ResIST: Layer-Wise Decomposition of ResNets for Distributed Training

Abstract

We propose ResIST, a novel distributed training protocol for Residual Networks (ResNets). ResIST randomly decomposes a global ResNet into several shallow sub-ResNets that are trained independently in a distributed manner for several local iterations, before having their updates synchronized and aggregated into the global model. In the next round, new sub-ResNets are randomly generated and the process repeats until convergence. By construction, per iteration, ResIST communicates only a small portion of network parameters to each machine and never uses the full model during training. Thus, ResIST reduces the per-iteration communication, memory, and time requirements of ResNet training to only a fraction of the requirements of full-model training. In comparison to common protocols, like data-parallel training and data-parallel training with local SGD, ResIST yields a decrease in communication and compute requirements, while being competitive with respect to model performance.

1 INTRODUCTION

Background. The field of Computer Vision (CV) has seen a revolution, beginning with the introduction of AlexNet during the ILSVRC2012 competition. Following this initial application of deep convolutional neural networks (CNNs), the introduction of the residual connection (ResNets) allowed scaling to massive depths without being crippled by issues of unstable gradients during training [He et al., 2016b]. The capabilities of ResNets have been further expanded in recent years, but the basic ResNet architecture has remained widely-used. While ResNets have become a standard building block for the advancement of CV research, the computational requirements for training them are significant. For example, training a ResNet50 on ImageNet with a single NVIDIA M40 GPU takes 14 days [You et al., 2018].

Distributed training with multiple GPUs is commonly adopted to speed up the training process for ResNets. Yet, such acceleration is achieved at the cost of a remarkably large number of GPUs (e.g 256 NVIDIA Tesla P100 GPU in [Goyal et al., 2017]). Additionally, frequent synchronization and high communication costs create bottlenecks that hinder such methods from achieving speedups with respect to the number of available GPUs [Shi et al., 2018]. Asynchronous approaches avoid the cost of synchronization, but stale updates complicate their optimization process [Assran et al., 2020]. Other methods, such as data-parallel training with local SGD [Stich, 2019, Lin et al., 2018, Zhang et al., 2016, McMahan et al., 2017], reduce the frequency of synchronization. Similarly, model-parallel training has gained in popularity by decreasing the cost of local training between synchronization rounds [Ben-Nun and Hoefler, 2019, Zhu et al., 2020, Kirby et al., 2020, Gunther et al., 2020, Guan et al., 2019, Chen et al., 2018a].

This paper. We focus on efficient distributed training of CNNs with residual skip connections. Our proposed methodology accelerates synchronous, distributed training by leveraging ResNet robustness to layer removal [Huang et al., 2016]. In particular, a group of high-performing subnetworks (sub-ResNets) is created by partitioning the layers of a shared ResNet model to create multiple, shallower sub-ResNets. These sub-ResNets are then trained independently (in parallel) for several iterations before aggregating their updates into the global model and beginning the next iteration. Through the local, independent training of shallow sub-ResNets, this methodology both limits synchronization and communicates fewer parameters per synchronization cycle, thus drastically reducing communication overhead. We name this scheme *ResNet Independent Subnetwork Training* (ResIST). The contributions of this work are:

- We propose a distributed training scheme for ResNets, dubbed ResIST, that partitions the layers of a global model to multiple, shallow sub-ResNets, which are then

Submitted to the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022). **To be used for reviewing only**.

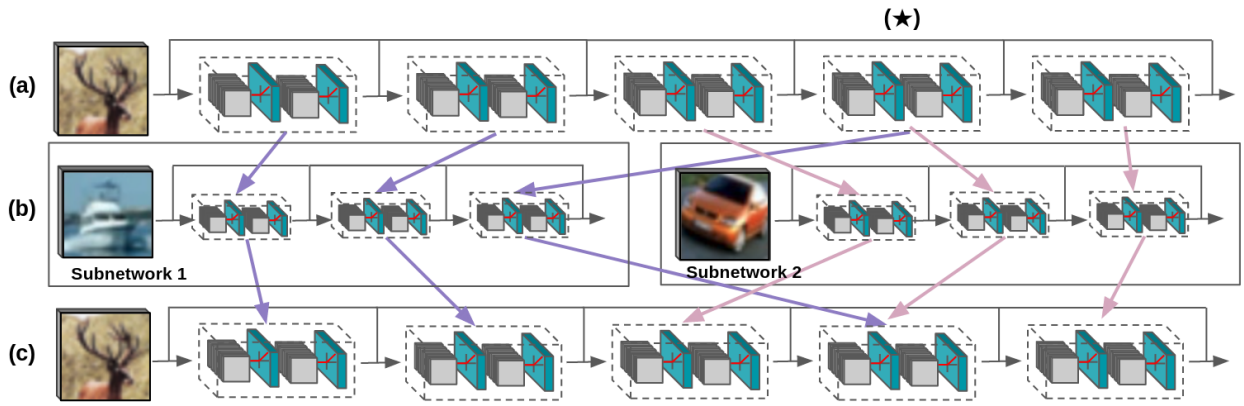

Figure 1: The ResIST model: **Row** (**a**) represents the original global ResNet. **Row** (**b**) shows the creation of two sub-ResNets. Observe that subnetwork 1 contains the residual blocks #1, #2 and #4, while subnetwork 2 contains the residual blocks #3, #4 and #5. **Row** (**c**) shows the reassembly of the global ResNet, after locally training subnetworks 1 and 2 for some number of local SGD iterations; residual blocks that are common across subnetworks (e.g., residual block #4, marked with a $\star$) are aggregated appropriately during the reassembly.

trained independently between synchronization rounds.

- We provide theory that ResIST (based on simple ResNet architectures) converges linearly, up to an error neighborhood, using distributed gradient descent with local iterations. We show that the behavior of ResIST is controlled by the overparameterization parameter m, as well as the number of workers S in the distributed setting, the number of local iterations, as well as the depth H of the ResNet architecture. Such findings reflect practical observations that are made in the experimental section.
- We perform extensive ablation experiments to motivate the design choices for ResIST, indicating that optimal performance is achieved by $i)$ using pre-activation ResNets, $ii)$ scaling intermediate activations of the global network at inference time, $iii)$ sharing layers between sub-ResNets that are sensitive to pruning, and $iv)$ imposing a minimum depth on sub-ResNets during training.
- ResIST is shown to achieve high accuracy and time efficiency in all cases. We conduct experiments on several image classification and object detection datasets, including CIFAR10/100, ImageNet, and PascalVOC.
- We utilize ResIST to train numerous different ResNet architectures (e.g., ResNet101, ResNet152, and ResNet200) and provide implementations for each in PyTorch [Paszke et al., 2019].

2 SUB-RESNET TRAINING

ResIST operates by partitioning the layers of a global ResNet to different, shallower sub-ResNets, training those independently, and intermittently aggregating their updates into the global model. The high-level process followed by ResIST is depicted in Fig. 1 and outlined in more detail by Algorithm 1. *We note that a naive, uniform partitioning of blocks to each subnetwork, resembling a distributed implementation of [Huang et al., 2016], performs poorly (see Fig. 6).* To improve upon this procedure, extensive design choices, outlined in Section A.1 in the Appendix, are studied to motivate ResIST, leading to a final methodology that generalizes well across domains and datasets.

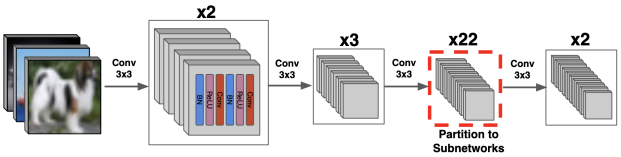

Figure 2: The ResNet101 model used in the majority of experiments. The figure identifies the convolutional blocks that are partitioned to subnetworks. The plot depicts the pre-activation ResNet setting, where we use BN, ReLU, and Conv layers twice in sequence. The network is comprised of four major "sections", each containing a certain number of convolutional blocks of equal channel dimension.

2.1 MODEL ARCHITECTURE

To achieve optimal performance with ResIST, the global model must be sufficiently deep. Otherwise, sub-ResNets may become too shallow after partitioning, leading to poor performance. For most experiments, a ResNet101 architecture is selected, which balances sufficient depth with reasonable computational complexity. Experiments with deeper architectures are provided in Section A.4 in the Appendix.

ResIST performs best with pre-activation ResNets [He et al., 2016a]. Intuitively, applying batch normalization prior to the convolution ensures that the input distribution of remaining residual blocks will remain fixed, even when certain

layers are removed from the architecture. The Pre-activation ResNet101, which we utilize for the majority of experiments, is depicted in Fig. 2. This model, as well as deeper variants (e.g., ResNet152 and ResNet200), are readily available through deep learning packages like PyTorch [Paszke et al., 2019] and Tensorflow [Abadi et al., 2015].

2.2 SUB-RESNET CONSTRUCTION

Pruning literature has shown that strided-, initial-, and final-layers within CNNs are sensitive to pruning [Li et al., 2017]. Additionally, repeated blocks of identical convolutions (i.e., equal channel size and spatial resolution) are less sensitive to pruning [Li et al., 2017]. Drawing upon these results, `ResIST` only partitions blocks within the third section of the ResNet (see the highlighted section in Fig. 2), while all other blocks are shared between sub-ResNets. These blocks are chosen for partitioning because $i)$ they account for the majority of layers; $ii)$ they are not strided; $iii)$ they are located within the middle of the network (i.e., initial/final layers are excluded); and $iv)$ they reside within a long chain of identical convolutions. By partitioning these blocks, `ResIST` allows sub-ResNets to be shallower than the global model, while maintaining high performance.

The process of constructing sub-ResNets follows a simple procedure; see Figure 1. From row (a) to (b) within Figure 1, indices of partitioned layers within the global model are randomly permuted and distributed to sub-ResNets in a round-robin fashion. Each sub-ResNet receives an equal number of convolutional blocks (e.g., see row (b)). In cases, residual blocks may be simultaneously partitioned to multiple sub-ResNets to ensure sufficient depth (e.g., see $(\star)$ in Figure 1). `ResIST` produces subnetworks with $\mathcal{O}(\frac{1}{S})$ of the global model depth, where S is the number of independently-trained sub-ResNets.[1] To contrast this with existing non-distributed attempts, stochastic depth networks [Huang et al., 2016] have an expected depth of 75% of the global model.

The shallow sub-ResNets created by `ResIST` accelerate training and reduce communication in comparison to methods that communicate and train the full model. Table 1 shows the comparison of local SGD to `ResIST` with respect to the amount of data communicated during each synchronization round for different numbers of machines, highlighting the superior communication-efficiency of `ResIST`.

2.3 DISTRIBUTED TRAINING

The `ResIST` training procedure is outlined in Algorithm 1. Sub-ResNet construction (i.e., $\texttt{subResNets}(\cdot)$ in Algorithm 1) follows the procedure outlined in Sec. 2.2. After constructing the sub-ResNets, they are trained independently in a distributed manner for ℓ iterations. Following independent training, the updates from each sub-ResNet are aggregated into the global model. Aggregation (i.e., $\texttt{aggregate}(\cdot)$ in Algorithm 1) sets each global network parameter to its average value across the sub-ResNets to which it was partitioned. If a parameter is only partitioned to a single sub-ResNet, aggregation simplifies to copying the parameter into the global model. After aggregation, the global model is re-partitioned randomly to create a new group of sub-ResNets, and this entire process is repeated.

[1] A fixed number of blocks is excluded from partitioning (i.e., blocks not in the third section). As a result, this approximation of $\mathcal{O}(\frac{1}{S})$ becomes more accurate as the network becomes deeper (i.e., deeper ResNet variants only add blocks to the third section), as a larger ratio of total blocks are included in the partitioning process.

Table 1: Data communicated during each communication round (in GB) of both local SGD [Stich, 2019] and `ResIST` across different numbers of machines with ResNet101.

Method	2 Machine	4 Machine	8 Machine
Local SGD	0.662 GB	1.325 GB	2.649 GB
`ResIST`	**0.454** GB	**0.720** GB	**1.289** GB

Algorithm 1 RESIST Meta Algorithm

Parameters: T synchronization iterations, S sub-ResNets, ℓ local iterations, $\mathcal{W}$ ResNet weights.
$h(W) \leftarrow$ randomly initialized ResNet.
for $t = 0, \ldots, T-1$ **do**
  $\{h_s(W_s)\}_{s=1}^{S} = \texttt{subResNets}(h(W),\ S)$.
  Distribute each $h_s(W_s)$ to a different worker.
  for $s = 1, \ldots, S$ **do**
    // Train $h_s(W_s)$ for ℓ iterations using local SGD.
    for $l_t = 1, \ldots, \ell$ **do**
      $W_s = W_s - \eta \frac{\partial L}{W_s}$
    end for
  end for
  $h(\mathcal{W}) = \texttt{aggregate}\left(\{h_s(W_s)\}_{s=1}^{S}\right)$.
end for

2.4 BASELINE CHOICE

Common baselines for distributed training are generally split into data- and model-parallelism protocols. Focusing on the former, the communication efficiency of `ResIST` significantly surpasses data-parallelism. In particular, data parallel methods need to synchronize the whole model at every training iteration, while `ResIST` only needs to communicate the weights of sub-ResNets among the workers.

Typically, model parallel techniques split the model into modules (such as layers) and distribute these modules to each worker. At every training iteration, input data is first passed to the worker containing the network's beginning module (e.g., the first layer). Then, at each module, the worker $i)$ performs a forward pass of its module and $ii)$

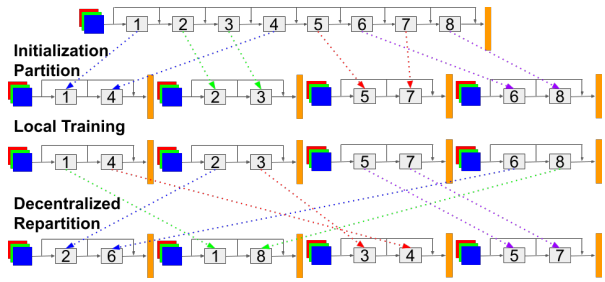

Figure 3: A depiction of the decentralized repartition procedure. This example partitions a ResNet with eight blocks into four different sub-ResNets. The "blue-green-red" squares dictate the data that lies per worker; the orange column dictates the last classification layer. As seen in the figure, each worker is responsible for only a fraction of parameters of the whole network. The whole ResNet is never fully stored, communicated or updated on a single worker.

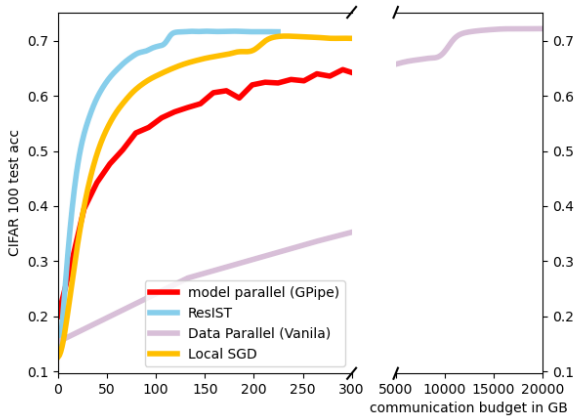

Figure 4: Communication efficiency of `ResIST` versus data parallelism (vanila), model parallelism (GPipe - [Huang et al., 2019]) and local SGD (LSGD) on CIFAR100.

sends the resulting output activation to the worker containing the next module. After the last module is activated, the final loss and gradient is calculated before the backward pass is performed, where each worker receives gradient information needed for updating module weights.

Model-parallelism often suffers from higher communication frequency and volume, in comparison to data parallel methods, due to the significant cost of transmitting network activation maps between workers. E.g., model parallel training of ResNets requires transmission of the full batch activation map between layers, which is more cumbersome than simply communicating network parameters. `ResIST` is more communication efficient compared to common model parallel methods (e.g., GPipe [Huang et al., 2019]).

