoted as E) of mini-batch SGD. For the locally accumulated means and variances in BN, they are updated by Equation 1. During global aggregation, all the parameters in the locally updated models $\{\boldsymbol{\theta}_m^{(t)}\}$, including the BN statistics, are averaged element-wise over clients. Typically, $E \gg 1$ due to communication constraints.

3.2 Problem: BN in FL cannot recover centralized performance

The non-IID issue is particularly problematic for DNNs with BN layers in FEDAVG since they depend on the activation mean and variance estimation computed on non-IID mini-batches.

We first consider communicating after every SGD step. That is, in the local training in Equation 2, we only perform a single mini-batch SGD update in each round, i.e., $E = 1$. At first glance, this should recover mini-batch SGD in centralized learning (e.g., training on multi-GPUs with local shuffling). However, as shown in Table 1 and Figure 1, even with high-frequency communication after every SGD step, there is a huge accuracy gap (about 45%) between centralized and federated learning for DNNs with BN. As a reference, *such a gap very much disappears for DNNs with* GN. Intrigued by this observation, we investigate the potential reasons from three aspects below, focusing on the non-IID FL setting.

Table 1: **FL with communication *every step*** ($E = 1$)**.** We train a ResNet20 with either BN or GN on the non-IID CIFAR-10 dataset (5 clients, 2 classes per client). Both the FL and centralized training use SGD without momentum.

Norm	Centralized Acc.	FL Acc.
GN	87.46±0.57	87.37±1.16
BN	89.30±0.89	42.93±2.75

3.3 BN training dynamics

We first consider the properties of BN in standard training. We note that BN normalizes the activations in the forward pass to ensure stable forward and backward propagation [39]. A naive workaround for the non-IID issue is to force all clients to normalize with the same statistics. We investigate this idea by "decoupling" the updates of the model weights and the BN statistics. Specifically, under the high-frequency communication setting with $E = 1$, we modify Equation 2 as follows. (a) At round t, given frozen weights in $\bar{\boldsymbol{\theta}}^{(t)}$, we update local statistics $\{\boldsymbol{S}_m^{(t+1)}\}_{m=1}^{M}$ via Equation 1 and aggregate them into $\bar{\boldsymbol{S}}^{(t+1)}$. (b) We then locally update the model weights in the evaluation mode, using the global statistics $\bar{\boldsymbol{S}}^{(t+1)}$ to normalize the activations. (c) Finally, we aggregate the local models into $\bar{\boldsymbol{\theta}}^{(t+1)}$. In the same FL experiment of subsection 3.2, we observe it achieves 52% accuracy, still far from the BN centralized performance 89%.

We hypothesize the weights and statistics need to collaborate carefully to enjoy the benefits of BN dynamics. First, using fixed statistics in local training sacrifices the "sampling" noise of the estimated statistics from different mini-batch $\boldsymbol{S}_{\mathcal{B}} = (\boldsymbol{\mu}_{\mathcal{B}}, \boldsymbol{\sigma}^2_{\mathcal{B}})$, which is believed to help search a flatter loss landscape [33]. Second, using fixed statistics cannot properly normalize the activations in a mini-batch and could make DNN training fragile due to gradient explosion, especially in the earlier rounds of FEDAVG when the model weights and intermediate activations are changing rapidly.

3.4 Re-examining the BN statistics mismatch between training and testing

The reason why BN degrades in FL is believed to be the *statistics mismatch* issue pointed out by [14]. In section 5 of [14], the authors argued that since the local accumulated statistics $\{\boldsymbol{S}_m = (\boldsymbol{\mu}_{\mathcal{D}_m}, \boldsymbol{\sigma}^2_{\mathcal{D}_m})\}$ are estimated on each of the non-IID local data $\{\mathcal{D}_m\}$, their average could be significantly different from the true statistics of the global data $\mathcal{D} = \cup_m \mathcal{D}_m$. In other words, the average of $\{\boldsymbol{S}_m\}$ (over m) may not be ideal in testing. To verify its impact on performance, we design a simple experiment (details in Appendix B) aiming to *remove* the statistics mismatch.

After the entire FEDAVG is finished, we re-accumulate the statistics $\{(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)\}$ per BN layer directly on the aggregated training data $\mathcal{D} = \cup_m \mathcal{D}_m$ over clients, using Equation 1. Interestingly, we see a fairly small gain, i.e., less than 1% accuracy gain even on extreme non-IID settings.

Based on the verification, we surmise that while the statistics mismatch problem indeed has a minor impact, it seems unlikely to account for the primary performance drops of BN *in FL.*

3.5 BN makes the gradients biased in local training

We hypothesize that under the non-IID settings, the major reason for the performance drop comes from BN's influence on local model training. As a simple illustration, we derive the forward-backward pass of the plain BN layer f_{BN} (see BN equation in subsection 3.1) for one example $\boldsymbol{x}_i$ in a mini-batch $\mathcal{B}$.

$$\textbf{Forward: } \ell(\hat{\boldsymbol{x}}_i) = \ell(f_{\text{BN}}(\boldsymbol{x}_i; (\boldsymbol{\gamma}, \boldsymbol{\beta}), (\boldsymbol{\mu}_{\mathcal{B}}, \boldsymbol{\sigma}^2_{\mathcal{B}}))) = \ell(\boldsymbol{\gamma} \frac{\boldsymbol{x}_i - \boldsymbol{\mu}_{\mathcal{B}}}{\sqrt{\boldsymbol{\sigma}^2_{\mathcal{B}} + \epsilon}} + \boldsymbol{\beta}) = \ell(\boldsymbol{\gamma}\tilde{\boldsymbol{x}}_i + \boldsymbol{\beta}); \quad (3)$$

$$\textbf{Backward: } \frac{\partial \ell}{\partial \boldsymbol{x}_i} = \frac{|\mathcal{B}| \frac{\partial \ell}{\partial \tilde{\boldsymbol{x}}_i} - \sum_{j=1}^{|\mathcal{B}|} \frac{\partial \ell}{\partial \tilde{\boldsymbol{x}}_j} - \tilde{\boldsymbol{x}}_i \cdot \sum_{j=1}^{|\mathcal{B}|} \frac{\partial \ell}{\partial \tilde{\boldsymbol{x}}_j} \cdot \tilde{\boldsymbol{x}}_j}{|\mathcal{B}| \sqrt{\boldsymbol{\sigma}^2_{\mathcal{B}} + \epsilon}}, \quad (4)$$

where ℓ is an arbitrary loss function on the BN layer's output $\hat{\boldsymbol{x}}_i$, "$\cdot$" is element-wise multiplication, and $\frac{\partial \ell}{\partial \tilde{\boldsymbol{x}}} = \boldsymbol{\gamma} \frac{\partial \ell}{\partial \hat{\boldsymbol{x}}}$. Please see Section 3 of [12] for the derivation of the gradient.