Within this work, we adopt local SGD [Lin et al., 2018]—a strong variant of data parallel training—as our baseline. Similar to `ResIST`, local SGD performs local training iterations on each worker between synchronizations, thus largely decreasing communication frequency and volume. To justify this selection, we perform a baseline comparison, which is displayed in Figure 4 and further detailed in Sections 2.5 and 5. As shown in Figure 4, `ResIST` is significantly more communication efficient in comparison to data-parallelism (vanilla) and model-parallelism (GPipe), thus making local SGD a more appropriate baseline.

2.5 IMPLEMENTATION DETAILS

`ResIST` is implemented in PyTorch [Paszke et al., 2019], using the NCCL communication package. We use basic `broadcast` and `reduce` operations for communicating blocks in the third section and `all reduce` for blocks in other sections. We adopt the same communication procedure for the local SGD baseline to ensure fair comparison. *The implementation of `ResIST` is decentralized, meaning that it does not assume a single, central parameter server.*

As shown in Figure 3, during the synchronization and repartition step following local training, each sub-ResNet will directly send each of its locally-updated blocks to the designated new sub-ResNet. Each worker will only need sufficient memory to store a single sub-ResNet, thus limiting the memory requirements. Such a decentralized implementation allows parallel communication between sub-ResNets, which leads to further speedups by preventing any single machine from causing slow-downs due to communication bottlenecks. The implementation is easily scalable to eight or more machines, either on nodes with multiple GPUs or across distributed nodes with dedicated GPUs.

`ResIST` reduces the number of bits communicated at each synchronization round and accelerates local training with the use of shallow sub-ResNets. The authors are well-aware of many highly-optimized versions of data-parallel and synchronous training methodologies [Paszke et al., 2019, Abadi et al., 2015, Sergeev and Del Balso, 2018]. `ResIST` is fully compatible with these frameworks and can be further accelerated by leveraging highly-optimized distributed communication protocols at the systems level, which we leave as future work. Further, the authors are well-aware of advanced recent decentralized distributed computing techniques as in [Koloskova et al., 2020, Nedic and Ozdaglar, 2009, Assran and Rabbat, 2020, Koloskova et al., 2019]; our aim is to show the benefits of our approach even on simpler distributed frameworks, and we leave the extension of `ResIST` to such more advanced protocols as future work.

2.6 SUPPLEMENTAL TECHNIQUES

Scaling Activations. Similar to [Huang et al., 2016], activations must be scaled appropriately to account for the full depth of the resulting network at test time. To handle this, the output of residual blocks in the third section of the network (see Figure 2) is scaled by $1/S$, where S is the number

of sub-ResNets. Such scaling allows the global model to perform well, despite using all layers at test time.

Subnetwork Depth. Within `ResIST`, sub-ResNets may become too shallow as the number of sub-ResNets increases. To solve this issue, `ResIST` enforces a minimum depth requirement, which is satisfied by sharing certain blocks between multiple sub-ResNets. Through experimental analysis, a minimum of five blocks partitioned to each sub-ResNet was found to perform optimally. Such a finding motivates our choice of the ResNet101 architecture, as ResNet50 contains only five blocks for partitioning. `ResIST` is extensible to deeper architectures; see Section A.4 in the Appendix.

Tuning Local Iterations. We use a default value of $\ell = 50$, as $\ell < 50$ did not noticeably improve performance. In some cases, the performance of `ResIST` can be improved by tuning ℓ (see Figure 7). The optimal ℓ setting in `ResIST` is further explored in Section A.3 in the Appendix.

Local SGD Warm-up Phase. Directly applying `ResIST` may harm performance on some large-scale datasets (e.g., ImageNet). To resolve this, we perform a few epochs with data parallel local SGD before training the model with `ResIST`.[2] By simply pre-training a model for a few epochs with local SGD, the remainder of training is completed using `ResIST` without a significant performance decrease.

3 THEORETICAL RESULT

We provide proof that the gradient descent direction of combined updates from all sub-ResNets, during distributed local training, is close to the hypothetical gradient descent direction of the whole model as if trained centrally.

Theorem 3.1 (Convergence Rate of Gradient Descent for `ResIST`)**.** *Assume there are S workers, ℓ local and T global steps. Assume the depth of the whole ResNet is H. Assume for all data indices $i \in [n]$, the data input satisfies $\|\mathbf{x}_i\|_2 = 1$, the data output satisfies $|y_i| = O(1)$, and the number of hidden nodes per layer satisfies $m =$*

$$\Omega\left(\max\left\{\frac{n^4}{\lambda_{\min}^4\big(\mathbf{K}^{(H)}\big)H^6}, \frac{n^2}{\lambda_{\min}^2(\mathbf{K}^{(H)})H^2}, \frac{n}{\delta}, \frac{n^2\log\Big(\frac{Hn}{\delta}\Big)}{\lambda_{\min}^2\big(\mathbf{K}^{(H)}\big)}\right\}\right).$$

Set the step size $\eta = O\left(\frac{\lambda_{\min}\big(\mathbf{K}^{(H)}\big)H^2}{n^2\ell^2 S}\right)$ in gradient descent in local training iteration, and follow the procedure as in Algorithm 1. Let the squared-norm loss be $L(\theta(t)) := \frac{1}{2}\|\mathbf{y} - f(\theta(t))\|_2^2$, per t global synchronization round, $t = 1, 2, \ldots T$; here, $\mathbf{y}$ corresponds to the data "labels", and $\theta(t)$ and $f(\theta(t))$ represent the parameters and the output of the whole ResNet, respectively, after t-global rounds of `ResIST`. Here, θ includes weights $\mathbf{W}^{(h)}$ at depth h and the last layer's weights $\mathbf{a}$. Then, with probability at least $1 - \delta$ over the random initialization, we have:

$$L(\theta(t)) \leq \left(1 - \frac{\eta\ell\lambda_{\min}\big(\mathbf{K}^{(H)}\big)}{2}\right)^t \cdot L(\theta(0)).$$

First, some definitions; more details in the Appendix. Similar to [Du et al., 2019], $\mathbf{K}^{(H)} \in \mathbb{R}^{n\times n}$ is a fixed matrix that depends on the input data, neural network architecture and the activation but does not depend on neural network parameters. Next, we present our method of proving this global result on `ResIST`. Our proof technique is inspired by [Du et al., 2019]: Let the prediction of the network at some k-th iteration be $\mathbf{u}(k) = f(\theta(k))$.[3] We formulate the training dynamics as:

$$\mathbf{y} - \mathbf{u}(k+1) = (\mathbf{I} - \eta\mathbf{G}(k))(\mathbf{y} - \mathbf{u}(k)),$$

where $\mathbf{G}_{ij}(k) = \left\langle \frac{\partial u_i(k)}{\partial\theta(k)}, \frac{\partial u_j(k)}{\partial\theta(k)} \right\rangle =$

$$\sum_{h=1}^{H}\left\langle \frac{\partial u_i(k)}{\partial\mathbf{W}^{(h)}(k)}, \frac{\partial u_j(k)}{\partial\mathbf{W}^{(h)}(k)} \right\rangle + \left\langle \frac{\partial u_i(k)}{\partial\mathbf{a}(k)}, \frac{\partial u_j(k)}{\partial\mathbf{a}(k)} \right\rangle$$
$$\triangleq \sum_{h=1}^{H+1}\mathbf{G}_{ij}^{(h)}(k).$$

The proof in [Du et al., 2019] obeys the following ideas: when the width m of deep ResNet is sufficiently large, $\mathbf{G}^{(H)}(k)$ will be very close to $\mathbf{G}^{(H)}(0)$, and all of $\mathbf{G}^{(H)}(k)$'s will be close to the fixed population gram matrix $\mathbf{K}^{(H)}$. The exact definition of $\mathbf{K}^{(H)}$ for ResNet can be found in Section 6 of Du et al. [2019]. Further, $\lambda_{\min}(\mathbf{G}^{(H)}(0))$ is larger than 0. Thus, by standard matrix perturbation analysis, it is shown that $\lambda_{\min}(\mathbf{G}^{(H)}(0))$ is also strictly positive, which will result in linear convergence of deep ResNet.

Here, we further generalize such technique to distributed `ResIST` with layer dropout. The novelty of our proof is that we only conduct gradient descent on sub-ResNets assigned to each local worker. *There is no training iteration with the whole model: this includes the generation of random masks that "champion" parts of the whole ResNet model per worker.* Handling such constructions is the gist of this proof: We carefully analyze the convergence of each subnetwork during local training iterations ℓ, and prove the global convergence of the combined whole model throughout synchronization rounds t. The full proof is provided in Section B in the Appendix.

4 RELATED WORK

Following ResNet, most novel architectures continued to leverage residual connections, which became standard in most architectures. The ResNet architecture has been further modified. *This work focuses on the pre-activation ResNet*

[2] Activations of blocks within 3rd section are still scaled during local SGD pre-training to maintain consistency with `ResIST`.

[3] We use k to abstract the notion of an iteration in [Du et al., 2019]; in our case, a different analysis includes two different iteration indices, ℓ and t.

variant [He et al., 2016a], as it achieves high performance and is well-suited to layer-wise decomposition.

The focus of this study is on synchronous methods of distributed optimization, such as data parallel training, parallel SGD [Zinkevich et al., 2010], or local SGD [Stich, 2019]. Our methodology is also a variant of model-parallel training [Ben-Nun and Hoefler, 2019, Zhu et al., 2020, Kirby et al., 2020, Gunther et al., 2020, Guan et al., 2019, Chen et al., 2018a]. Many studies have explored possible techniques of synchronous, distributed optimization, yielding a wide number of viable variants [Lin et al., 2018].

To reduce communication costs in the distributed setting, both quantization [Alistarh et al., 2017, Tang et al., 2018] and sparsification [Aji and Heafield, 2017, Jiang and Agrawal, 2018, Wangni et al., 2018] methods have been explored. Similarly, other studies have achieved speedups through the use of low-precision arithmetic during training [Jia et al., 2018]. However, *this line of work is orthogonal to our proposal and can be easily combined with the provided methodology; see Section A.5 in the Appendix.*

Large batch training is used to amortize communication and increase throughput for distributed training [Goyal et al., 2017, You et al.]. The properties of large batch training have since been studied extensively [Akiba et al., 2017, You et al., 2017, 2018]. Large batches alter training dynamics, warranting the use of complex heuristics to maintain comparable performance [You et al., 2017]. Here, *we do not focus on the extension of* `ResIST` *to the large-batch training domain. Rather, we consider this as future work.*

ResNet robustness to layer removal was explored in [Huang et al., 2016], while [Veit et al., 2016] showed that ensembles of shallow ResNets can yield high performance. [Huang et al., 2016] uses shallow networks during training and scales activations so that all layers may be used for inference. However, our approach is distinct in numerous ways. Primarily, *our method partitions blocks in a stochastic, round-robin fashion, which explicitly prevents the exclusion of layers from training rounds and yields reduced subnetwork depth compared to [Huang et al., 2016].* Inspired by [Li et al., 2017], we also selectively partition residual blocks that are least sensitive to pruning, allowing other layers (i.e., 30% of total layers) to be shared between subnetworks. Unlike [Huang et al., 2016], we avoid partitioning strided layers, which are sensitive to pruning [Li et al., 2017]. Furthermore, our methodology, instead of proposing a form of regularization, focuses on utilizing independent training of shallow sub-ResNets for efficient, distributed training.

Our approach also relates to neural ODE literature. This research connects ResNets as a discrete approximation to a continuous transformation from input to output [Lu et al., 2018]. The neural ODE perspective has been studied both empirically [Chen et al., 2018b, Dupont et al., 2019, Lu et al., 2018] and theoretically [Lu et al., 2020, Thorpe and van Gennip, 2018]. *This provides justification to our approach, as removing ResNet layers can be viewed as approximating the same transformation with a coarser discretization.*

5 EXPERIMENTAL DETAILS

Hyperparameters are tuned using a holdout validation set and results are obtained using optimal hyperparameters from the validation set. All experiments are repeated for three trials, and the average performance is presented. We adopt local SGD as our baseline for synchronous, distributed training methods. *In all cases,* `ResIST` *achieves comparable performance to local SGD, while lowering the total wall-clock time of training.* We use AWS p3.8xlarge instances for experiments with two or four machines[4] and p3.16xlarge instances for experiments with eight machines. We use each GPU as a single worker that hosts a different sub-ResNet.

Small-Scale Image Classification. Models are trained with `ResIST` on CIFAR10 and CIFAR100 for image classification. We adopt standard data augmentation techniques during training and testing [He et al., 2016b]. We adopt a batch size of 128 for each worker. Training is conducted for 80 epochs for experiments with two machines and 160 epochs for experiments with four or eight machines. The recorded performance reflects the best test accuracy achieved throughout training, averaged across three trials. The total wall-clock training time is also reported for each experiment.

ImageNet Classification. Models are trained with `ResIST` on the 1,000-class ILSVRC2012 image classification dataset [Deng et al., 2009]. We adopt standard data augmentation techniques during training and testing, and use a batch size of 256 for each worker [He et al., 2016b]. Training is conducted for 90 epochs. We initialize the learning rate to 0.1 and decrease it $10\times$ at epochs 30 and 60. For all experiments, we set $\ell = 15$, adopt a minimum depth of 10 blocks for each sub-ResNet, and warm-up pre-training using local SGD. For both `ResIST` and baseline experiments, we utilize momentum restarts and aggregate batch statistics every 1300 synchronization rounds.

Object Detection. `ResIST` is tested in the object detection domain on the Pascal VOC dataset [Everingham et al., 2010]. Our model, inspired by the Yolo-v2 object detection model [Redmon and Farhadi, 2017], consists of a ResNet101 backbone followed by a detection layer (i.e., a 1×1 convolution that outputs anchor box predictions). The ResNet backbone of this model is similar to the classification model described in Sec. 2.1, but without the pre-activation structure. The model is trained for 100 epochs with an image dimension of 448×448 and batch size of 10. No data augmentation techniques are used. The learning rate is increased from 10^{-5} to 10^{-4} over the first 30 epochs, and decreased by

[4] In Section A.4-Appendix and for the Pacal VOC experiment with two machines, we use a cluster with eight V100 GPUs.

$10\times$ at epochs 60 and 90. Both Pascal VOC 2007 and 2012 training sets are used during training, and performance is evaluated on the Pascal VOC 2007 test set. We report the wall-clock training time and the best loss achieved on the test set throughout training. Experiments are conducted on two and four machines using both local SGD and `ResIST`.

Table 2: Test accuracy of baseline LocalSGD versus `ResIST` on small-scale image classification datasets.

	# Machines	CIFAR10	CIFAR100
Local SGD	2	92.36% $\pm$ 0.01	70.67% $\pm$ 0.03
	4	92.90% $\pm$ 0.06	71.51% $\pm$ 0.04
	8	92.00% $\pm$ 0.07	69.64% $\pm$ 0.05
`ResIST`	2	91.95% $\pm$ 0.32	70.06% $\pm$ 0.51
	4	92.35% $\pm$ 0.22	71.30% $\pm$ 0.20
	8	91.45% $\pm$ 0.30	70.26% $\pm$ 0.21

6 RESULTS

6.1 SMALL-SCALE IMAGE CLASSIFICATION

Accuracy. The test accuracy on small-scale image classification datasets is listed in Table 2. *`ResIST` achieves comparable test accuracy in all cases where the same number of machines are used.* `ResIST` outperforms localSGD on CIFAR100 experiments with eight machines. The performance of `ResIST` and local SGD are strikingly similar in terms of test accuracy. In fact, the performance gap between the two method does not exceed 1% in any experimental setting. Furthermore, `ResIST` performance remains stable as the number of sub-ResNets increases, allowing greater acceleration to be achieved without degraded performance (e.g., see CIFAR100 results in Table 2). Generally, using four sub-ResNets yields the best performance with `ResIST`.

Efficiency. In addition to achieving comparable test accuracy, `ResIST` significantly accelerates training. This acceleration is due to $i)$ fewer parameters being communicated between machines and $ii)$ locally-trained sub-ResNets being shallower than the global model. Wall-clock training times for four and eight machine experiments are presented in Tables 4. `ResIST` provides 3.58 to $3.81\times$ speedup in comparison to local SGD. For eight machine experiments, a significant speedup over four machine experiments is not observed due to the minimum depth requirement and a reduction in the number of local iterations to improve training stability. We conjecture that for cases with higher communication cost at each synchronization and a similar number of synchronizations, eight worker `ResIST` could lead to more significant speedups in comparison to the four worker case.

A visualization of the speedup provided by `ResIST` on the CIFAR10 and CIFAR100 datasets is illustrated in Fig. 5. Models trained with `ResIST` match the final accuracy of those trained with local SGD. Furthermore, increasing

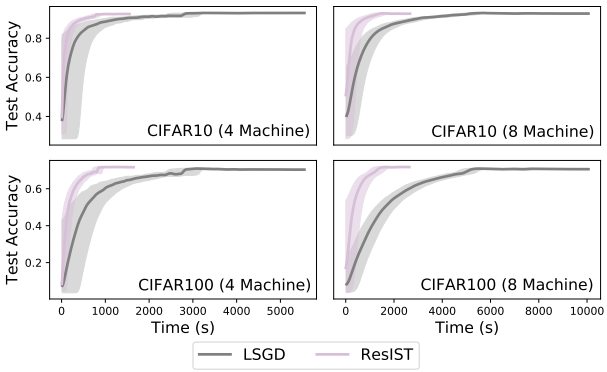

Figure 5: Both methodologies complete 160 epochs of training. Accuracy values are smoothed using a 1-D gaussian filter, and shaded regions represent deviations in accuracy.

the number of sub-ResNets yields an improved speedup for `ResIST` in comparison to localSGD. It is clear that the communication-efficiency of `ResIST` allows the benefit of more devices to be better realized in the distributed setting.

6.2 LARGE-SCALE IMAGE CLASSIFICATION

Accuracy. The test accuracy of models trained with both `ResIST` and local SGD for different numbers of machines on the ImageNet dataset is listed in Table 3. As can be seen, *`ResIST` achieves comparable test accuracy ($< 2\%$ difference) to local SGD in all cases.* Additionally, as shown in [Recht et al., 2019], many current image classification models overfit to the ImageNet test set and cannot generalize well to new data. Thus, models trained with both local SGD and `ResIST` are also evaluated on three different Imagenet V2 testing sets [Recht et al., 2019]. As shown in Table 3, `ResIST` consistently achieves comparable test accuracy in comparison to local SGD on these supplemental test sets.

Efficiency. As shown in Tables 3 and 5, `ResIST` significantly accelerates the ImageNet training process. However, due to the use of fewer local iterations and the local SGD warm-up phase, the speedup provided by `ResIST` is smaller relative to experiments on small-scale datasets. In Table 3, `ResIST` can reduce the total communication volume during training, which is an important feature in the implementation of distributed systems with high computational costs.

6.3 OBJECT DETECTION

Loss. The test loss of models trained with both `ResIST` and local SGD for different numbers of machines on the Pascal VOC object detection dataset is listed in Table 6. Notably, `ResIST` achieves a lower test loss in comparison to local SGD for the experiment with two machines. Although the test loss achieved by `ResIST` is slightly worse than local SGD in the four machine case, the performance

Table 3: Performance of baseline models and models trained with ResIST on 1K Imagenet [Recht et al., 2019]. MF stands for test set "MatchedFrequency" and was sampled to match the MTurk selection frequency distribution of the original ImageNet validation set; T-0.7 stands for test set "Threshold0.7" and was built by sampling ten images for each class among the candidates with selection frequency at least 0.7; TI stands for test set "TopImages" and contains the ten images with highest selection frequency for each class.

	# Machines	Imagenet	Imagenet V2 Test Set			Training Time	Speedup	Communication	Cost Ratio
			MF	T-0.7	TI				
Local SGD	2	73.32%	60.72%	69.47%	75.48%	48.61 hours	-	7546.80 GB	-
	4	72.66%	59.88%	68.34%	74.27%	29.29 hours	-	7546.80 GB	-
ResIST	2	71.60%	58.92%	67.51%	73.56%	36.79 hours	**1.32**×	5831.2 GB	**1.29**×
	4	70.74%	57.56%	66.46%	72.65%	22.37 hours	**1.31**×	6007.6 GB	**1.26**×

Table 4: Training time in seconds of baseline models and models trained with ResIST on small-scale image classification datasets.

	# Machines	Dataset	Total Time	Speedup
Local SGD	4	C10	5486 ± 7.05	-
		C100	5528 ± 65.90	-
	8	C10	10072 ± 5.12	-
		C100	10058 ± 8.71	-
ResIST	4	C10	1532 ± 0.83	**3.60**×
		C100	1545 ± 1.27	**3.58**×
	8	C10	2671 ± 3.25	**3.77**×
		C100	2639 ± 3.89	**3.81**×

Table 5: Total training time on Imagenet (in hours) of models trained with both local SGD and ResIST using two and four machines to reach a fixed test accuracy.

# Machines	Target Accuracy	Local SGD	ResIST	Speedup
2	71.00	33.26	26.63	**1.25**×
4	70.70	18.50	18.12	**1.02**×

is comparable. Namely, the difference in test loss achieved by local SGD and ResIST never exceeds a value of one.

Efficiency. In addition to achieving comparable or improved test loss in comparison to local SGD, ResIST also provides a significant training acceleration on the PascalVOC dataset. In particular, models trained with ResIST achieve up to a $1.64\times$ acceleration in comparison to object detection models trained with localSGD.

6.4 MORE EXPERIMENTS

In the Appendix A, we outline numerous ablation experiments that were performed using ResIST. These experiments provide an understanding of the algorithm's behavior, as well as empirical support for its design: they include ResIST design decisions (Section A.1), comparison of ResIST with ensemble methods (Section A.2), robustness to local iterations (Section A.3), applicability of ResIST to deeper architectures (Section A.4), and compatibility to existing quantization/sparsification techniques (Section A.5).

Table 6: Test loss and total training time on Pascal VOC for models trained with both local SGD and ResIST using two and four machines. Training time in seconds.

	# Mach.	Test Loss	Training Time	Speedup
Local SGD	2	6.15 ± 0.03	39621 ± 9.12	-
	4	6.22 ± 0.06	16840 ± 0.11	-
ResIST	2	5.99 ± 0.01	24058 ± 3.22	**1.64**×
	4	6.69 ± 0.17	11264 ± 49.38	**1.49**×

7 CONCLUSION

In the work, we present ResIST, a novel algorithm for synchronous, distributed training of ResNets. ResIST operates by decomposing a global ResNet model into several shallower sub-ResNets, which are trained independently and itermittently aggregated into the global model. By only communicating parameters of sub-ResNets between machines and training shallower, less expensive networks, ResIST reduces the communication and local training cost of synchronous, distributed training. We demonstrate the impact of ResIST on several image classification datasets, as well as in the object detection domain, by highlighting the significant training acceleration it provides in comparison to methods like local SGD [Lin et al., 2018] without any deterioration in performance.

We aim to extend ResIST to other network architectures, as ResIST is fully-extensible to all network architectures with residual connections. Because residual connections are now standard in most important deep learning architectures (e.g., transformers), many opportunities to extend applications of ResIST exist. On the other hand, ResIST has been shown to be fully-compatible with various gradient compression methods. As such, we will investigate the prospect of fully integrating such compression methods within ResIST, both during training and communication phases, to further decrease memory and computation costs.

References

Martín Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015.

Alham Fikri Aji and Kenneth Heafield. Sparse communication for distributed gradient descent. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 440–445, 2017.

Takuya Akiba, Shuji Suzuki, and Keisuke Fukuda. Extremely large minibatch SGD: Training ResNet-50 on imagenet in 15 minutes. *arXiv:1711.04325*, 2017.

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. QSGD: Communication-efficient SGD via gradient quantization and encoding. *Advances in Neural Information Processing Systems*, 30, 2017.

Mahmoud Assran, Arda Aytekin, Hamid Reza Feyzmahdavian, Mikael Johansson, and Michael G Rabbat. Advances in asynchronous parallel and distributed optimization. *Proceedings of the IEEE*, 108(11):2013–2031, 2020.

Mahmoud S Assran and Michael G Rabbat. Asynchronous gradient push. *IEEE Transactions on Automatic Control*, 66(1):168–183, 2020.

Tal Ben-Nun and Torsten Hoefler. Demystifying parallel and distributed deep learning: An in-depth concurrency analysis. *ACM Computing Surveys (CSUR)*, 52(4):1–43, 2019.

Chi-Chung Chen, Chia-Lin Yang, and Hsiang-Yun Cheng. Efficient and robust parallel dnn training through model parallelism on multi-gpu platform. *arXiv preprint arXiv:1809.02839*, 2018a.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018b.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

Simon Du, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. In *International conference on machine learning*, pages 1675–1685. PMLR, 2019.

Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural ODEs. In *Advances in Neural Information Processing Systems*, pages 3140–3150, 2019.

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

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:1706.02677*, 2017.

Lei Guan, Wotao Yin, Dongsheng Li, and Xicheng Lu. XPipe: Efficient pipeline model parallelism for multi-GPU DNN training. *arXiv preprint arXiv:1911.04610*, 2019.

Stefanie Gunther, Lars Ruthotto, Jacob B Schroder, Eric C Cyr, and Nicolas R Gauger. Layer-parallel training of deep residual neural networks. *SIAM Journal on Mathematics of Data Science*, 2(1):1–23, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016a.

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*, pages 770–778, 2016b.

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European conference on computer vision*, pages 646–661. Springer, 2016.

Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Dehao Chen, Mia Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V Le, Yonghui Wu, et al. Gpipe: Efficient training of giant neural networks using pipeline parallelism. *Advances in neural information processing systems*, 32, 2019.

Xianyan Jia et al. Highly scalable deep learning training system with mixed-precision: Training imagenet in four minutes. *arXiv preprint arXiv:1807.11205*, 2018.

Peng Jiang and Gagan Agrawal. A linear speedup analysis of distributed deep learning with sparse and quantized communication. *Advances in Neural Information Processing Systems*, 31, 2018.

Andrew Kirby, Siddharth Samsi, Michael Jones, Albert Reuther, Jeremy Kepner, and Vijay Gadepally. Layer-parallel training with gpu concurrency of deep residual neural networks via nonlinear multigrid. In *2020 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–7. IEEE, 2020.

Anastasia Koloskova, Sebastian Stich, and Martin Jaggi. Decentralized stochastic optimization and gossip algorithms with compressed communication. In *International Conference on Machine Learning*, pages 3478–3487. PMLR, 2019.

Anastasia Koloskova, Nicolas Loizou, Sadra Boreiri, Martin Jaggi, and Sebastian Stich. A unified theory of decentralized sgd with changing topology and local updates. In *International Conference on Machine Learning*, pages 5381–5393. PMLR, 2020.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *5th International Conference on Learning Representations, ICLR 2017*, 2017.

Tao Lin, Sebastian U. Stich, Kumar Kshitij Patel, and Martin Jaggi. Don't Use Large Mini-Batches, Use Local SGD. art. arXiv:1808.07217, August 2018.

Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. In *International Conference on Machine Learning*, pages 3276–3285. PMLR, 2018.

Yiping Lu, Chao Ma, Yulong Lu, Jianfeng Lu, and Lexing Ying. A mean field analysis of deep resnet and beyond: Towards provably optimization via overparameterization from depth. In *International Conference on Machine Learning*, pages 6426–6436. PMLR, 2020.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

Angelia Nedic and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.

Adam Paszke et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. 2019.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, pages 5389–5400. PMLR, 2019.

Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

Alexander Sergeev and Mike Del Balso. Horovod: fast and easy distributed deep learning in Tensorflow. *arXiv:1802.05799*, 2018.

Shaohuai Shi, Qiang Wang, and Xiaowen Chu. Performance modeling and evaluation of distributed deep learning frameworks on GPUs. In *2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing*, pages 949–957. IEEE, 2018.

Sebastian U. Stich. Local SGD converges fast and communicates little. In *International Conference on Learning Representations*, 2019.

Hanlin Tang, Shaoduo Gan, Ce Zhang, Tong Zhang, and Ji Liu. Communication compression for decentralized training. *Advances in Neural Information Processing Systems*, 31, 2018.

Matthew Thorpe and Yves van Gennip. Deep limits of residual neural networks. *arXiv:1810.11741*, 2018.

Andreas Veit, Michael J Wilber, and Serge Belongie. Residual networks behave like ensembles of relatively shallow networks. *Advances in neural information processing systems*, 29, 2016.

Jianqiao Wangni, Jialei Wang, Ji Liu, and Tong Zhang. Gradient sparsification for communication-efficient distributed optimization. *Advances in Neural Information Processing Systems*, 31, 2018.

Yang You, Jing Li, Sashank J. Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training BERT in 76 minutes. In *8th International Conference on Learning Representations, ICLR 2020*.

Yang You, Igor Gitman, and Boris Ginsburg. Scaling SGD batch size to 32K for Imagenet training. *arXiv preprint arXiv:1708.03888*, 6(12):6, 2017.

Yang You, Zhao Zhang, Cho-Jui Hsieh, James Demmel, and Kurt Keutzer. Imagenet training in minutes. In *Proceedings of the 47th International Conference on Parallel Processing*, pages 1–10, 2018.

Yue Yu, Jiaxiang Wu, and Longbo Huang. Double quantization for communication-efficient distributed optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

Jian Zhang, Christopher De Sa, Ioannis Mitliagkas, and Christopher Ré. Parallel SGD: When does averaging help? art. arXiv:1606.07365, June 2016.

Wentao Zhu, Can Zhao, Wenqi Li, Holger Roth, Ziyue Xu, and Daguang Xu. Lamp: Large deep nets with automated model parallelism for image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 374–384. Springer, 2020.

Martin Zinkevich, Markus Weimer, Lihong Li, and Alex J Smola. Parallelized stochastic gradient descent. In *Advances in neural information processing systems*, pages 2595–2603, 2010.

A ABLATIONS

These experiments provide an understanding of the algorithm's behavior, as well as empirical support for its design.

A.1 DESIGNING RESIST

Extensive ablation experiments are conducted on the CIFAR10 dataset, outlined in Fig. 6, to empirically motivate the design choices made within ResIST (i.e., see Sec. 2.6). For the two sub-ResNet case, the naive implementation of ResIST, which evenly splits all convolutional blocks between subnetworks, is shown to perform poorly (i.e., <70% on CIFAR10). The accuracy of ResIST is improved over 25% by only allowing select layers to be partitioned and ensuring activations are scaled correctly when performing inference with the full network. The pre-activation ResNet is shown to yield an improvement in accuracy, leading ResIST to perform near optimally with two sub-ResNets.

Modification	**Naive Model**						**ResIST**
Share strided layers		✓	✓	✓	✓	✓	✓
Only Partition Section 3			✓	✓	✓	✓	✓
Scale Activations				✓	✓	✓	✓
Pre-Act ResNet					✓	✓	✓
Minimum Depth						✓	✓
Tune Local Iterations							✓
2 Sub-ResNet Acc.	66.3%	68.3%	84.3%	91.5%	92.0%	-	92.0%
8 Sub-ResNet Acc.	-	-	-	-	86.5%	89.9%	91.3%

Figure 6: Test accuracies on the CIFAR10 dataset for a single run for the major ablation experiments performed with ResIST.

When ResIST is expanded to eight sub-ResNets, we initially observe a significant decrease in model accuracy. However, as can be seen in Fig. 6, this gap can be closed by enforcing a minimum depth on sub-ResNets and tuning the number of local iterations. By making these extra modifications, ResIST begins to perform similarly with two to eight sub-ResNets, yielding compelling performance.