We can see that many terms in Equation 4 (colored in red) depend on the mini-batch features $\{\boldsymbol{x}_j\}_{j=1}^{|\mathcal{B}|}$ or statistics $(\boldsymbol{\mu}_{\mathcal{B}}, \boldsymbol{\sigma}^2_{\mathcal{B}})$. The background gradient $\frac{\partial \ell}{\partial \boldsymbol{x}_i}$ w.r.t. the input vector $\boldsymbol{x}_i$ is thus sensitive to what other examples in the mini-batch are. This is particularly problematic in FL on DNNs when clients' data are non-IID. Suppose $\boldsymbol{x}_i$ belongs to client m, the gradient $\frac{\partial \ell}{\partial \boldsymbol{x}_i}$ will be different when it is calculated locally with other data sampled only from $\mathcal{D}_m$ and when it is calculated globally (in centralized training) with other data sampled from $\mathcal{D} = \cup_m \mathcal{D}_m$. Such bias will propagate to the latter layers in a DNN. Namely, even if communicating after every mini-match SGD step, *how a particular data example influences the DNN parameters is already different between FL and centralized training.*

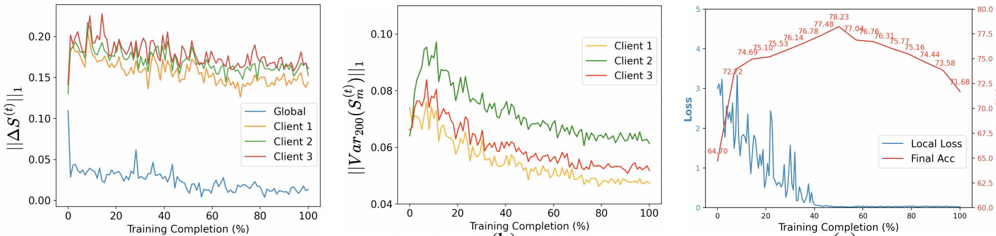
We surmise that this is the fundamental reason why DNNs with BN *degrade in FL. Although it becomes quite intuitive after our elaboration, to our surprise, such a gradient issue was not explicitly pointed out by previous works*[1]*. They mostly referred to the mismatch problem in [14].*

4 FIXBN: Towards a Proper Use of BN in Federated Learning

4.1 On *fixing* BN in FL

Given the analysis in section 3, we ask, *Is there a way to bypass the issues of* BN *in FL to reclaim its superior performance in centralized training?* We start our exploration by taking a deeper look into

[1]We recently noticed that a concurrent work [51] pointed out this finding as well. Nevertheless, our analysis and solution are quite different from theirs.



(a) (b) (c)

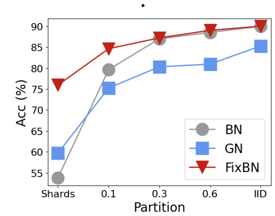
Figure 2: (a) Changes of global ($\|\bar{\boldsymbol{S}}^{(t+1)} - \bar{\boldsymbol{S}}^{(t)}\|_1$) and local mini-batch statistics ($\|\boldsymbol{S}_{m,\mathcal{B}}{}^{(t+1)} - \bar{\boldsymbol{S}}^{(t)}\|_1$). (b) Variances (running over $t-200$ to t) of local statistics $\boldsymbol{S}_m^{(t)}$. (c) Loss of global model on the training data and **final-round accuracy** when freezing BN statistics at different intermediate rounds (CIFAR-10, $E = 100$).

the dynamics of BN statistics during standard FEDAVG training. Under the same $E = 1$ experiments in subsection 3.2, we highlight two critical observations from Figure 2 (details in the captions).

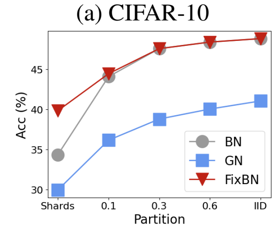
First, as shown in Figure 2 (a), the local mini-batch statistics remain largely different from the global statistics, even at later rounds, which results from the discrepancy between non-IID local data. This is not surprising. However, it re-emphasizes the potentially huge impact of the issue in subsection 3.5.

Second, still in Figure 2 (a), we look at each curve alone. We find that both the global and local mini-batch statistics essentially converge. We further show the variances of the local statistics within each client become static in Figure 2 (b). This opens up the possibility to revisit the "decoupling" attempt in subsection 3.3.

Concretely, if local mini-batch statistics remain almost static in later rounds, replacing them with the fixed global statistics in local training may not degrade the benefits of BN. In contrast, it may fundamentally resolve the issue in subsection 3.5 — using the fixed global statistics in Equation 4 could prevent local gradients from diverging. We investigate this idea by replacing local mini-batch statistics with fixed global statistics starting at different rounds. As shown in Figure 2 (c), if the round is chosen properly, the *final accuracy* can be largely improved. Based on this insight, we propose our FIXBN method to address the issues in section 3.



(a) CIFAR-10



(b) Tiny-ImageNet

Figure 3: Non-IID partitions with $E = 100$ steps.

4.2 Two-stage training

To address the drawbacks discussed in section 3 simultaneously, we propose to divide FEDAVG training with BN into two stages, separated at round $T^\star$, inspired by the widely-used decay learning strategy for SGD [52]. Supported by the insight in subsection 4.1, we first follow standard FEDAVG to explore a decent model solution space, thanks to BN's important training dynamics as studied in subsection 3.3. Next, we propose to fine-tune the model for the rest of the training with the BN statistics *fixed*. This eliminates the statistics mismatch problem in subsection 3.4 since now training and test share the same BN statistics. It also addresses the biased gradients caused by non-IID statistics in subsection 3.5 as the local gradients no longer rely on mini-batches.

Stage I: Exploration. This stage is the standard FEDAVG with BN for two purposes: (a) to explore a proper model subspace without sacrificing BN's benefits on optimization [12]; (b) to burn in the model and make it fitted to the training data. At the end, we save the aggregated statistics $\bar{\boldsymbol{S}}^{(T^\star)}$ of each BN layer from the average model $\bar{\boldsymbol{\theta}}^{(T^\star)}$.

Stage II: Calibration. We anticipate that the exploration stage already finds a proper region of the model solution, and calibrated fine-tuning is needed to further improve the performance. Starting from round $T^\star + 1$, we use the saved statistics as approximated global statistics to normalize the activations in local training. Since the model has been burned in, training with the fixed statistics is unlikely to suffer from the mentioned instability issue. In Figure 2 (c), we evaluate the training loss of the global model $\bar{\boldsymbol{\theta}}^{(T^\star)}$. It typically reaches a small loss after the first learning rate decay. Thus, *in experiments, we will simply fix the* BN *statistics since* 50% *of the total rounds of the FL training*.