A.2 SHALLOW ENSEMBLES

The ResIST algorithm requires that independently-trained sub-ResNets must have their parameters synchronized intermittently. Such synchronization, however, can be completely avoided by training each sub-ResNet separately and forming an ensemble (i.e., ResIST without any aggregation). Although maintaining an ensemble has several drawbacks (e.g., slower inference, more parameters, etc.), the training time of the ensemble would nonetheless be reduced in comparison to ResIST by avoiding communication altogether. Therefore, the performance of such an ensemble should be compared to the models trained with ResIST.

Table 7: Performance of indpendently-trained ensembles of shallow ResNets in comparison to ResIST on CIFAR10 and CIFAR100 (denoted as C10 and C100, respectively).

Dataset	Method	2 Model	4 Model	8 Model
C10	Ensemble	92.27 % $\pm$ 0.00	92.56% $\pm$ 0.03	90.67 % $\pm$ 0.04
	ResIST	91.95% $\pm$ 0.32	92.35% $\pm$ 0.22	91.45% $\pm$ 0.30
C100	Ensemble	72.08% $\pm$ 0.05	72.12% $\pm$ 0.04	67.98 % $\pm$ 0.12
	ResIST	70.06% $\pm$ 0.51	71.30% $\pm$ 0.20	70.26% $\pm$ 0.21

Table 8: Test accuracy on CIFAR10 (C10) and CIFAR100 (C100) for deeper architectures trained with ResIST and local SGD (LSGD). All tests were performed with 100 local iterations between synchronization rounds. All models were trained for 80 epochs.

			ResNet152			ResNet200		
Dataset	# Machines	Method	Time	Test Acc.	Speedup	Time	Test Acc.	Speedup
C10	2	LSGD	3512s	92.27% $\pm$ 0.003		4575s	92.31% $\pm$ 0.001	
		ResIST	2215s	92.01% $\pm$ 0.002	**1.58**$\times$	2380s	92.10% $\pm$ 0.001	**1.92**$\times$
	4	LSGD	3598s	91.39% $\pm$ 0.001		4357s	91.35% $\pm$ 0.000	
		ResIST	1054s	90.67% $\pm$ 0.001	**3.41**$\times$	1161s	90.27% $\pm$ 0.001	**3.75**$\times$
C100	2	LSGD	3528s	70.50% $\pm$ 0.003		4639s	71.05% $\pm$ 0.005	
		ResIST	2291s	70.32% $\pm$ 0.005	**1.53**$\times$	2202s	70.71% $\pm$ 0.002	**2.10**$\times$
	4	LSGD	3518s	68.39% $\pm$ 0.004		4391s	69.05% $\pm$ 0.003	
		ResIST	1164s	67.27% $\pm$ 0.003	**3.02**$\times$	1195s	67.62% $\pm$ 0.001	**3.67**$\times$

The performance of sub-ResNet ensembles in comparison to models trained with ResIST is displayed in Table 7. For 8 Sub-ResNets, the shallow ensembles achieve inferior performance in comparison to ResIST. When two and four Sub-ResNets are used, the performance of shallow ensembles and ResIST is comparable (i.e., $< 1\%$ performance difference in most cases). However, it should be noted that such shallow ensembles of two or four sub-ResNets, in comparison to ResIST, cause a $2\times$ to $4\times$ slowdown in inference time (i.e., inference time for a single Sub-ResNet is not significantly faster than that of the global ResNet). Furthermore, the ensembles consume more parameters in comparison to global ResNet trained with ResIST.

A.3 ROBUSTNESS TO LOCAL ITERATIONS

ResIST is robust to various numbers of local iterations [Lin et al., 2018, Zhang et al., 2016, McMahan et al., 2017]. An extensive sweep over possible values of ℓ is performed on CIFAR100. The results of this experiment are depicted in Fig. 7. As can be seen, ResIST achieves high accuracy even with thousands of local SGD iterations (i.e., previous work typically uses much fewer [Lin et al., 2018]). However, if more sub-ResNets are used, performance tends to deteriorate more quickly as local iterations increase. Due to the robustness of ResIST to large numbers of local iterations, training can be accelerated without deteriorating model performance by simply increasing the value of ℓ. Local SGD was found to demonstrate similar robustness to the number of local iterations, as shown in Fig. 7.

A.4 DEEPER ARCHITECTURES

The ResIST methodology is easily applicable to deeper architectures. To demonstrate this, results are replicated for CIFAR10 and CIFAR100 datasets with ResNet152 and ResNet200. These deeper architectures are identical to the original ResNet101 architecture (i.e., see Fig. 2). However, more residual blocks are added to the third section of the ResNet (i.e., the highlighted portion of Fig. 2) to increase the model's depth. It should be noted that convolutional blocks within the third section of the ResNet are partitioned in ResIST by default (see Sec. 2.2). As a result, all extra residual blocks within these deeper architectures are partitioned to sub-ResNets by ResIST (i.e., no extra blocks are shared between sub-ResNets), allowing ResIST to achieve greater acceleration in comparison to local SGD.

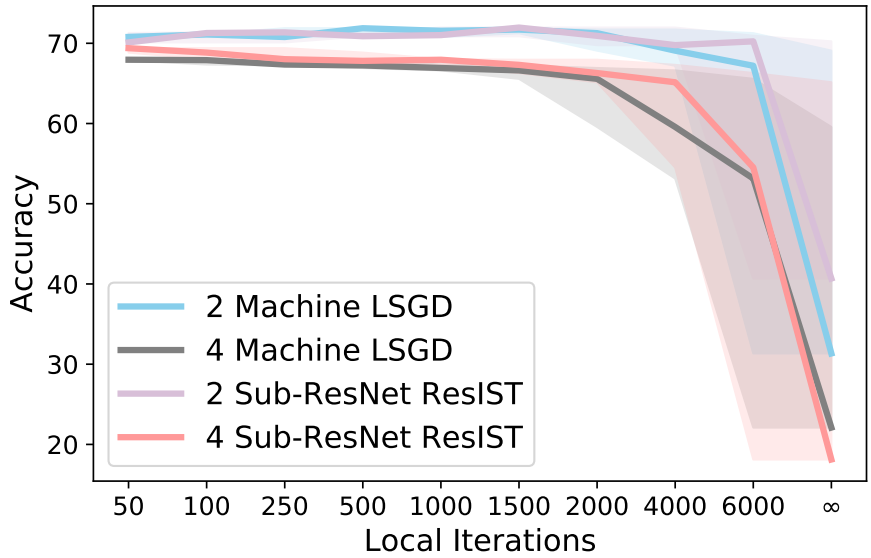

Figure 7: Test accuracy on CIFAR100 for ResNet-101 trained with both `ResIST` and local SGD (LSGD) with different numbers of local iterations. ∞ local iterations refers to aggregating parameters only once at the end of training (i.e., single-shot averaging). Shaded regions reflect deviations in accuracy.

The results of experiments with deeper ResNets are presented in Table 8. `ResIST` performs competitively with localSGD in all cases. Furthermore, `ResIST` achieves a significant speedup in comparison to local SGD that becomes more pronounced as the model becomes deeper. E.g., for 4-GPUs, `ResIST` completes training $> 3\times$ faster than local SGD for ResNet200 on both datasets. This speedup is caused by a greater ratio of total network blocks being partitioned to sub-ResNets in `ResIST`. While local SGD must communicate all parameters between machines, `ResIST` achieves a relative decrease in communication by partitioning all extra residual blocks evenly between sub-ResNets.

A.5 `RESIST` AND QUANTIZATION/SPARSE GRADIENTS

Many quantization [Alistarh et al., 2017, Yu et al., 2019] and sparsification [Aji and Heafield, 2017, Jiang and Agrawal, 2018] techniques have been proposed for reducing communication costs in distributed training. Such techniques focus on compressing communicated data, and they do not interfere with our methodology, which provides a novel approach to model synchronization and training. The proposed approach can be easily combined with existing compression techniques to further reduce communication costs and accelerate training *with no extra tuning or modifications*. To demonstrate that `ResIST` works well with quantization, we compress all communicated parameters using both four-bit and eight-bit compression. Table 9 shows that `ResIST` retains its performance until the compression level reaches five-bit and lower. We also perform experiments with sparsification of communicated weights by only keeping 25% of total weights within each synchronization round. Such a strategy reaches a validation peformance of 71.25% on CIFAR100. We summarize the results of all quantization experiments in Fig. 8, where we compare communication budgets across different compression techniques with `ResIST`. From this figure, it is clear that `ResIST` is most efficient with six-bit quantization and is compatible with most main-stream compression techniques.

Table 9: Test Accuracy for `ResIST` combined with quantization on CIFAR10 and CIFAR100 (denoted as C10 and C100).

Dataset	8 bit	7 bit	6 bit	5 bit	4 bit
C10	92.14%	92.26%	91.91%	91.35%	76.33%
C100	71.38%	72.15%	71.37%	68.29%	40.48%

B PROOF FOR `RESIST`

Suppose we have S workers, for subnetwork v at local training step $l_t \leq \ell$ and global synchronization step $t \leq T$:

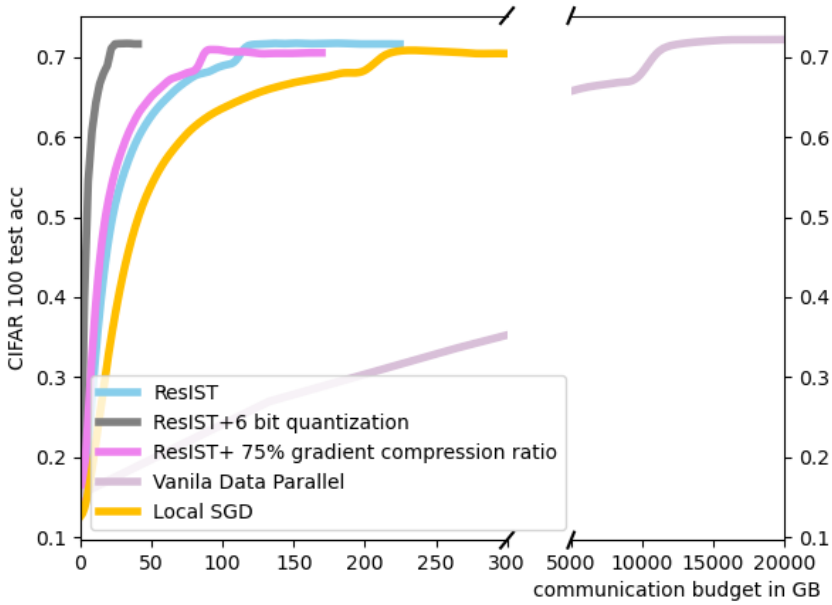

Figure 8: Test accuracy vs. communication budget for `ResIST`, `ResIST`+quantization, `ResIST`+gradient compression, local SGD and vanilla data parallel on CIFAR100. All models are trained over a 4-GPU cluster.

$$
\begin{aligned}
\mathbf{x}_{v,l_t,t}^{(1)} &= \sqrt{\frac{c_\sigma}{m}}\sigma\left(\mathbf{W}_{v,l_t,t}^{(1)}\mathbf{x}_{\mathbf{v},\mathbf{l_t},\mathbf{t}}\right), \\
\mathbf{x}_{v,l_t,t}^{(h)} &= \mathbf{x}_{v,l_t,t}^{(h-1)} + \frac{c_{res}}{H\sqrt{m}}\sigma\left(\mathbf{W}_{v,l_t,t}^{(h)}\mathbf{x}_{v,l_t,t}^{(h-1)}\right)M_{v,t}^{(h)} \\
&\qquad\qquad \text{for } 2 \le h \le H, \\
f_{res}(\mathbf{x},\theta) &= \mathbf{a}_{v,l_t,t}^{\top}\mathbf{x}_{v,l_t,t}^{(H)}
\end{aligned}
$$

where $0 < c_{res} < 1$ is a small constant and $M_{v,t}^{(h)}$ is random binary variable in layer dropout or the indicator in `ResIST` that indicates whether this layer is partitioned to this subnetwork. such mask variable is constant during local training steps and re-sampled/re-assigned at global synchronization step. In `ResIST` and other research on layer dropout for ResNet, last layer is never dropped/paritioned but shared with all workers. Thus, in the following proof, we will follow this setting.

Note here we use a $\frac{c_{res}}{H\sqrt{m}}$ scaling.

The gradient for subnetwork is

$$
\frac{\partial L}{\partial \mathbf{W}_{v,l_t,t}^{(h)}} = \frac{c_{res}}{H\sqrt{m}}\sum_{i=1}^{n}(y_i - u_i)\mathbf{x}_{i,v,l_t,t}^{(h-1)} \cdot \left[\mathbf{a}_{v,l_t,t}^{\top}\prod_{l=h+1}^{H}\left(\mathbf{I} + \frac{c_{res}}{H\sqrt{m}}\mathbf{J}_{i,v,l_t,t}^{(l)}\mathbf{W}_{v,l_t,t}^{(l)}M_{v,t}^{(l)}\right)\mathbf{J}_{i,v,l_t,t}^{(h)}M_{v,t}^{(h)}\right]
$$

For subnetwork, $\mathbf{G}^{(H)}$ has the same form as in layer drop ResNet.

The accumulated gradients of all the subnetworks:

$$
\begin{aligned}
\mathcal{W}_{t+1}^{(h)} - \mathcal{W}_t^{(h)} &= \eta\frac{\sum_{v=1}^{S}\sum_{l_t=1}^{\ell}\frac{\partial L}{\partial \mathbf{W}_{v,l_t,t}^{(h)}}}{\sum_{v=1}^{S}M_{v,t}^{(h)}} \\
&= \frac{\eta}{\sum_{v=1}^{S}M_{v,t}^{(h)}}\sum_{v=1}^{S}\sum_{l_t=1}^{\ell}\frac{c_{res}}{H\sqrt{m}}\sum_{i=1}^{n}(y_i - u_i)\mathbf{x}_{i,v,l_t,t}^{(h-1)} \cdot \left[\mathbf{a}_{v,l_t,t}^{\top}\prod_{l=h+1}^{H}\left(\mathbf{I} + \frac{c_{res}}{H\sqrt{m}}\mathbf{J}_{i,v,l_t,t}^{(l)}\mathbf{W}_{v,l_t,t}^{(l)}M_{v,t}^{(l)}\right)\mathbf{J}_{i,v,l_t,t}^{(h)}M_{v,t}^{(h)}\right]
\end{aligned}
$$

The whole network at global synchronization step t+1

$$
\begin{aligned}
\mathbf{x}_t^{(1)} &= \sqrt{\frac{c_\sigma}{m}}\sigma\left(\frac{\sum_{v=1}^{S}\mathbf{W}_{v,\ell,t}^{(1)}}{S}\mathbf{x_t}\right), \\
\mathbf{x}_t^{(h)} &= \mathbf{x}_t^{(h-1)} + \frac{c_{res}}{H\sqrt{m}}\sigma\left(\frac{\sum_{v=1}^{S}\mathbf{W}_{v,\ell,t}^{(h)}M_{v,t}^{(h)}}{\sum_{v=1}^{S}M_{v,t}^{(h)}}\mathbf{x}_t^{(h-1)}\right) \\
&\qquad \text{for } 2 \le h \le H, \\
f_{res}(\mathbf{x},\theta) &= \frac{\sum_{v=1}^{S}\mathbf{a}_{v,\ell,t}}{S}^{\top}\mathbf{x}_t^{(H)}
\end{aligned}
$$

Let $\mathcal{W}_t^{(h)} = \frac{\sum_{v=1}^{S}\mathbf{W}_{v,\ell,t}^{(h)}M_{v,t}^{(h)}}{\sum_{v=1}^{S}M_{v,t}^{(h)}}$, $\mathbf{a}_t = \frac{\sum_{v=1}^{S}\mathbf{a}_{v,\ell,t}}{S}$