While fairly simple, FIXBN cleverly leverages the global statistics to resolve the concerns in section 3, *with no architecture and objective change, no additional training and communication costs*.

Table 4: **Comparison to other FL normalizer methods.** Test accuracy (%) of ResNet20 on CIFAR-10 given # of rounds. The setting follows [51].

FL Scheme	#R	IID	Non-IID
Centralized+BN	-	91.53	
Centralized+FIXBN	-	91.62	
FEDTAN [51]	580K	91.26	87.66
FEDAVG +BN	10K	91.35	45.96
FEDAVG +GN	10K	91.26	82.66
HETEROFL [42]	10K	91.21	30.62
FEDDNA [43]	10K	91.42	76.01
FEDAVG +FIXBN (Ours)	10K	91.35	87.71

Table 5: ResNet20 with different normalization layers FL on CIFAR-10 (Shards, $E = 100$).

Normalization Layer	Acc (%)
BN [12]	53.97 $\pm$ 4.18
GN [16]	59.69 $\pm$ 0.76
GN +WN [60]	66.90 $\pm$ 0.81
LN [61]	54.54 $\pm$ 1.21
IN [62]	59.76 $\pm$ 0.43
FIXUP [63]	70.66 $\pm$ 0.24
FIXBN (Ours)	**76.56** $\pm$ 0.66

5 Experiments (more in the appendix)

Results on CIFAR-10 and Tiny-ImageNet. We experiment on the FL benchmark CIFAR-10 [53] and Tiny-ImageNet [54] with 5/10 clients, ResNet20/ResNet18 [13], respectively. For the hyperparameters, we generally follow [14] to use the SGD optimizer with 0.9 momentum, learning rate 0.02 (decay by 0.1 at 50% and 75% of the total rounds, respectively), batch size 20, and full participation, $E = 100$. We train **fixed 128 epochs** of total local SGD updates over all the clients and communication rounds. We consider **different non-IID degrees** including IID, Dirichlet(0.1, 0.3, 0.6) sampling follows [55], and Shards that each client only has data for 20% of the classes. We show FIXBN consistently outperforms BN and GN in Figure 3, especially in severe non-IID cases. Surprisingly, we found BN can sometimes outperform GN. A fine-grained comparison is in Appendix B.

Results on ImageNet. We extend FIXBN to ImageNet-1K [57] dataset which is split into 100 clients Dirichlet (0.1) non-IID over classes. We learn a ResNet18 with 10% randomly sampled clients per round, 20 batch size, 0.1 learning rate (decay by 0.1 every 30% of the total rounds), 2 local epochs, and 64 epochs in total. Results in Table 2 show that FIXBN also perform the best.

Table 2: Class-non-IID ImageNet.

Method	Network	Acc. $\Delta_{\text{-BN}}$
GN	ResNet18 [13]	33.33$\pm$ 0.57
BN		48.30$\pm$ 1.21
FIXBN		**52.43**$\pm$ 0.68 (+4.1)

Comparison on realistic non-IID Cityscape. We further consider a natural non-IID setting on the image segmentation dataset Cityscape [58]. We make each "city" a client and train 100 FEDAVG rounds using DeepLab-v3+ [59]. More details are in the appendix. Results in Table 3 show that FIXBN's effectiveness is generalized to different architectures and vision tasks.

Table 3: Pixel-wise accuracy and mean IoU (%) of image segmentation on Cityscapes.

Method	Backbone	Mean IoU $\Delta_{\text{-BN}}$
GN	MobileNet-v2 [56]	43.2$\pm$0.33
BN		48.9$\pm$0.36
FIXBN		**54.0**$\pm$0.29 (+5.1)
GN	ResNet18 [13]	47.8$\pm$0.30
BN		52.6$\pm$0.38
FIXBN		**57.2**$\pm$0.32 (+4.6)

Other FL baselines. We reproduce Table 1 in FEDTAN [51] to compare to other BN variants FL methods in Table 4. We note that FEDTAN requires communication rounds linear to the numbers of BN layers L as $\Theta(3L + 1)$, which is much more expensive than FIXBN. HETEROFL [42] directly normalizes the activations, which cannot resolve the non-IID issue.

Beyond BN & GN, is there any FL-friendly alternative? So far, we mainly focus on BN. Here we further compare to other normalization layers in Table 5. FIXBN still outperforms others. Interestingly, the normalization-free FIXUP [63] initialization for residual networks[2] performs much better than GN, suggesting a new alternative in FL besides FIXBN.

6 More Discussions and Conclusion

We revisit the use of BN layers and its common alternative, GN, in non-IID federated deep learning and conduct an in-depth analysis. We dissect the issues of BN in FL and propose a simple yet highly effective treatment named FIXBN to bridge the performance gap between FL and centralized training. We hope our study provides the community with a good foundation for the full (theoretical) understanding of the effectiveness of BN towards training deeper models in FL.

[2]Another concurrent work [64] also reports improving by replacing BN with scaled weight normalization, similar to [60] in Table 5.

References

[1] Bommasani, R., D. A. Hudson, E. Adeli, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[2] McMahan, H. B., E. Moore, D. Ramage, et al. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*. 2017.

[3] Zhou, F., G. Cong. On the convergence properties of a k-step averaging stochastic gradient descent algorithm for nonconvex optimization. *arXiv preprint arXiv:1708.01012*, 2017.

[4] Stich, S. U. Local sgd converges fast and communicates little. In *ICLR*. 2019.

[5] Haddadpour, F., M. Mahdavi. On the convergence of local descent methods in federated learning. *arXiv preprint arXiv:1910.14425*, 2019.

[6] Li, X., K. Huang, W. Yang, et al. On the convergence of fedavg on non-iid data. In *ICLR*. 2020.

[7] Zhao, Y., M. Li, L. Lai, et al. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

[8] Li, T., A. K. Sahu, M. Zaheer, et al. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450, 2020.

[9] Karimireddy, S. P., S. Kale, M. Mohri, et al. Scaffold: Stochastic controlled averaging for federated learning. In *ICML*. 2020.

[10] Acar, D. A. E., Y. Zhao, R. Matas, et al. Federated learning based on dynamic regularization. In *ICLR*. 2021.

[11] Chen, H.-Y., W.-L. Chao. Fedbe: Making bayesian model ensemble applicable to federated learning. In *ICLR*. 2021.

[12] Ioffe, S., C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[13] He, K., X. Zhang, S. Ren, et al. Deep residual learning for image recognition. In *CVPR*. 2016.

[14] Hsieh, K., A. Phanishayee, O. Mutlu, et al. The non-iid data quagmire of decentralized machine learning. In *International Conference on Machine Learning*, pages 4387–4398. PMLR, 2020.