The whole network at global synchronization step 0

$$
\begin{aligned}
\mathbf{x}_0^{(1)} &= \sqrt{\frac{c_\sigma}{m}}\sigma\left(\mathbf{W}_0^{(1)}\mathbf{x}\right), \\
\mathbf{x}_0^{(h)} &= \mathbf{x}_0^{(h-1)} + \frac{c_{res}}{H\sqrt{m}}\sigma\left(\mathbf{W}_0^{(h)}\mathbf{x}_0^{(h-1)}\right) \\
&\qquad \text{for } 2 \le h \le H, \\
f_{res}(\mathbf{x},\theta) &= \mathbf{a}_0^{\top}\mathbf{x}_0^{(H)}
\end{aligned}
$$

B.1 PROOF SKETCH

We can write the loss of the whole network at global syncrhonization step t+1 as

$$
L(\theta(t), \mathbf{M}_{1,t}, \mathbf{M}_{2,t}...\mathbf{M}_{S,t}) = \frac{1}{2}\left\|\mathbf{y} - \mathbf{u}(t, \mathbf{M}_{1,t}, \mathbf{M}_{2,t}...\mathbf{M}_{S,t})\right\|_2^2.
$$

where $\mathbf{M}_{v,t} = \{M_{v,t}^{(1)}, M_{v,t}^{(2)}...M_{v,t}^{(H)}\}$ Let $\mathcal{M}_t = \{\mathbf{M}_{1,t}, \mathbf{M}_{2,t}...\mathbf{M}_{S,t}\}$

For convience, we drop all mask notation in the following proof. Let $\hat{u}(t)$ be the output of the whole network at global synchronization step $t+1$. Now recall the progress of loss function:

$$
\|\mathbf{y} - \hat{\mathbf{u}}(t+1)\|_2^2 = \|\mathbf{y} - \hat{\mathbf{u}}(t)\|_2^2 - 2\left(\mathbf{y} - \hat{\mathbf{u}}(t)\right)^{\top}\left(\hat{\mathbf{u}}(t+1) - \hat{\mathbf{u}}(t)\right) + \|\hat{\mathbf{u}}(t+1) - \hat{\mathbf{u}}(t)\|_2^2
$$

Following Du et al. [2019], we apply Taylor expansion on $(\hat{\mathbf{u}}(t+1) - \hat{\mathbf{u}}(t))$ and look at the ith coordinate.

$$
\begin{aligned}
\hat{u}_i(t+1) - \hat{u}_i(t) &= -\langle\theta(t+1) - \theta(t), \hat{u}_i'\left(\theta(t)\right)\rangle + \int_{s=0}^{1}\langle\theta(t+1) - \theta(t), \hat{u}_i'\left(\theta(t)\right) - \hat{u}_i'\left(\theta(t) - s(\theta(t) - \theta(t+1))\right)\rangle ds \\
&\triangleq I_1^i(t) + I_2^i(t)
\end{aligned}
$$

However, it is not obvious that $I_1(t)$ and $I_2(t)$ can be directed bounded to show the decrease of the loss of the whole network as both of them involve the accumulated gradient change from distributed local subnetwork training. Thus, we introduce a new term $I_1'^i(t)$ as below, which relates to the hypothetical global gradient direction as if the whole network trained centrally.

$$
\begin{aligned}
I_1'^i(t) &= -\eta\ell\langle L'(\theta(t)), \hat{u}_i'\left(\theta(t)\right)\rangle \\
&= -\eta\ell\sum_{j=1}^{n}(\hat{u}_j - y_j)\langle\hat{u}_j'(\theta(t)), \hat{u}_i'\left(\theta(t)\right)\rangle
\end{aligned}
$$

$$\triangleq -\eta\ell \sum_{j=1}^{n}(\hat{u}_j - y_j) \sum_{h=1}^{H+1} \hat{\mathbf{G}}_{ij}^{(h)}(t)$$

Accordingly,

$$\begin{aligned}
&\|\mathbf{y} - \hat{\mathbf{u}}(t+1)\|_2^2 \\
&= \|\mathbf{y} - \hat{\mathbf{u}}(t)\|_2^2 - 2\left(\mathbf{y} - \hat{\mathbf{u}}(t)\right)^\top \left(\mathbf{I}_1(t) + \mathbf{I}_2(t) + \mathbf{I'}_1(t) - \mathbf{I'}_1(t)\right) + \|\hat{\mathbf{u}}(t+1) - \hat{\mathbf{u}}(t)\|_2^2 \\
&= \|\mathbf{y} - \hat{\mathbf{u}}(t)\|_2^2 - 2\left(\mathbf{y} - \hat{\mathbf{u}}(t)\right)^\top \mathbf{I'}_1(t) + 2\left(\mathbf{y} - \hat{\mathbf{u}}(t)\right)^\top \left(\mathbf{I'}_1(t) - \mathbf{I}_1(t)\right) - 2\left(\mathbf{y} - \hat{\mathbf{u}}(t)\right)^\top \mathbf{I}_2(t) + \|\hat{\mathbf{u}}(t+1) - \hat{\mathbf{u}}(t)\|_2^2 \\
&\leq \left(1 - \eta\ell\lambda_{\min}\left(\hat{\mathbf{G}}^{(H)}(t)\right)\right) \|\mathbf{y} - \hat{\mathbf{u}}(t)\|_2^2 + 2\left(\mathbf{y} - \hat{\mathbf{u}}(t)\right)^\top \left(\mathbf{I'}_1(t) - \mathbf{I}_1(t)\right) \\
&\quad - 2\left(\mathbf{y} - \hat{\mathbf{u}}(t)\right)^\top \mathbf{I}_2(t) + \|\hat{\mathbf{u}}(t+1) - \hat{\mathbf{u}}(t)\|_2^2 .
\end{aligned}$$

Our hypothesis is:

Condition B.1. *At the $t+1$-th global synchronization, for the whole network, we have*

$$\|\mathbf{y} - \hat{\mathbf{u}}(t, \mathcal{M}_t)\|_2^2 \leq (1 - \frac{\eta\ell\lambda_0}{2})^t \|\mathbf{y} - \hat{\mathbf{u}}(0)\|_2^2 .$$

In order to prove this, we need to show $2\left(\mathbf{y} - \hat{\mathbf{u}}(t)\right)^\top \left(\mathbf{I'}_1(t) - \mathbf{I}_1(t)\right)$,$-2\left(\mathbf{y} - \hat{\mathbf{u}}(t)\right)^\top \mathbf{I}_2(t)$ and $\|\hat{\mathbf{u}}(t+1) - \hat{\mathbf{u}}(t)\|_2^2$ are proportional to $\eta^2 \|\mathbf{y} - \hat{\mathbf{u}}(t)\|_2^2$ so if we set η sufficiently small, this term is smaller than $\eta\lambda_{\min}\left(\hat{\mathbf{G}}^{(H)}(t)\right) \|\mathbf{y} - \hat{\mathbf{u}}(t)\|_2^2$ and thus the loss function decreases with a linear rate.

Further, similar to Du et al. [2019], to prove the induction hypothesis, it suffices to prove $\lambda_{\min}\left(\hat{\mathbf{G}}^{(H)}(t)\right) \geq \frac{\lambda_0}{2}$ for $t' = 0, \ldots, t$, where λ_0 is independent of m. Similar to Du et al. [2019], we can show at the beginning

$$\lambda_{\min}\left(\hat{\mathbf{G}}^{(H)}(0)\right) \geq \frac{3}{4}\lambda_0.$$

Now for the t-th global iteration, by matrix perturbation analysis, we know it is sufficient to show $\left\|\hat{\mathbf{G}}^{(H)}(t) - \hat{\mathbf{G}}^{(H)}(0)\right\|_2 \leq \frac{1}{4}\lambda_0$. To do this, we show as long as m is large enough, every weight matrix is close its initialization in a relative error sense.

Lemma B.1 (Lemma on Initialization Norms for the whole network)**.** *If $\sigma(\cdot)$ is $L-$Lipschitz and $m = \Omega\left(\frac{n}{\delta}\right)$, assuming $\left\|\mathcal{W}_0^{(h)}\right\|_2 \leq c_{w,0}\sqrt{m}$ for $h \in [2, H]$ and $c_{w,0} \approx 2$ for Gaussian initialization. We have with probability at least $1-\delta$ over random initialization, for every $h \in [H]$ and $i \in [n]$,*

$$\frac{1}{c_{x,0}} \leq \left\|\mathbf{x}_{i,0}^{(h)}\right\|_2 \leq c_{x,0}$$

for some universal constant $c_{x,0} > 1$

Proof of Lemma B.1. As the global model at initialization is the same with original ResNet in Du et al. [2019], we can use the same proof in Lemma C.1 in Du et al. [2019]. □

The following lemma lower bounds $\hat{\mathbf{G}}^{(H)}(0)$'s least eigenvalue.

Lemma B.2 (Least Eigenvalue at the Initialization)**.** *If $m = \Omega\left(\frac{n^2 \log(Hn/\delta)}{\lambda_0^2}\right)$, we have*

$$\lambda_{\min}(\hat{\mathbf{G}}^{(H)}(0)) \geq \frac{3}{4}\lambda_0.$$

Proof of Lemma B.2. As the global model at initialization is the same with original ResNet in Du et al. [2019], we can use the same proof in Lemma C.2 in Du et al. [2019]. □

Lemma B.3. *Suppose $\sigma(\cdot)$ is L-Lipschitz and for $h \in [H]$, $\left\|\mathcal{W}_0^{(h)}\right\|_2 \leq c_{w,0}\sqrt{m}$, $\left\|\mathbf{x}_0^{(h)}\right\|_2 \leq c_{x,0}$ and $\left\|\mathbf{W}_{v,l_t,t}^{(h)} - \mathcal{W}_0^{(h)}\right\|_F \leq \sqrt{m}R$ for some constant $c_{w,0}, c_{x,0} > 0$ and $R \leq c_{w,0}$. Then we have*

$$\left\|\mathbf{x}_{v,l_t,t}^{(h)} - \mathbf{x}_0^{(h)}\right\|_2 \leq \left(\sqrt{c_\sigma}L + \frac{c_{x,0}}{c_{w,0}} + \frac{c_{x,0}}{R}\right) e^{2c_{res}c_{w,0}L} R \triangleq c_x' R.$$

Proof of Lemma B.3. We prove this lemma by induction. Our induction hypothesis is

$$\left\|\mathbf{x}_{v,l_t,t}^{(h)} - \mathbf{x}_0^{(h)}\right\|_2 \leq g(h),$$

where

$$g(h) = \left[1 + \frac{2c_{res}c_{w,0}L}{H}\right] g(h-1) + \frac{c_{res}Lc_{x,0}}{H}(c_{w,0} + R).$$

For $h = 1$, we have

$$\begin{aligned}
\left\|\mathbf{x}_{v,l_t,t}^{(1)} - \mathbf{x}_0^{(1)}\right\|_2 &\leq \sqrt{\frac{c_\sigma}{m}} \left\|\sigma\left(\mathbf{W}_{v,l_t,t}^{(1)}\mathbf{x}\right) - \sigma\left(\mathcal{W}_0^{(1)}\mathbf{x}\right)\right\|_2 \\
&\leq \sqrt{\frac{c_\sigma}{m}} L \left\|\mathbf{W}_{v,l_t,t}^{(1)} - \mathcal{W}_0^{(1)}\right\|_F \leq \sqrt{c_\sigma} LR,
\end{aligned}$$

which implies $g(1) = \sqrt{c_\sigma}LR$, for $2 \leq h \leq H$, we have

$$\begin{aligned}
&\left\|\mathbf{x}_{v,l_t,t}^{(h)} - \mathbf{x}_0^{(h)}\right\|_2 \leq \frac{c_{res}}{H\sqrt{m}} \left\|\sigma\left(\mathbf{W}_{v,l_t,t}^{(h)}\mathbf{x}_{v,l_t,t}^{(h-1)}\right) M_{v,t}^{(h)} - \sigma\left(\mathcal{W}_0^{(h)}\mathbf{x}_0^{(h-1)}\right) 1\right\|_2 \\
&+ \left\|\mathbf{x}_{v,l_t,t}^{(h-1)} - \mathbf{x}_0^{(h-1)}\right\|_2 \\
&\leq \frac{c_{res}}{H\sqrt{m}} \left\|[\sigma\left(\mathbf{W}_{v,l_t,t}^{(h)}\mathbf{x}_{v,l_t,t}^{(h-1)}\right) - \sigma\left(\mathcal{W}_0^{(h)}\mathbf{x}_0^{(h-1)}\right)]M_{v,t}^{(h)} + \sigma\left(\mathcal{W}_0^{(h)}\mathbf{x}_0^{(h-1)}\right)(M_{v,t}^{(h)} - 1)\right\|_2 \\
&+ \left\|\mathbf{x}_{v,l_t,t}^{(h-1)} - \mathbf{x}_0^{(h-1)}\right\|_2 \\
&\leq \frac{c_{res}}{H\sqrt{m}} \left\|[\sigma\left(\mathbf{W}_{v,l_t,t}^{(h)}\mathbf{x}_{v,l_t,t}^{(h-1)}\right) - \sigma\left(\mathcal{W}_0^{(h)}\mathbf{x}_0^{(h-1)}\right)]M_{v,t}^{(h)}\right\|_2 + \frac{c_{res}}{H\sqrt{m}} \left\|\sigma\left(\mathcal{W}_0^{(h)}\mathbf{x}_0^{(h-1)}\right)(M_{v,t}^{(h)} - 1)\right\|_2 \\
&+ \left\|\mathbf{x}_{v,l_t,t}^{(h-1)} - \mathbf{x}_0^{(h-1)}\right\|_2 \\
&\leq \frac{c_{res}}{H\sqrt{m}} \left\|[\sigma\left(\mathbf{W}_{v,l_t,t}^{(h)}\mathbf{x}_{v,l_t,t}^{(h-1)}\right) - \sigma\left(\mathcal{W}_0^{(h)}\mathbf{x}_0^{(h-1)}\right)]\right\|_2 \left\|M_{v,t}^{(h)}\right\|_2 + \frac{c_{res}}{H\sqrt{m}} \left\|\sigma\left(\mathcal{W}_0^{(h)}\mathbf{x}_0^{(h-1)}\right)\right\|_2 \left\|(M_{v,t}^{(h)} - 1)\right\|_2 \\
&+ \left\|\mathbf{x}_{v,l_t,t}^{(h-1)} - \mathbf{x}_0^{(h-1)}\right\|_2 \\
&\leq \frac{c_{res}}{H\sqrt{m}} \left\|[\sigma\left(\mathbf{W}_{v,l_t,t}^{(h)}\mathbf{x}_{v,l_t,t}^{(h-1)}\right) - \sigma\left(\mathbf{W}_{v,l_t,t}^{(h)}\mathbf{x}_0^{(h-1)}\right) + \sigma\left(\mathbf{W}_{v,l_t,t}^{(h)}\mathbf{x}_{v,l_t,t}^{(h-1)}\right) - \sigma\left(\mathcal{W}_0^{(h)}\mathbf{x}_0^{(h-1)}\right)]\right\|_2 \\
&+ \frac{c_{res}L}{H\sqrt{m}} \left\|\mathcal{W}_0^{(h)}\mathbf{x}_0^{(h-1)}\right\| + \left\|\mathbf{x}_{v,l_t,t}^{(h-1)} - \mathbf{x}_0^{(h-1)}\right\|_2 \\
&\leq \frac{c_{res}}{H\sqrt{m}} \left\|\sigma\left(\mathbf{W}_{v,l_t,t}^{(h)}\mathbf{x}_{v,l_t,t}^{(h-1)}\right) - \sigma\left(\mathbf{W}_{v,l_t,t}^{(h)}\mathbf{x}_0^{(h-1)}\right)\right\|_2 \\
&+ \frac{c_{res}}{H\sqrt{m}} \left\|\sigma\left(\mathbf{W}_{v,l_t,t}^{(h)}\mathbf{x}_0^{(h-1)}\right) - \sigma\left(\mathcal{W}_0^{(h)}\mathbf{x}_0^{(h-1)}\right)\right\|_2 \\
&+ \left\|\mathbf{x}_{v,l_t,t}^{(h-1)} - \mathbf{x}_0^{(h-1)}\right\|_2 + \frac{c_{res}Lc_{w,0}c_{x,0}}{H} \\
&\leq \frac{c_{res}L}{H\sqrt{m}} \left(\left\|\mathcal{W}_0^{(h)}\right\|_2 + \left\|\mathbf{W}_{v,l_t,t}^{(h)} - \mathcal{W}_0^{(h)}\right\|_F\right) \cdot \left\|\mathbf{x}_{v,l_t,t}^{(h-1)} - \mathbf{x}_0^{(h-1)}\right\|_2 \\
&+ \frac{c_{res}L}{H\sqrt{m}} \left\|\mathbf{W}_{v,l_t,t}^{(h)} - \mathcal{W}_0^{(h)}\right\|_F \left\|\mathbf{x}_0^{(h-1)}\right\|_2 + \left\|\mathbf{x}_{v,l_t,t}^{(h-1)} - \mathbf{x}_0^{(h-1)}\right\|_2 + \frac{c_{res}Lc_{w,0}c_{x,0}}{H} \\
&\leq \left[1 + \frac{c_{res}L}{H\sqrt{m}}\left(c_{w,0}\sqrt{m} + R\sqrt{m}\right)\right] g(h-1) + \frac{c_{res}L}{H\sqrt{m}}\sqrt{m}Rc_{x,0} + \frac{c_{res}Lc_{w,0}c_{x,0}}{H}
\end{aligned}$$

$$\leq \left(1+\frac{2c_{res}c_{w,0}L}{H}\right)g(h-1)+\frac{c_{res}}{H}Lc_{x,0}R+\frac{c_{res}Lc_{w,0}c_{x,0}}{H}.$$

Lastly, simple calculations show $g(h) \leq \left(\sqrt{c_\sigma}L+\frac{c_{x,0}}{c_{w,0}}+\frac{c_{x,0}}{R}\right)e^{2c_{res}c_{w,0}L}R$.