[15] Du, Z., J. Sun, A. Li, et al. Rethinking normalization methods in federated learning. In *Proceedings of the 3rd International Workshop on Distributed Machine Learning*, pages 16–22. 2022.

[16] Wu, Y., K. He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19. 2018.

[17] Jin, J., J. Ren, Y. Zhou, et al. Accelerated federated learning with decoupled adaptive optimization. In *International Conference on Machine Learning*, pages 10298–10322. PMLR, 2022.

[18] Charles, Z., Z. Garrett, Z. Huo, et al. On large-cohort training for federated learning. *Advances in neural information processing systems*, 34:20461–20475, 2021.

[19] Lin, T., L. Kong, S. U. Stich, et al. Ensemble distillation for robust model fusion in federated learning. In *NeurIPS*. 2020.

[20] Yuan, H., W. Morningstar, L. Ning, et al. What do we mean by generalization in federated learning? In *ICLR*. 2021.

[21] Reddi, S., Z. Charles, M. Zaheer, et al. Adaptive federated optimization. In *ICLR*. 2020.

[22] Hyeon-Woo, N., M. Ye-Bin, T.-H. Oh. Fedpara: Low-rank hadamard product for communication-efficient federated learning. *arXiv preprint arXiv:2108.06098*, 2021.

[23] Yu, F., W. Zhang, Z. Qin, et al. Fed2: Feature-aligned federated learning. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 2066–2074. 2021.

[24] Hosseini, H., H. Park, S. Yun, et al. Federated learning of user verification models without sharing embeddings. In *International Conference on Machine Learning*, pages 4328–4336. PMLR, 2021.

[25] Mohamad, M., J. Neubert, J. S. Argayo. Fedos: using open-set learning to stabilize training in federated learning. *arXiv preprint arXiv:2208.11512*, 2022.

[26] Tenison, I., S. A. Sreeramadas, V. Mugunthan, et al. Gradient masked averaging for federated learning. *arXiv preprint arXiv:2201.11986*, 2022.

[27] Yang, S., H. Hwang, D. Kim, et al. Towards the practical utility of federated learning in the medical domain. *arXiv preprint arXiv:2207.03075*, 2022.

[28] PyTorch. Pytorch hub. `https://pytorch.org/hub/`, 2023.

[29] ONNX. Open neural network exchange (onnx) model zoo. `https://github.com/onnx/models`, 2023.

[30] Li, Y., G. Yuan, Y. Wen, et al. Efficientformer: Vision transformers at mobilenet speed. *Advances in Neural Information Processing Systems*, 35:12934–12949, 2022.

[31] Bjorck, N., C. P. Gomes, B. Selman, et al. Understanding batch normalization. *Advances in neural information processing systems*, 31, 2018.

[32] Santurkar, S., D. Tsipras, A. Ilyas, et al. How does batch normalization help optimization? *Advances in neural information processing systems*, 31, 2018.

[33] Luo, P., X. Wang, W. Shao, et al. Towards understanding regularization in batch normalization. In *ICLR*. 2019.

[34] Garbin, C., X. Zhu, O. Marques. Dropout vs. batch normalization: an empirical study of their impact to deep learning. *Multimedia Tools and Applications*, 79(19):12777–12815, 2020.

[35] Yang, G., J. Pennington, V. Rao, et al. A mean field theory of batch normalization. *arXiv preprint arXiv:1902.08129*, 2019.

[36] Wang, Y., Y. Xu, Q. Shi, et al. Quantized federated learning under transmission delay and outage constraints. *IEEE Journal on Selected Areas in Communications*, 40(1):323–341, 2021.

[37] Zheng, S., C. Shen, X. Chen. Design and analysis of uplink and downlink communications for federated learning. *IEEE Journal on Selected Areas in Communications*, 39(7):2150–2167, 2020.

[38] Chai, Z., Y. Chen, A. Anwar, et al. Fedat: a high-performance and communication-efficient federated learning system with asynchronous tiers. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. 2021.

[39] Lubana, E. S., R. Dick, H. Tanaka. Beyond batchnorm: towards a unified understanding of normalization in deep learning. *Advances in Neural Information Processing Systems*, 34:4778–4791, 2021.

[40] Zhang, G., M. Beitollahi, A. Bie, et al. Normalization is all you need: Understanding layer-normalized federated learning under extreme label shift. *arXiv preprint arXiv:2308.09565*, 2023.

[41] Hong, J., H. Wang, Z. Wang, et al. Federated robustness propagation: Sharing adversarial robustness in federated learning. *arXiv preprint arXiv:2106.10196*, 1, 2021.

[42] Diao, E., J. Ding, V. Tarokh. Heterofl: Computation and communication efficient federated learning for heterogeneous clients. In *ICLR*. 2020.

[43] Duan, J.-H., W. Li, S. Lu. Feddna: Federated learning with decoupled normalization-layer aggregation for non-iid data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 722–737. Springer, 2021.

[44] Idrissi, M. J., I. Berrada, G. Noubir. Fedbs: Learning on non-iid data in federated learning using batch normalization. In *2021 IEEE 33rd International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 861–867. IEEE, 2021.

[45] Bernecker, T., A. Peters, C. L. Schlett, et al. Fednorm: Modality-based normalization in federated learning for multi-modal liver segmentation. *arXiv preprint arXiv:2205.11096*, 2022.

[46] Li, X., M. JIANG, X. Zhang, et al. Fed{bn}: Federated learning on non-{iid} features via local batch normalization. In *ICLR*. 2021.

[47] Andreux, M., J. O. d. Terrail, C. Beguier, et al. Siloed federated learning for multi-centric histopathology datasets. In *Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning*, pages 129–139. Springer, 2020.

[48] Jiang, M., X. Zhang, M. Kamp, et al. Tsmobn: Interventional generalization for unseen clients in federated learning. *arXiv preprint arXiv:2110.09974*, 2021.

[49] Li, Y., N. Wang, J. Shi, et al. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*, 2016.

[50] Lu, W., J. Wang, Y. Chen, et al. Personalized federated learning with adaptive batchnorm for healthcare. *IEEE Transactions on Big Data*, 2022.

[51] Wang, Y., Q. Shi, T.-H. Chang. Why batch normalization damage federated learning on non-iid data? *arXiv preprint arXiv:2301.02982*, 2023.

[52] Robbins, H., S. Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[53] Krizhevsky, A., G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[54] Le, Y., X. Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 2015.

[55] Hsu, T.-M. H., H. Qi, M. Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

[56] Sandler, M., A. Howard, M. Zhu, et al. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520. 2018.

[57] Deng, J., W. Dong, R. Socher, et al. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. 2009.

[58] Cordts, M., M. Omran, S. Ramos, et al. The cityscapes dataset. In *CVPR Workshop on The Future of Datasets in Vision*. 2015.

[59] Chen, L.-C., Y. Zhu, G. Papandreou, et al. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818. 2018.