□

Lemma B.4. *Suppose $\sigma(\cdot)$ is L-Lipschitz and for $h \in [H]$, $\left\|\mathcal{W}_0^{(h)}\right\|_2 \leq c_{w,0}\sqrt{m}$, $\left\|\mathbf{x}_0^{(h)}\right\|_2 \leq c_{x,0}$ and $\left\|\mathcal{W}_t^{(h)}-\mathcal{W}_0^{(h)}\right\|_F \leq \sqrt{m}R$ for some constant $c_{w,0}, c_{x,0} > 0$ and $R \leq c_{w,0}$. Then we have*

$$\left\|\mathbf{x}_t^{(h)}-\mathbf{x}_0^{(h)}\right\|_2 \leq \left(\sqrt{c_\sigma}L+\frac{c_{x,0}}{c_{w,0}}\right)e^{2c_{res}c_{w,0}L}R \triangleq c_xR.$$

Proof of Lemma B.4. The proof is exactly the same with proof of C.3 in Du et al. [2019] □

Next, we characterize how the perturbation on the weight matrices affect $\hat{\mathbf{G}}^{(H)}$.

Lemma B.5. *Suppose $\sigma(\cdot)$ is differentiable, $L-$Lipschitz and $\beta-$smooth. Suppose $\sigma(\cdot)$ is $L-$Lipschitz and $\beta-$smooth. Suppose for $h \in [H]$, $\left\|\mathcal{W}_0^{(h)}\right\|_2 \leq c_{w,0}\sqrt{m}$, $\|\mathbf{a}_0\|_2 \leq a_{2,0}\sqrt{m}$, $\|\mathbf{a}_0\|_4 \leq a_{4,0}m^{1/4}$, $\frac{1}{c_{x,0}} \leq \left\|\mathbf{x}_0^{(h)}\right\|_2 \leq c_{x,0}$, if $\left\|\mathcal{W}_r^{(h)}-\mathcal{W}_0^{(h)}\right\|_F, \|\mathbf{a}_r-\mathbf{a}_0\|_2 \leq \sqrt{m}R$ where $R \leq c\lambda_0H^2n^{-1}$ and $R \leq c$ for some small constant c , we have*

$$\left\|\hat{\mathbf{G}}^{(H)}(t)-\hat{\mathbf{G}}^{(H)}(0)\right\|_2 \leq \frac{\lambda_0}{2}.$$

Proof of Lemma B.5. Simialr to C.4 in Du et al. [2019] Because Frobenius-norm of a matrix is bigger than the operator norm, it is sufficient to bound $\left\|\hat{\mathbf{G}}^{(H)}(t)-\hat{\mathbf{G}}^{(H)}(0)\right\|_F$. For simplicity define $z_{i,q}(t)=\mathcal{W}_{t,q}^{(H)\top}\mathbf{x}_{i,t}^{(H-1)}$, we have

$$\begin{aligned}
&\left|\hat{\mathbf{G}}_{i,j}^{(H)}(t)-\hat{\mathbf{G}}_{i,j}^{(H)}(0)\right| \\
=&\frac{c_{res}^2}{H^2m}\Big|\mathbf{x}_{i,t}^{(H-1)\top}\mathbf{x}_{j,t}^{(H-1)}\sum_{q=1}^{m}a_q(t)^2\sigma'\left(z_{i,q}(t)\right)\sigma'\left(z_{j,q}(t)\right) \\
&-\mathbf{x}_{i,0}^{(H-1)\top}\mathbf{x}_{j,0}^{(H-1)}\sum_{q=1}^{m}a_q(0)^2\sigma'\left(z_{i,q}(0)\right)\sigma'\left(z_{j,q}(0)\right)\Big| \\
\leq&\frac{c_{res}^2}{H^2}L^2a_{2,0}^2\left|\mathbf{x}_{i,t}^{(H-1)\top}\mathbf{x}_{j,t}^{(H-1)}-\mathbf{x}_{i,0}^{(H-1)\top}\mathbf{x}_{j,0}^{(H-1)}\right| \\
&+\frac{c_{res}^2}{H^2}\frac{c_{x,0}^2}{m}\left|\sum_{q=1}^{m}a_q(0)^2\left(\sigma'\left(z_{i,q}(t)\right)\sigma'\left(z_{j,q}(t)\right)-\sigma'\left(z_{i,q}(0)\right)\sigma'\left(z_{j,q}(0)\right)\right)\right| \\
&+\frac{c_{res}^2}{H^2m}\left|\mathbf{x}_{i,t}^{(H-1)\top}\mathbf{x}_{j,t}^{(H-1)}\right|\left|\sum_{q=1}^{m}\left(a_q(t)^2-a_q(0)^2\right)\sigma'\left(z_{i,q}(t)\right)\sigma'\left(z_{j,q}(t)\right)\right| \\
\triangleq&\frac{c_{res}^2}{H^2}(I_1^{i,j}+I_2^{i,j}+I_3^{i,j}).
\end{aligned}$$

For $I_1^{i,j}$, using Lemma B.4, we have

$$\begin{aligned}
I_1^{i,j}=&L^2a_{2,0}^2\left|\mathbf{x}_{i,t}^{(H-1)\top}\mathbf{x}_{j,t}^{(H-1)}-\mathbf{x}_{i,0}^{(H-1)\top}\mathbf{x}_{j,0}^{(H-1)}\right| \\
\leq&L^2a_{2,0}^2\left|(\mathbf{x}_{i,t}^{(H-1)}-\mathbf{x}_{i,0}^{(H-1)})^\top\mathbf{x}_{j,t}^{(H-1)}\right|+L^2a_{2,0}^2\left|\mathbf{x}_{i,0}^{(H-1)\top}(\mathbf{x}_{i,t}^{(H-1)}-\mathbf{x}_{i,0}^{(H-1)})\right| \\
\leq&c_xL^2a_{2,0}^2R\cdot(c_{x,0}+c_xR)+c_{x,0}c_xL^2a_{2,0}^2R \\
\leq&3c_{x,0}c_xL^2a_{2,0}^2R,
\end{aligned}$$

Same with C.4 Du et al. [2019], to bound $I_2^{i,j}$, we have

$$\begin{aligned} I_2^{i,j} =& c_{x,0}^2 \frac{1}{m} \left| \sum_{q=1}^{m} a_q(0)^2 \sigma'\left(z_{i,q}(t)\right) \sigma'\left(z_{j,q}(t)\right) - a_q(0)^2 \sigma'\left(z_{i,q}(0)\right) \sigma'\left(z_{j,q}(0)\right) \right| \\ \leq& \frac{\beta L a_{4,0}^2 c_{x,0}^2}{\sqrt{m}} \left(\sqrt{\sum_{q=1}^{m} \left|z_{i,q}(t) - z_{i,q}(0)\right|^2} + \sqrt{\sum_{q=1}^{m} \left|z_{j,q}(t) - z_{j,q}(0)\right|^2} \right). \end{aligned}$$

Using the same proof for Lemma B.4, it is easy to see

$$\sum_{q=1}^{m} \left|z_{i,q}(t) - z_{i,q}(0)\right|^2 \leq \left(2c_x c_{w,0} + c_{x,0}\right)^2 L^2 m R^2.$$

Thus

$$I_2^{i,j} \leq 2\beta c_{x,0}^2 \left(2c_x c_{w,0} + c_{x,0}\right) L^2 R.$$

The bound of $I_3^{i,j}$ is the same to that $I_3^{i,j}$ in Du et al. [2019] C.4,

$$I_3^{i,j} \leq 12 L^2 c_{x,0}^2 a_{2,0} R.$$

Therefore we can bound the perturbation

$$\begin{aligned} \left\| \hat{\mathbf{G}}^{(H)}(t) - \hat{\mathbf{G}}^{(H)}(0) \right\|_F =& \sqrt{\sum_{(i,j)}^{n,n} \left| \hat{\mathbf{G}}_{i,j}^{(H)}(t) - \hat{\mathbf{G}}_{i,j}^{(H)}(0) \right|^2} \\ \leq& \frac{c_{res}^2}{H^2} \sqrt{n^2 (3c_{x,0} c_x L^2 a_{2,0}^2 R + 2\beta c_{x,0}^2 \left(2c_x c_{w,0} + c_{x,0}\right) L^2 R + 12 L^2 c_{x,0}^2 a_{2,0} R)} \\ =& \frac{c_{res}^2}{H^2} n (3c_{x,0} c_x L^2 a_{2,0}^2 R + 2\beta c_{x,0}^2 \left(2c_x c_{w,0} + c_{x,0}\right) L^2 R + 12 L^2 c_{x,0}^2 a_{2,0} R) \end{aligned}$$

.

Plugging in the bound on R, we have the desired result. □

Now we prove theorem 3.1 by induction, assume the condition B.1, we want to bound the change of weight to satisfy lemma B.5 and then we want to show $2\left(\mathbf{y} - \hat{\mathbf{u}}(t)\right)^\top \left(\mathbf{I'}_1(t) - \mathbf{I}_1(t)\right)$,$-2\left(\mathbf{y} - \hat{\mathbf{u}}(t)\right)^\top \mathbf{I}_2(t)$ and $\|\hat{\mathbf{u}}(t+1) - \hat{\mathbf{u}}(t)\|_2^2$ are proportional to $\eta^2 \|\mathbf{y} - \mathbf{u}(t)\|_2^2$ so if we set η sufficiently small, this term is smaller than $\eta \lambda_{\min}\left(\hat{\mathbf{G}}^{(H)}(t)\right) \|\mathbf{y} - \mathbf{u}(t)\|_2^2$ and thus the loss function decreases with a linear rate.

Lemma B.6. *If Condition B.1 holds for $t' = 0, \ldots, t-1$, we have for any $1 \leq v \leq S$, $0 \leq l_t \leq \ell$*

$$\begin{aligned} &\left\| \mathbf{W}_{v,l_t,t}^{(h)} - \mathcal{W}_0^{(h)} \right\|_F, \left\| \mathbf{a}_{v,l_t,t} - \mathbf{a}_0 \right\|_2 \leq R' \sqrt{m}, \\ &\left\| \mathbf{W}_{v,l_t,t}^{(h)} - \mathbf{W}_{v,l_t-1,t}^{(h)} \right\|_F, \left\| \mathbf{a}_{v,l_t,t} - \mathbf{a}_{v,l_t-1,t} \right\|_2 \leq \eta Q'(l_t - 1, t), \end{aligned}$$

where $R' = \frac{16 c_{res} c_{x,0} a_{2,0} L e^{2c_{res} c_{w,0} L} \sqrt{n} \|\mathbf{y} - \mathbf{u}(0)\|_2}{H \lambda_0 \sqrt{m}} < c$ *for some small constant* c *,*
$Q'(l_t, t) = 4 c_{res} c_{x,0} a_{2,0} L e^{2c_{res} c_{w,0} L} \sqrt{n} \left\|\mathbf{y} - \mathbf{u}_{t,l_t}\right\|_2 / H$ *and*
$Q'(t) = 4 c_{res} c_{x,0} a_{2,0} L e^{2c_{res} c_{w,0} L} \sqrt{n} \left\|\mathbf{y} - \hat{\mathbf{u}}_t\right\|_2 / H.$

Proof of Lemma B.6. We will prove this corollary by induction. The induction hypothesis is

$$\begin{aligned} \left\| \mathbf{W}_{v,l_t,t}^{(h)} - \mathcal{W}_0^{(h)} \right\|_F &\leq R' \sqrt{m} \\ \left\| \mathbf{a}_{v,l_t,t} - \mathbf{a}_0 \right\|_2 &\leq R' \sqrt{m}. \end{aligned}$$

First we want to prove it holds for $t' = 0$ and $0 \le l_t \le \ell$.