[60] Qiao, S., H. Wang, C. Liu, et al. Micro-batch training with batch-channel normalization and weight standardization. *arXiv preprint arXiv:1903.10520*, 2019.

[61] Ba, J. L., J. R. Kiros, G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[62] Ulyanov, D., A. Vedaldi, V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[63] Zhang, H., Y. N. Dauphin, T. Ma. Fixup initialization: Residual learning without normalization. *arXiv preprint arXiv:1901.09321*, 2019.

[64] Zhuang, W., L. Lyu. Is normalization indispensable for multi-domain federated learning? *arXiv preprint arXiv:2306.05879*, 2023.

[65] LeCun, Y., L. Bottou, Y. Bengio, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[66] Karimireddy, S. P., M. Jaggi, S. Kale, et al. Mime: Mimicking centralized stochastic algorithms in federated learning. *arXiv preprint arXiv:2008.03606*, 2020.

Appendix

We provide details omitted in the main paper.

- Appendix A: details of experimental setups (cf. Appendix B and section 4 of the main paper).
- Appendix B: experimental results and analysis for BN vs GN (cf. section 5 and section 4 of the main paper).
- Appendix C: additional experimental results and analysis for BN parameters and other ablation studies (cf. Appendix B and section 4 of the main paper).

Table F: Summary of datasets and setups.

Dataset	Task	#Class	#Training	#Test/Valid	#Clients	Resolution	Networks
CIFAR-10	Classification	10	$50K$	$10K$	$5 \sim 100$	32^2	LeNet-CNN, ResNet-20
Tiny-ImageNet	Classification	200	$100K$	$10K$	10	64^2	ResNet-18
ImageNet	Classification	1,000	$1,200K$	$100K$	100	224^2	ResNet-18
Cityscapes	Segmentation	19	$3K$	$0.5K$	18	768^2	DeepLabv3 + {MobileNet-v2, ResNet-50}

Table G: Default FL settings and training hyperparameters in the main paper.

Dataset	Non-IID	Sampling	Optimizer	Learning rate	Batch size	$T^\star$ for FIXBN
CIFAR-10	Shards, Dirichlet({0.1, 0.3, 0.6}), IID	$10 \sim 100\%$	SGD + 0.9 momentum	0.2/0.02	20	50% of total rounds
Tiny-ImageNet	Shards, Dirichlet({0.1, 0.3, 0.6}), IID	50%	SGD + 0.9 momentum	0.02	20	50% of total rounds
ImageNet	Dirichlet 0.1	10%	SGD + 0.9 momentum	0.1	20	50% of total rounds
Cityscapes	Cities	50%	Adam	0.01/0.001	8	90th round

A Experiment Details

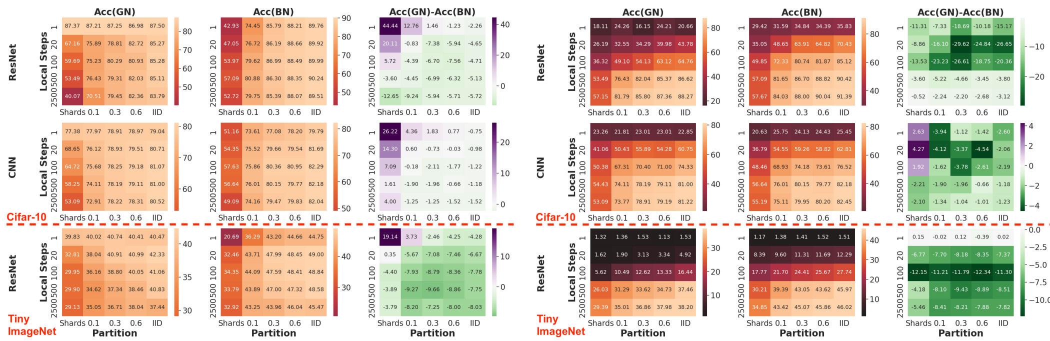
A.1 Datasets, FL settings, and hyperparameters

We use FEDAVG for our studies, with weight decay $1\mathrm{e}{-4}$ for local training. Learning rates are decayed by 0.1 at $50\%, 75\%$ of the total rounds, respectively. Besides that, we summarize the training hyperparameters for each of the federated experiments included in the main paper in Table G. Additionally, for the Cityscape experiments in Table 3, we make each "city" a client and run 100 rounds, with local steps to be 5 epochs. More details about the datasets are provided in Table F.

For pre-processing, we generally follow the standard practice which normalizes the images and applies some augmentations. CIFAR-10 images are padded 2 pixels on each side, randomly flipped horizontally, and then randomly cropped back to 32×32. For Tiny-ImageNet, we simply randomly cropped to the desired sizes and flipped horizontally following the official PyTorch ImageNet training script. For the Cityscapes dataset, we use output stride 16. In training, the images are randomly cropped to 768×768 and resized to 2048×1024 in testing.

B A Detailed Study of BN vs. GN

Results in section 5 come to an unexpected finding: BN outperforms GN in many cases, contradicting the common belief that one should replace BN with GN in FL proposed in [14] and followed by



(a) Fixed 128 **epochs** (b) Fixed 128 **rounds**

Figure D: **Is GN always better than BN? No.** We compare their test accuracy in various FL settings on CIAFR-10 and Tiny-ImageNet, including different non-IID partitions and numbers of local steps E. Fixed budget of the total number of SGD updates (e.g., for CIFAR-10, 20 $E \times 5$ clients $\times 3200$ rounds $= 128$ epochs) or the number of total rounds (128 rounds) are given.

many works summarized in section 1 and section 2. To answer this question, we revisit the study in [14] (which considers mere one FL setting) and provide a detailed study to compare BN and GN by varying several critical factors in FL to have a more complete picture.

Experiment setup. We focus on CIFAR-10 [53] and Tiny-ImageNet [54] datasets, following the setup in section 5. We consider more factors like **(1) degrees of non-IID**, ordered in increasing skewness: **IID, Dirichlet(0.1, 0.3, 0.6), and Shards**. As practical FL is constrained on computation, we consider two **(2) budget criteria**: **fixed 128 epochs** of total local SGD updates over all the clients and communication rounds, and **fixed 128 rounds** of communication. In every round, each client runs $\{1, 20, 100, 500, 2500\}$ of **(3) local steps (E)**. We further include LeNet-like CNN [65] for CIFAR-10.