We prove it by induction w.r.t l_t: It is easy to see that it holds for $t' = 0$ and $l'_t = 0$. Suppose it holds for $0 \le l'_t \le l_t$, we want to prove it holds for $l'_t = l_t + 1$ Following C.5 in Du et al. [2019], note $\left\|\mathbf{J}^{(k)}_{i,v,l_t,t}\right\|_2 \le L$. We have

$$
\begin{aligned}
&\left\|\mathbf{W}^{(h)}_{v,l_t+1,t} - \mathbf{W}^{(h)}_{v,l_t,t}\right\|_F \\
\le& \eta \frac{c_{res}}{H\sqrt{m}} \left\|\mathbf{a_{v,l_t,t}}\right\|_2 \sum_{i=1}^{n} |y_i - u_{i,v,l_t,t}| \left\|\mathbf{x}^{(h-1)}_{i,v,l_t,t}\right\|_2 \prod_{k=h+1}^{H} \left\|\mathbf{I} + \frac{c_{res}}{H\sqrt{m}} \mathbf{J}^{(k)}_{i,v,l_t,t} \mathbf{W}^{(k)}_{v,l_t,t} M^{(k)}_{v,t}\right\|_2 \left\|\mathbf{J}^{(k)}_{i,v,l_t,t}\right\|_2 \left\|M^{(h)}_{v,t}\right\|_2 \\
\le& \eta \frac{Lc_{res}}{H\sqrt{m}} \left\|\mathbf{a_{v,l_t,t}}\right\|_2 \sum_{i=1}^{n} |y_i - u_{i,v,l_t,t}| \left\|\mathbf{x}^{(h-1)}_{i,v,l_t,t}\right\|_2 \prod_{k=h+1}^{H} \left\|\mathbf{I} + \frac{c_{res}}{H\sqrt{m}} \mathbf{J}^{(k)}_{i,v,l_t,t} \mathbf{W}^{(k)}_{v,l_t,t} M^{(k)}_{v,t}\right\|_2
\end{aligned}
$$

Further

$$
\begin{aligned}
&\prod_{k=h+1}^{H} \left\|\mathbf{I} + \frac{c_{res}}{H\sqrt{m}} \mathbf{J}^{(k)}_{i,v,l_t,t} \mathbf{W}^{(k)}_{v,l_t,t} M^{(k)}_{v,t}\right\|_2 \\
\le& \prod_{k=h+1}^{H} \|\mathbf{I}\|_2 + \left\|\frac{c_{res}}{H\sqrt{m}} \mathbf{J}^{(k)}_{i,v,l_t,t} \mathbf{W}^{(k)}_{v,l_t,t} M^{(k)}_{v,t}\right\|_2 \\
\le& \prod_{k=h+1}^{H} \|\mathbf{I}\|_2 + \frac{c_{res}}{H\sqrt{m}} \left\|\mathbf{J}^{(k)}_{i,v,l_t,t}\right\|_2 \left\|\mathbf{W}^{(k)}_{v,l_t,t}\right\|_2 \left\|M^{(k)}_{v,t}\right\|_2 \\
\le& \prod_{k=h+1}^{H} \|\mathbf{I}\|_2 + \frac{c_{res}L}{H\sqrt{m}} (\left\|\mathcal{W}^{(k)}_0\right\|_F + \left\|\mathbf{W}^{(k)}_{v,l_t,t} - \mathcal{W}^{(k)}_0\right\|_F) \\
\le& \prod_{k=h+1}^{H} 1 + \frac{c_{res}L}{H} (c_{w,0} + R') \\
\le& \prod_{k=h+1}^{H} 1 + \frac{c_{res}L}{H} 2c_{w,0} \\
\le& e^{2c_{res}x_{w,0}L}
\end{aligned}
$$

Thus

$$
\begin{aligned}
&\left\|\mathbf{W}^{(h)}_{v,l_t+1,t} - \mathbf{W}^{(h)}_{v,l_t,t}\right\|_F \\
\le& \eta \frac{Lc_{res}}{H\sqrt{m}} \left\|\mathbf{a}_{v,l_t,t}\right\|_2 \sum_{i=1}^{n} |y_i - u_i(s)| \left\|\mathbf{x}^{(h-1)}_{i,v,l_t,t}\right\|_2 \prod_{k=h+1}^{H} \left\|\mathbf{I} + \frac{c_{res}}{H\sqrt{m}} \mathbf{J}^{(k)}_{i,v,l_t,t} \mathbf{W}^{(k)}_{v,l_t,t} M^{(k)}_{v,t}\right\|_2 \\
\le& \eta \frac{Lc_{res}}{H\sqrt{m}} \left\|\mathbf{a}_{v,l_t,t}\right\|_2 \sum_{i=1}^{n} |y_i - u_i(s)| \left\|\mathbf{x}^{(h-1)}_{i,v,l_t,t}\right\|_2 e^{2c_{res}x_{w,0}L} \\
\le& \eta c_{res}(c_{x,0} + c_x R') L a_{2,0} e^{2c_{res}c_{w,0}L} \sqrt{n} \left\|\mathbf{y} - \mathbf{u}_{l_t,t}\right\|_2 / H \\
\le& 3\eta c_{res} c_{x,0} L a_{2,0} e^{2c_{res}c_{w,0}L} \sqrt{n} \left\|\mathbf{y} - \mathbf{u}_{l_t,t}\right\|_2 / H \\
\le& \eta Q'(l_t, t) \\
\le& (1 - \frac{\eta\lambda_0}{2})^{s/2} \frac{1}{4} \eta\lambda_0 R' \sqrt{m}
\end{aligned}
$$

Similarly, we have

$$
\left\|\mathbf{a}_{v,l_t+1,t} - \mathbf{a}_{v,l_t,t}\right\|_2 \le 3\eta c_{x,0} \sum_{i=1}^{n} |y_i - u_{l_t,t}|
$$

$$
\begin{aligned}
&\leq \eta Q'(l_t, t) \\
&\leq (1 - \frac{\eta\lambda_0}{2})^{l_t/2} \frac{1}{4} \eta\lambda_0 R' \sqrt{m}.
\end{aligned}
$$

Thus

$$
\begin{aligned}
&\left\| \mathbf{W}^{(h)}_{v,l_t+1,t} - \mathcal{W}^{(h)}_0 \right\|_F \\
\leq &\left\| \mathbf{W}^{(h)}_{v,l_t+1,t} - \mathcal{W}^{(h)}_{v,l_t,t} \right\|_F + \left\| \mathbf{W}^{(h)}_{v,l_t,t} - \mathcal{W}^{(h)}_0 \right\|_F \\
\leq &\sum_{l'_t=0}^{l_t} \eta (1 - \frac{\eta\lambda_0}{2})^{l'_t/2} \frac{1}{4} \eta\lambda_0 R' \sqrt{m}.
\end{aligned}
$$

Similarly,

$$
\begin{aligned}
&\left\| \mathbf{a}_{v,l_t+1,t} - \mathbf{a}_0 \right\|_2 \\
\leq &\sum_{l'_t=0}^{l_t} \eta (1 - \frac{\eta\lambda_0}{2})^{l'_t/2} \frac{1}{4} \eta\lambda_0 R' \sqrt{m}.
\end{aligned}
$$

Now suppose the hypothesis hold for t'=0,1..,t and for $0 \leq l_t \leq \ell$. We want to prove for $t' = t + 1$, the hypothesis holds. By Lemma B.7, we know $\left\| \mathcal{W}^{(h)}_t - \mathcal{W}^{(h)}_0 \right\|_F \leq \sqrt{m} R'$ Thus, $\left\| \mathbf{W}^{(h)}_{v,l_t=0,t+1} - \mathcal{W}^{(h)}_0 \right\|_F \leq \sqrt{m} R'$ Thus, by using the same induction on l_t above, we can prove the hypothesis for $t + 1$.

□

Lemma B.7. *Assume*

$$
\left\| \mathbf{W}^{(h)}_{v,l_t,t} - \mathcal{W}^{(h)}_0 \right\|_F, \left\| \mathbf{a}_{v,l_t,t} - \mathbf{a}_0 \right\|_2 \leq \sqrt{m} R'
$$

We have

$$
\left\| \mathcal{W}^{(h)}_t - \mathcal{W}^{(h)}_0 \right\|_F, \left\| \mathbf{a}_t - \mathbf{a}_0 \right\|_2 \leq \sqrt{m} R'
$$

Proof of Lemma B.7.

$$
\begin{aligned}
\left\| \mathcal{W}^{(h)}_t - \mathcal{W}^{(h)}_0 \right\|_F &= \left\| \frac{\sum_{v=1}^{S} \mathbf{W}^{(h)}_{v,\ell,t} M^{(h)}_{v,t}}{\sum_{v=1}^{S} M^{(h)}_{v,t}} - \mathcal{W}^{(h)}_0 \right\|_F \\
&\leq \frac{\sum_{v: M^{(h)}_{v,t}=1} \left\| \mathbf{W}^{(h)}_{v,\ell,t} - \mathcal{W}^{(h)}_0 \right\|_F}{\sum_{v=1}^{S} M^{(h)}_{v,t}} \\
&\leq \sqrt{m} R'
\end{aligned}
$$

Similarly,

$$
\begin{aligned}
\left\| \mathbf{a}_t - \mathbf{a}_0 \right\|_2 &\leq \frac{\sum_{v=1}^{S} \left\| \mathbf{a}_{v,l_t,t} - \mathbf{a}_0 \right\|_2}{S} \\
&\leq \sqrt{m} R'
\end{aligned}
$$

□

Lemma B.8. *If Condition B.1 holds for $t' = 0, \ldots, t-1$ and $\eta \leq c\lambda_0 H^2 n^{-2} \ell^{-2} S^{-1}$ for some small constant c, we have $\left\| \mathbf{I'}_1^i(t) - \mathbf{I}_1^i(t) \right\|_2 \leq C_{I_1}^* \eta^2 \left\| y_i - \hat{u}_{i,t-1} \right\|_2$ where $C_{I_1}^*$ is a constant and thus $\left\| \mathbf{I'}_1(t) - \mathbf{I}_1(t) \right\|_2 \leq \frac{1}{16} \eta \lambda_0 \left\| \mathbf{y} - \hat{\mathbf{u}}(k) \right\|_2$.*

Proof of Lemma B.8.

$$
\begin{aligned}
\left\| \mathbf{I'}_1^i(t) - \mathbf{I}_1^i(t) \right\|_2 &= \left\| \langle \eta \ell L'(\theta(t)) - (\theta(t+1) - \theta(t)), \hat{u}_i'(\theta(t)) \rangle \right\|_2 \\
&\leq \sum_{h=1}^{H} \left\| \eta \frac{\sum_{v=1}^{S} \sum_{l_t=1}^{\ell} \frac{\partial L}{\partial \mathbf{W}_{v,l_t,t}^{(h)}}}{\sum_{v=1}^{S} M_{v,t}^{(h)}} - \eta \ell \frac{\partial L}{\partial \mathcal{W}_{t-1}^{(h)}} \right\|_F \left\| \hat{u}_i'(\theta(t) \right\|_2 \\
&+ \left\| \eta \frac{\sum_{v=1}^{S} \sum_{l_t=1}^{\ell} \frac{\partial L}{\partial \mathbf{a}_{v,l_t,t}}}{S} - \eta \ell \frac{\partial L}{\partial \mathbf{a}_t} \right\|_2 \left\| \hat{u}_i'(\theta(t) \right\|_2
\end{aligned}
$$

Let $M_{t,h} = \sum_{v=1}^{S} M_{v,t}^{(h)}$

$$
\begin{aligned}
&\left\| \eta \frac{\sum_{v=1}^{S} \sum_{l_t=1}^{\ell} \frac{\partial L}{\partial \mathbf{W}_{v,l_t,t}^{(h)}}}{\sum_{v=1}^{S} M_{v,t}^{(h)}} - \eta \ell \frac{\partial L}{\partial \mathcal{W}_{t-1}^{(h)}} \right\|_F \\
\leq & \eta 1/M_{t,h} \sum_{v=1}^{S} \sum_{l_t=1}^{\ell} \left\| \frac{\partial L}{\partial \mathbf{W}_{v,l_t,t}^{(h)}} - \frac{\partial L}{\partial \mathcal{W}_{t-1}^{(h)}} \right\|_F \\
\leq & \eta 1/M_{t,h} \sum_{v=1}^{S} \sum_{l_t=1}^{\ell} || \frac{c_{res}}{H\sqrt{m}} \sum_{i=1}^{n} (y_i - u_{i,v,l_t,t}) \mathbf{x}_{i,v,l_t,t}^{(h-1)} \cdot \left[\mathbf{a}_{v,l_t,t}^{\top} \prod_{l=h+1}^{H} \left(\mathbf{I} + \frac{c_{res}}{H\sqrt{m}} \mathbf{J}_{i,v,l_t,t}^{(l)} \mathbf{W}_{v,l_t,t}^{(l)} M_{v,t}^{(l)} \right) \mathbf{J}_{i,v,l_t,t}^{(h)} M_{v,t}^{(h)} \right] \\
& - \frac{c_{res}}{H\sqrt{m}} \sum_{i=1}^{n} (y_i - \hat{u}_{i,t-1}) \mathbf{x}_{i,t-1}^{(h-1)} \cdot \left[\mathbf{a}_{t-1}^{\top} \prod_{l=h+1}^{H} \left(\mathbf{I} + \frac{c_{res}}{H\sqrt{m}} \mathbf{J}_{i,t-1}^{(l)} \mathcal{W}_{t-1}^{(l)} \right) \mathbf{J}_{i,t-1}^{(h)} \right] ||_F \\
\leq & \eta 1/M_{t,h} \sum_{v=1}^{S} \sum_{l_t=1}^{\ell} \frac{c_{res}}{H\sqrt{m}} \sum_{i=1}^{n} || (y_i - u_{i,v,l_t,t}) \mathbf{x}_{i,v,l_t,t}^{(h-1)} \cdot \left[\mathbf{a}_{v,l_t,t}^{\top} \prod_{l=h+1}^{H} \left(\mathbf{I} + \frac{c_{res}}{H\sqrt{m}} \mathbf{J}_{i,v,l_t,t}^{(l)} \mathbf{W}_{v,l_t,t}^{(l)} M_{v,t}^{(l)} \right) \mathbf{J}_{i,v,l_t,t}^{(h)} M_{v,t}^{(h)} \right] \\
& - (y_i - \hat{u}_{i,t-1}) \mathbf{x}_{i,t-1}^{(h-1)} \cdot \left[\mathbf{a}_{t-1}^{\top} \prod_{l=h+1}^{H} \left(\mathbf{I} + \frac{c_{res}}{H\sqrt{m}} \mathbf{J}_{i,t-1}^{(l)} \mathcal{W}_{t-1}^{(l)} \right) \mathbf{J}_{i,t-1}^{(h)} \right] ||_F
\end{aligned}
$$

Through standard calculations, we have

$$
\begin{aligned}
\left\| \mathcal{W}_{t-1}^{(l)} - \mathbf{W}_{v,l_t,t}^{(l)} \right\|_F &\leq \eta \ell Q'(0,t), \\
\left\| \mathbf{a}_{t-1} - \mathbf{a}_{v,l_t,t} \right\|_F &\leq \eta \ell Q'(0,t), \\
\left\| \mathbf{x}_{i,t-1}^{(h-1)} - \mathbf{x}_{i,v,l_t,t}^{(h-1)} \right\|_F &\leq \eta \ell c_x' \frac{Q'(0,t)}{\sqrt{m}}, \\
\left\| \mathbf{J}_{i,t-1}^{(l)} - \mathbf{J}_{i,v,l_t,t}^{(l)} \right\|_F &\leq 2\ell \left(c_{x,0} + c_{w,0} c_x' \right) \eta \beta Q'(0,t),
\end{aligned}
$$

where $c_x' \triangleq \left(\sqrt{c_\sigma} L + \frac{c_{x,0}}{c_{w,0}} + \frac{c_{x,0}}{R} \right) e^{2 c_{res} c_{w,0} L}$. As we know $\left\| y_i - u_{i,v,l_t,t} \right\| \leq \left\| y_i - \hat{u}_{i,t-1} \right\|$, suppose $\left\| u_{i,v,l_t,t} - \hat{u}_{i,t-1} \right\| \leq C_u$

According to Lemma G.1 in Du et al. [2019], we have

$$
\eta 1/M_{t,h} \sum_{v=1}^{S} \sum_{l_t=1}^{\ell} \frac{c_{res}}{H\sqrt{m}} \sum_{i=1}^{n} || (y_i - u_{i,v,l_t,t}) \mathbf{x}_{i,v,l_t,t}^{(h-1)} \cdot \left[\mathbf{a}_{v,l_t,t}^{\top} \prod_{l=h+1}^{H} \left(\mathbf{I} + \frac{c_{res}}{H\sqrt{m}} \mathbf{J}_{i,v,l_t,t}^{(l)} \mathbf{W}_{v,l_t,t}^{(l)} M_{v,t}^{(l)} \right) \mathbf{J}_{i,v,l_t,t}^{(h)} M_{v,t}^{(h)} \right]
$$

$$
\begin{aligned}
&- (y_i - \hat{u}_{i,t-1})\mathbf{x}_{i,t-1}^{(h-1)} \cdot \left[\mathbf{a}_{t-1}^\top \prod_{l=h+1}^{H} \left(\mathbf{I} + \frac{c_{res}}{H\sqrt{m}}\mathbf{J}_{i,t-1}^{(l)}\mathcal{W}_{t-1}^{(l)}\right)\mathbf{J}_{i,t-1}^{(h)}\right] ||_F \\
\leq & \eta 1/M_{t,h} S\ell n \frac{4}{H} c_{res} c_{x,0} L a_{2,0} e^{2Lc_{w,0}} (C_u \\
&+ \eta\ell \frac{Q'(0,t)}{\sqrt{m}} \left(\frac{c_x}{c_{x,0}} + \frac{2}{L}\left(c_{x,0} + c_{w,0}c_x\right)\beta\sqrt{m} + 4c_{w,0}\left(c_{x,0} + c_{w,0}c_x\right)\beta + L + 1\right)) \left\|y_i - \hat{u}_{i,t-1}\right\|_2
\end{aligned}
$$