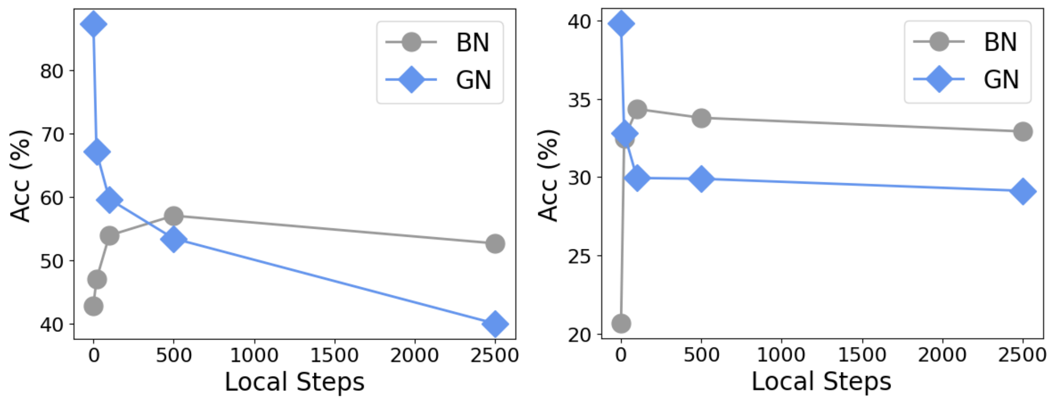
B.1 Resvisiting: Is GN really better than BN?

Observations. We highlight the following observation from Figure D, augmenting the findings in [14]:

- **No definite winners.** GN is often considered the default replacement for BN in previous FL works (section 1 and section 2). However, according to Figure D, GN is not always better than BN.
- **BN often outperforms GN.** Instead, in most settings, BN outperforms GN. This can be seen from the green cells in "Acc(GN)-Acc(BN)" heatmaps of Figure D.
- **GN outperforms BN merely in extreme cases.** We find that GN outperforms BN (the purple cells in "Acc(GN)-Acc(BN)" heatmaps) only in the extreme non-IID (e.g., Shards) and highly frequent communication (e.g., $E = 1$) settings. When clients cannot communicate frequently, the case where many existing FL works focus on, BN seems to be the better choice for normalization.
- **The trends along the number of local steps E per communication round.** It is a perhaps well-known fact that increasing the number of local steps leads to greater drift as the local models become more biased [9]. However, using more local steps also allows for more updates to the local models, potentially leading to an improved average model. To balance these competing considerations, we will discuss two criteria. For **(a) fixed epochs over all communication rounds**, a larger number of local steps means fewer communication rounds, in which GN degrades monotonically "as expected". *Interestingly,* BN *has an opposite trend.* BN actually improves and outperforms GN with larger Es. For **(b) fixed rounds**, understandably, using more local steps improves both GN and BN, since more local SGD updates are made in total. Nevertheless, the improvement saturates (e.g., $E \geq 500$).
- **Small difference from statistics mismatch.** In subsection 3.4, we discuss that the BN statistics mismatch problem might be minor. We re-estimate the statistics on global data and see a negligible accuracy gain from 44.09% to 44.87% on the Tiny-ImageNet (Dir(0.1), fixed epochs, $E = 100$).
- **More settings.** We verify in the next section that factors like participation rates and the number of clients for partitioning the data do not change the above observation.

Additional figures. At the beginning of this section, we provide a detailed empirical study to compare BN and GN across various FL settings to understand their sweet spots. Here we provide a closer look at the observations we summarized above.

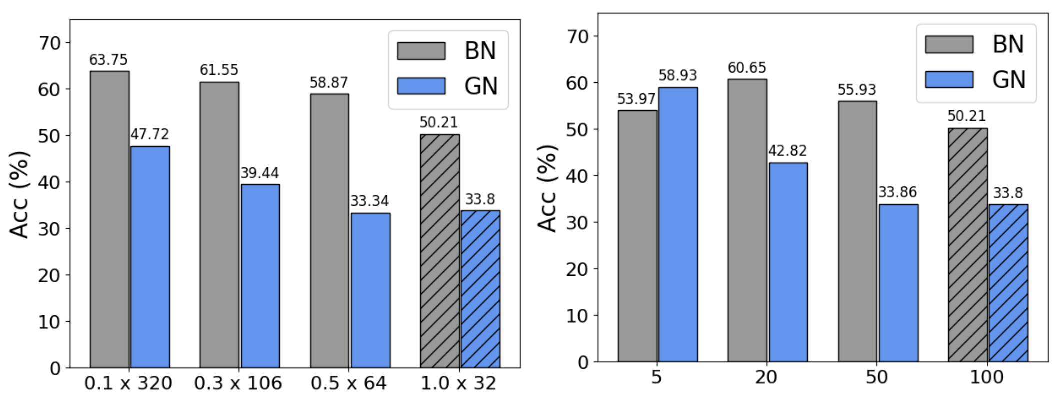
- **The trends along the number of local steps E per communication round.** In subsection B.2, we identify the opposite trends along #local steps E between BN and GN. As shown in Figure F, we see GN drops with less communication as expected due to the well-known non-IID model drift problem in FL. Interestingly, we found that BN can actually improve within a certain range of communication frequencies (for local steps in [1,500]), which suggests that further investigation and theoretical analysis are required for BN in FL.
- **More settings.** We further verify that factors such as participation rate and the number of clients for partitioning the data in Figure G. As expected, the results are consistent with the observations summarized in subsection B.1, particularly in that there is no definite winner between BN and GN while BN often outperforms GN.



(a) CIFAR-10

(b) Tiny-ImageNet

Figure F: **The opposite trends along #local steps E.** We consider the (Shards, **fixed epochs**) setting: the more the local step E is, the fewer the total number of communication rounds is. GN drops with less communication as expected, while BN can improve.



(a) Participation (%) $\times$ #Rounds

(b) Different #clients

Figure G: **More settings.** We consider more clients ($M = 5 \sim 100, E = 100$) for partitioning CIFAR-10 (Shards) with fixed epochs and varying the participation rate of clients every round.

B.2 Effects of communication frequency

The constraint in communication, i.e., clients cannot aggregate the gradients frequently, is commonly believed as a major reason that hinders the performance of FL due to model drift [9]. As BN cannot recover the centralized gradient even with high communication frequency and is outperformed by GN in such a setting, one may expect that BN will be consistently surpassed by GN when the frequency drops. But surprisingly, as observed in subsection B.1, BN *is unreasonably effective when training with fewer communication rounds but more local steps per round.*

In Figure E, we vary the number of local SGD steps per communication round (i.e., E) but fix the total number of SGD steps. We see the drastically different effect of E on BN and GN. In particular, while the performance of GN drops along with increasing E, BN somehow benefits from a larger E. Such a discrepancy suggests the need for a deeper (theoretical) analysis of the usage of BN in FL.

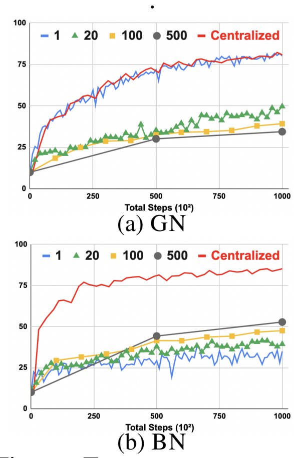


Figure E: Test accuracy on CIFAR-10 for different local steps (E) per communication given a fixed number of SGD steps.

C Additional experimental results and analysis

C.1 Additional study of fixing BN parameters

In subsection 3.4, we discuss that the BN statistics are the main critical parameters in FL and thus motivate our design in FIXBN to fix the BN statistics to be the global aggregated ones after certain rounds. Here we include a further study to confirm the importance of BN statistics by comparing them with the learnable affine transformation parameterized by $(\boldsymbol{\gamma}, \boldsymbol{\beta})$.

For FIXBN, besides fixing the BN statistics at round $T^\star$, we consider fixing the $(\boldsymbol{\gamma}, \boldsymbol{\beta})$ alone or together. The results on CIFAR-10 (Shards, fixed epochs, $E = 100$) setting using ResNet20 is in Table H. We observe that fixing the $(\boldsymbol{\gamma}, \boldsymbol{\beta})$ only has slight effects on the test accuracy either in combination with fixing $(\boldsymbol{\gamma}, \boldsymbol{\beta})$ or not, validating that the statistics are the main reason making it suffers more in FL, compared to the affine transformation. Fixing $(\boldsymbol{\gamma}, \boldsymbol{\beta})$ alone cannot match the performance of the originally proposed FIXBN.

Table H: **Fixing different parameters as in FIXBN.** We consider fixing the BN statistics $(\boldsymbol{\mu}, \boldsymbol{\sigma})$ as in original FIXBN or fixing the parameters $(\boldsymbol{\gamma}, \boldsymbol{\beta})$ of the affine transformation in BN layers. on CIFAR-10 (Shards, fixed epochs, $E = 100$) setting using ResNet20.

$(\boldsymbol{\mu}, \boldsymbol{\sigma})$	$(\boldsymbol{\gamma}, \boldsymbol{\beta})$	Acc (%)
✓	✓	75.22
✓	✗	76.56
✗	✓	55.33
✗	✗	53.97

C.2 Different # of groups for GN

For experiments in our study, we set the $\#$ of groups $= 2$ for GN layers. We did not find the group size a significant factor for the performance, as confirmed in Table I.

Table I: **Effects of the groupsize for GN.** We experiment with different # of groups ($2 \sim 8$) to divide the channels in GN layers in the CIFAR-10 (Shards, $E = 100$) with fixed epochs setting.

Groupsize	Acc(%)
2	59.42
4	57.61
8	58.86

C.3 Effects of Batch Size for BN

We experiment with various batch sizes for both BN and FIXBN in the CIFAR-10 (Shards, $E = 1$) setting and saw FIXBN maintains the advantage over standard FEDAVG +BN.

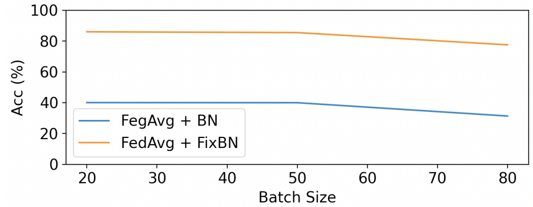
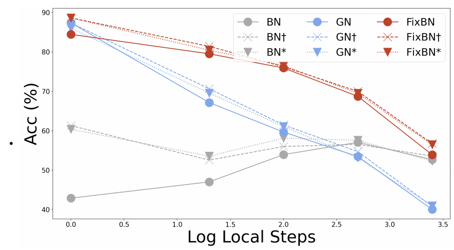


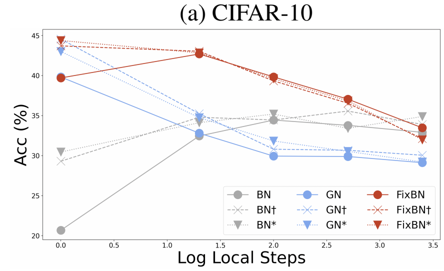
Figure H: FIXBN maintains advantage over different batch size selections.

C.4 Maintained SGD momentum further bridge FL to centralized performance

Maintained SGD momentum. Besides BN, we identify another gap between FEDAVG and centralized training. While using SGD momentum in standard FEDAVG during local training is common, it will be discarded at the end of the round and re-initialized (along with any optimizer states) at the beginning of the next round of local training in FEDAVG. That is, the first several SGD steps in a round cannot benefit from it.



(a) CIFAR-10



(b) Tiny-ImageNet

Figure I: **Maintained momentum.** Normalizers augmented with maintained **global momentum** (†) and **local momentum** (*) with different numbers of local steps per communication E.

To further bridge the gap, we present a fairly simple method, which is to keep the **local momentum** without re-initialization after the end of the local training in each round. This makes it a stateful method suitable for cross-silo FL. Another stateless choice is to maintain **global momentum** [66] by uploading the local momentum to the server in every round and aggregating it with Equation 2, for initializing the momentum of the next round of local training, with the cost of double message size. Empirically, we found the two methods yield similar gains (as will be shown in Figure I) and recover centralized performance if communicating every step (Figure 1).

Experimental Results. We combine each normalizer with the maintained local momentum and global momentum proposed in subsection C.4, respectively. We show FIXBN's effectiveness against BN and GN in Figure I in the (Shards, fixed epoch) setting with different numbers of local steps per communication E of $\{1, 20, 100, 500, 2500\}$. We see FIXBN performs consistently better. More importantly, FIXBN remains highly accurate in fast communication, unlike BN, confirming that it mitigates the deviation issue in subsection 3.5 well. The improvements of using maintained global/local momentum are similar, providing the flexibility of stateless/stateful use cases. More gains are at small E, supporting our motivation to fix the zero initialization issue of the momentum. Across different settings, we see FIXBN $\geq$ BN $>$ GN in performance, consistent with Figure 3.

Both of them improve BN notably, especially at small E, supporting our motivation to fix the zero initialization issue of the local SGD momentum to stabilize the gradients. Indeed, in Figure 1 with $E = 1$, we show FIXBN largely recovers centralized performance, making BN much more applicable in FL.

C.5 Training curves

We provide the training curves of FIXBN and other normalizers under various settings in fixed 128 epochs using ResNet20 in Figure J, Figure K, Figure L, and Figure M, corresponding to Appendix B.

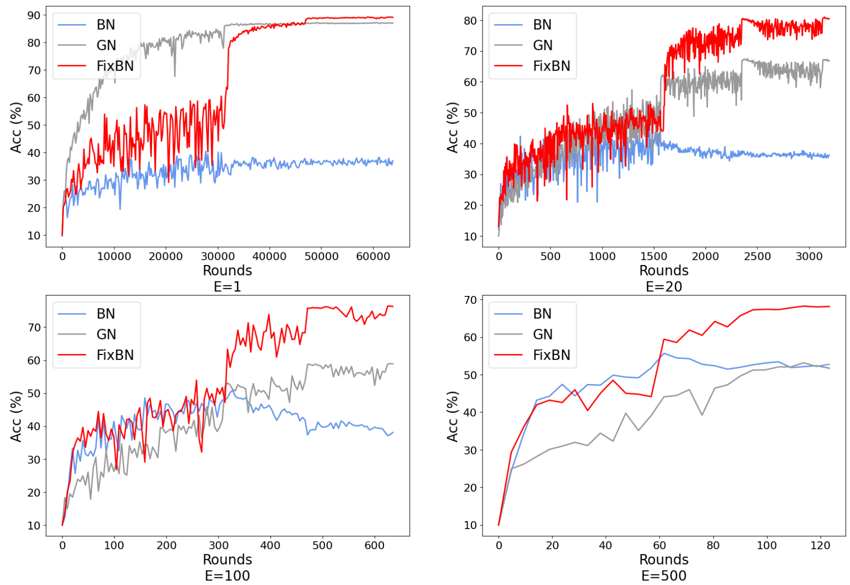


Figure J: Convergence curves of the test accuracy of CIFAR-10 with fixed epoch and **Shards non-IID** partitions, with $E = 1 \sim 500$.

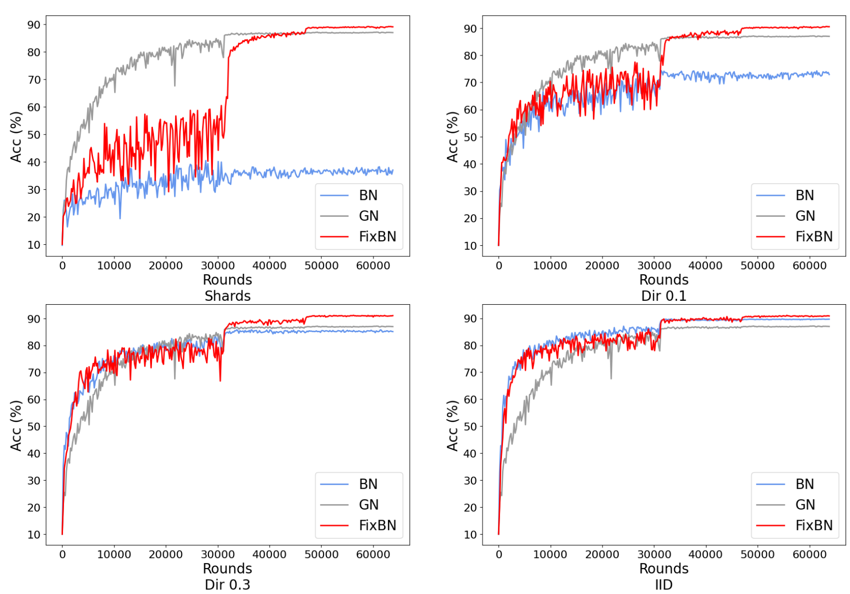


Figure K: Convergence curves of the test accuracy of CIFAR-10 in fixed epoch, **different non-IID partitions**, and $E = 1$ setting.

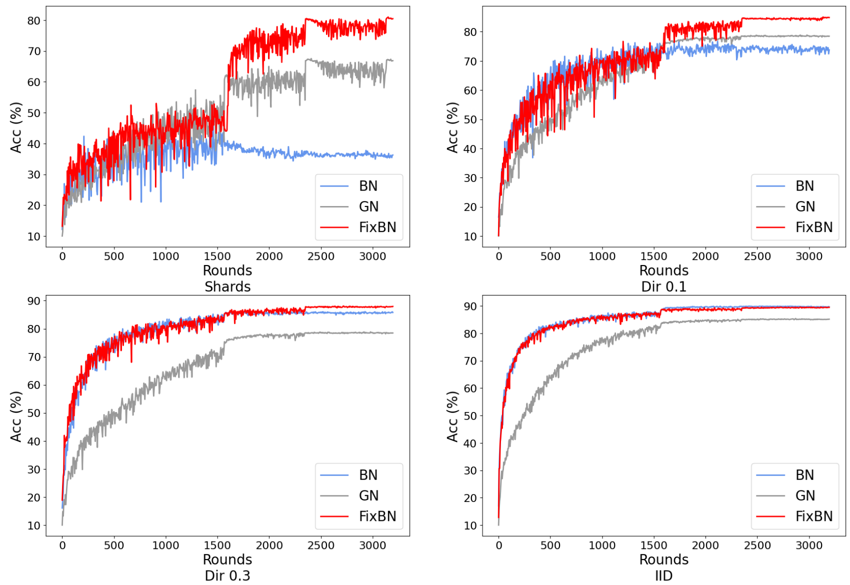


Figure L: Convergence curves of the test accuracy of CIFAR-10 in fixed epoch, **different non-IID partitions**, and $E = 20$ setting.

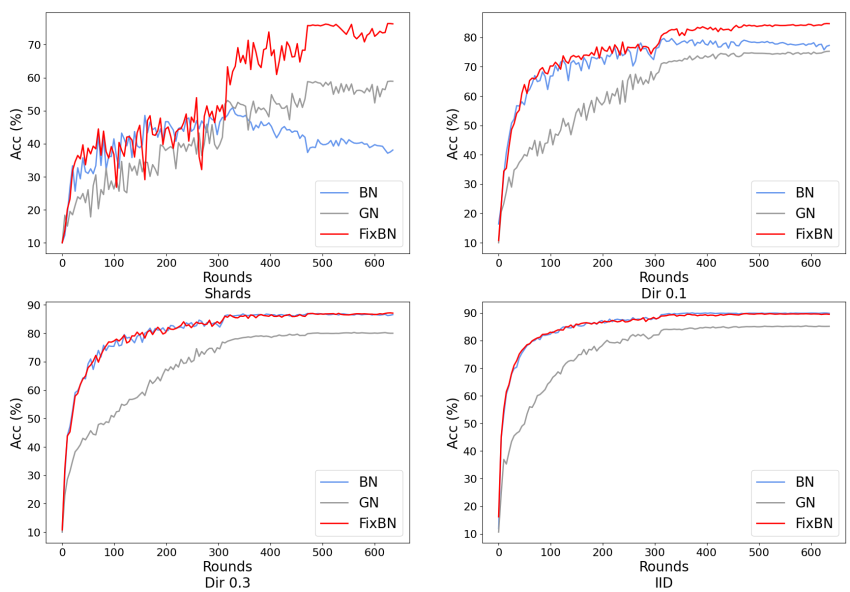


Figure M: Convergence curves of the test accuracy of CIFAR-10 in fixed epoch, **different non-IID partitions**, and $E = 100$ setting.