On the other hand,

$$
\begin{aligned}
\left\|\eta \frac{\sum_{v=1}^{S}\sum_{l_t=1}^{\ell} \frac{\partial L}{\partial \mathbf{a}_{v,l_t,t}}}{S} - \eta\ell \frac{\partial L}{\partial \mathbf{a}_t}\right\|_2 &\leq \eta 1/S \sum_{v=1}^{S}\sum_{l_t=1}^{\ell} \left\|\frac{\partial L}{\partial \mathbf{a}_{v,l_t,t}} - \frac{\partial L}{\partial \mathbf{a}_t}\right\|_2 \\
&\leq \eta 1/S \sum_{v=1}^{S}\sum_{l_t=1}^{\ell}\sum_{i=1}^{n} \left\|(y_i - u_{i,v,l_t,t})\mathbf{x}_{i,v,l_t,t}^{(H)} - (y_i - \hat{u}_{i,t})\mathbf{x}_{i,t-1}^{(H)}\right\|_2 \\
&\leq \eta\ell n (C_u + \eta\ell c_x' \frac{Q'(0,t)}{\sqrt{m}}) \left\|y_i - \hat{u}_{i,t-1}\right\|_2
\end{aligned}
$$

Also,

$$
\begin{aligned}
\left\|\hat{u}_i'\left(\theta(t)\right)\right\|_2 &\leq \frac{c_{res}}{H\sqrt{m}} \sum_{h=1}^{H} \left\|\frac{\partial \hat{u}_i\left(\theta(t)\right)}{\partial \mathcal{W}_{t-1}^{(h)}}\right\|_2 \\
&= \frac{c_{res}}{H\sqrt{m}} \sum_{h=1}^{H} \left\|\mathbf{x}_{i,t-1}^{(h-1)} \cdot \left[\mathbf{a}_{t-1}^\top \prod_{l=h+1}^{H} \left(\mathbf{I} + \frac{c_{res}}{H\sqrt{m}}\mathbf{J}_{i,t-1}^{(l)}\mathcal{W}_{t-1}^{(l)}\right)\mathbf{J}_{i,t-1}^{(h)}\right]\right\|_2 \\
&\leq \frac{c_{res}}{H\sqrt{m}} \sum_{h=1}^{H} \left\|\mathbf{x}_{i,t-1}^{(h-1)}\right\|_2 \left\|\mathbf{a}_{t-1}\right\|_2 \left\|\prod_{l=h+1}^{H} \left(\mathbf{I} + \frac{c_{res}}{H\sqrt{m}}\mathbf{J}_{i,t-1}^{(l)}\mathcal{W}_{t-1}^{(l)}\right)\right\|_2 \left\|\mathbf{J}_{i,t-1}^{(h)}\right\|_2 \\
&\leq \frac{c_{res}}{H} H 2c_{x,0}a_{2,0}Le^{2c_{res}x_{w,0}L} \\
&= 2c_{res}c_{x,0}a_{2,0}Le^{2c_{res}x_{w,0}L}
\end{aligned}
$$

Thus, combine all above and also according to Lemma B.9

$$
\begin{aligned}
\left\|\mathbf{I}'^i_1(t) - \mathbf{I}^i_1(t)\right\|_2 \leq \sum_{h=1}^{H} &\left\|\eta \frac{\sum_{v=1}^{S}\sum_{l_t=1}^{\ell} \frac{\partial L}{\partial \mathbf{W}_{v,l_t,t}^{(h)}}}{\sum_{v=1}^{S} M_{v,t}^{(h)}} - \eta\ell \frac{\partial L}{\partial \mathcal{W}_{t-1}^{(h)}}\right\|_2 \left\|\hat{u}_i'\left(\theta(t)\right)\right\|_2 \\
&+ \left\|\eta \frac{\sum_{v=1}^{S}\sum_{l_t=1}^{\ell} \frac{\partial L}{\partial \mathbf{a}_{v,l_t,t}}}{S} - \eta\ell \frac{\partial L}{\partial \mathbf{a}_t}\right\|_2 \left\|\hat{u}_i'\left(\theta(t)\right)\right\|_2 \\
&\leq C_{I_1}^* \eta^2 \left\|y_i - \hat{u}_{i,t-1}\right\| \text{ where } C_{I_1}^* \text{ is a constant}
\end{aligned}
$$

Using the bound on η and following Du et al. [2019] $\left\|\mathbf{y} - \hat{\mathbf{u}}\right\|_2 = O(\sqrt{n})$,

$$
\left\|\mathbf{I}'_1(t) - \mathbf{I}_1(t)\right\| \leq \frac{1}{16}\eta\lambda_0 \left\|\mathbf{y} - \hat{\mathbf{u}}(k)\right\|_2
$$

□

Lemma B.9.

$$
\left\|u_{i,v,l_t,t} - \hat{u}_{i,t}\right\|_2 \leq \eta\ell Q'(0,t)B \textit{ where B is a constant}
$$

Proof of Lemma B.9.

$$\|u_{i,v,l_t,t} - \hat{u}_{i,t}\|_2 = \left\|\mathbf{a}_{v,l_t,t}^\top \mathbf{x}_{v,l_t,t}^{(H)} - \mathbf{a}_t^\top \mathbf{x}_t^{(H)}\right\|_2$$
$$\leq \eta(2a_{2,0}3c_{x,0}\ell Q'(0,t)(1+\frac{c_x}{\sqrt{m}}))$$

□

Lemma B.10. *If Condition B.1 holds for $t' = 0, \ldots, t-1$ and $\eta \leq c\lambda_0 H^2 n^{-2}\ell^{-2}S^{-1}$ for some small constant c, we have $\|\mathbf{I}_2(t)\|_2 \leq C_{I_2}^* \eta^2 \|y_i - \hat{u}_{i,t-1}\|_2$ where $C_{I_2}^*$ is a constant and thus $\|\mathbf{I}_2(t)\|_2 \leq \frac{1}{8}\eta\lambda_0 \|\mathbf{y} - \hat{\mathbf{u}}(k)\|_2$.*

Proof of Leamma B.10.

$$I_2^i(t) = \int_{s=0}^1 \langle \theta(t+1) - \theta(t), \hat{u}_i'(\theta(t)) - \hat{u}_i'(\theta(t) - s(\theta(t) - \theta(t+1)))\rangle ds$$

Define for $1 \leq h \leq H$

$$\hat{u}_i'^{(h)}(\theta(t)) = \frac{\partial \hat{u}(\theta(t))}{\mathcal{W}_t^{(h)}}$$

And

$$\hat{u}_i'^{(H+1)}(\theta(t)) = \frac{\partial \hat{u}(\theta(t))}{\mathbf{a}_t}$$

$$\left|I_2^i(t)\right| \leq \max_{0\leq s\leq 1} \sum_{h=1}^H \left\|\mathcal{W}_t^{(h)} - \mathcal{W}_{t-1}^{(h)}\right\|_F \left\|\hat{u}_i'^{(h)}(\theta(t)) - \hat{u}_i'^{(h)}(\theta(t) - s(\theta(t+1) - \theta(t)))\right\|_F$$
$$+ \|\mathbf{a}_t - \mathbf{a}_{t-1}\|_2 \left\|\hat{u}_i'^{(H+1)}(\theta(t)) - \hat{u}_i'^{(H+1)}(\theta(t) - s(\theta(t+1) - \theta(t)))\right\|_2.$$

From Lemma B.8 and Lemma B.6,

$$\left\|\mathcal{W}_t^{(h)} - \mathcal{W}_{t-1}^{(h)}\right\|_F \leq \eta\ell\hat{Q}'(t-1)$$
$$\|\mathbf{a}_t - \mathbf{a}_{t-1}\|_2 \leq \eta\ell\hat{Q}'(t-1)$$

Let $\mathbf{x}_{i,t-1,s}^{(l)}$ be the activation of global network with $\mathcal{W}_{t-1,s} = \mathcal{W}_{t-1} - s(\mathcal{W}_{t-1} - \mathcal{W}_t)$. We similarly define $\mathbf{J}_{i,t-1,s}^{(l)}$ and $\mathbf{a}_{t-1,s}$

$$\left\|\hat{u}_i'^{(h)}(\theta(t)) - \hat{u}_i'^{(h)}(\theta(t) - s(\mathcal{W}_{t-1} - \mathcal{W}_t))\right\|_F$$
$$\leq \frac{c_{res}}{H\sqrt{m}} \Big|\Big|\mathbf{x}_{i,t-1,s}^{(h-1)} \cdot \left[\mathbf{a}_{t-1,s}^\top \prod_{l=h+1}^H \left(\mathbf{I} + \frac{c_{res}}{H\sqrt{m}}\mathbf{J}_{i,t-1,s}^{(l)}\mathcal{W}_{t-1,s}^{(l)}\right)\mathbf{J}_{i,t-1,s}^{(h)}\right]$$
$$- \mathbf{x}_{i,t-1}^{(h-1)} \cdot \left[\mathbf{a}_{t-1}^\top \prod_{l=h+1}^H \left(\mathbf{I} + \frac{c_{res}}{H\sqrt{m}}\mathbf{J}_{i,t-1}^{(l)}\mathcal{W}_{t-1}^{(l)}\right)\mathbf{J}_{i,t-1}^{(h)}\right]\Big|\Big|_F$$

Through similar calculation in Lemma B.8,

$$\left\|\mathcal{W}_{t-1,s}^{(l)} - \mathcal{W}_{t-1}^{(l)}\right\|_F = s\left\|(\mathcal{W}_{t-1}^{(l)} - \mathcal{W}_t^{(l)})\right\|_F$$
$$\leq \left\|(\mathcal{W}_{t-1}^{(l)} - \mathcal{W}_t^{(l)})\right\|_F$$
$$\leq \eta\ell\hat{Q}'(t-1)$$

$$
\begin{aligned}
\left\|\mathbf{x}_{i,t-1,s}^{(l)}-\mathbf{x}_{i,t-1}^{(l)}\right\|_2 &\leq \frac{c_{res}}{H\sqrt{m}}\left\|\sigma\left(\mathcal{W}_{t-1,s}^{(l)}\mathbf{x}_{t-1,s}^{(l-1)}\right)-\sigma\left(\mathcal{W}_{t-1}^{(l)}\mathbf{x}_{t-1}^{(l-1)}\right)\right\|_2+\left\|\mathbf{x}_{t-1,s}^{(l-1)}-\mathbf{x}_{t-1}^{(l-1)}\right\|_2 \\
&\leq \frac{c_{res}}{H\sqrt{m}}\left\|\sigma\left(\mathcal{W}_{t-1,s}^{(l)}\mathbf{x}_{t-1,s}^{(l-1)}\right)-\sigma\left(\mathcal{W}_{t-1,s}^{(l)}\mathbf{x}_{t-1}^{(l-1)}\right)\right\|_2 \\
&+\frac{c_{res}}{H\sqrt{m}}\left\|\sigma\left(\mathcal{W}_{t-1,s}^{(l)}\mathbf{x}_{t-1}^{(l-1)}\right)-\sigma\left(\mathcal{W}_{t-1}^{(l)}\mathbf{x}_{t-1}^{(l-1)}\right)\right\|_2+\left\|\mathbf{x}_{t-1,s}^{(l-1)}-\mathbf{x}_{t-1}^{(l-1)}\right\|_2 \\
&\leq \frac{c_{res}L}{H\sqrt{m}}\left\|\mathcal{W}_{t-1,s}^{(l)}\right\|_F\left\|\mathbf{x}_{t-1,s}^{(l-1)}-\mathbf{x}_{t-1}^{(l-1)}\right\|_2 \\
&+\frac{c_{res}L}{H\sqrt{m}}\left\|\mathcal{W}_{t-1,s}^{(l)}-\mathcal{W}_{t-1}^{(l)}\right\|_F\left\|\mathbf{x}_{t-1}^{(l-1)}\right\|_2+\left\|\mathbf{x}_{t-1,s}^{(l-1)}-\mathbf{x}_{t-1}^{(l-1)}\right\|_2 \\
&\leq (1+\frac{c_{res}L}{H\sqrt{m}}(c_{w,0}\sqrt{m}+R'\sqrt{m}+\eta\ell\hat{Q}'(t-1)))\left\|\mathbf{x}_{t-1,s}^{(l-1)}-\mathbf{x}_{t-1}^{(l-1)}\right\|_2 \\
&+\frac{c_{res}L}{H\sqrt{m}}\eta\ell\hat{Q}'(t-1)(c_xR'+c_{x,0})
\end{aligned}
$$

Also

$$
\begin{aligned}
\left\|\mathbf{x}_{i,t-1,s}^{(0)}-\mathbf{x}_{i,t-1}^{(0)}\right\|_2 &= \frac{c_{res}}{H\sqrt{m}}\left\|\sigma\left(\mathcal{W}_{t-1,s}^{(1)}\mathbf{x}_i\right)-\sigma\left(\mathcal{W}_{t-1}^{(0)}\mathbf{x}_i\right)\right\|_2 \\
&\leq \frac{c_{res}}{H\sqrt{m}}L\left\|\mathcal{W}_{t-1,s}^{(1)}-\mathcal{W}_{t-1}^{(0)}\right\|_2 \\
&\leq \frac{c_{res}}{H\sqrt{m}}L\eta\ell\hat{Q}'(t-1)
\end{aligned}
$$

Thus

$$
\begin{aligned}
\left\|\mathbf{x}_{i,t-1,s}^{(l)}-\mathbf{x}_{i,t-1}^{(l)}\right\|_2 &\leq (\frac{c_{res}}{H\sqrt{m}}L\eta\ell\hat{Q}'(t-1)+\frac{\frac{c_{res}L}{H\sqrt{m}}\eta\ell\hat{Q}'(t-1)(c_xR'+c_{x,0})}{\frac{c_{res}L}{H\sqrt{m}}(c_{w,0}\sqrt{m}+R'\sqrt{m})})e^{\frac{c_{res}L}{H\sqrt{m}}(c_{w,0}\sqrt{m}+R'\sqrt{m}+\eta\ell\hat{Q}'(t-1))} \\
&\leq \eta\ell\hat{Q}'(t-1)(\frac{c_{res}}{H\sqrt{m}}L+\frac{(c_xR'+c_{x,0})}{(c_{w,0}\sqrt{m}+R'\sqrt{m})})e^{\frac{c_{res}L}{\sqrt{m}}(c_{w,0}\sqrt{m}+R'\sqrt{m}+\eta\ell\hat{Q}'(t-1))} \\
&\triangleq \eta\ell\hat{Q}'(t-1)C_x^*
\end{aligned}
$$

Similarly, through standard calculation we can get

$$
\|\mathbf{a}_{t-1,s}-\mathbf{a}_{t-1}\|_2 \leq \eta\ell\hat{Q}'(t-1)
$$

Lastly,

$$
\begin{aligned}
\left\|\mathbf{J}_{i,t-1,s}^{(l)}-\mathbf{J}_{i,t-1}^{(l)}\right\|_2 &= \left\|\sigma'(\mathcal{W}_{t-1,s}^{(l)}\mathbf{x}_{t-1,s}^{(l-1)})-\sigma'(\mathcal{W}_{t-1}^{(l)}\mathbf{x}_{t-1}^{(l-1)})\right\|_2 \\
&\leq \beta\left\|\mathcal{W}_{t-1,s}^{(l)}\mathbf{x}_{t-1,s}^{(l-1)}-\mathcal{W}_{t-1}^{(l)}\mathbf{x}_{t-1}^{(l-1)}\right\|_2 \\
&\leq \beta(\left\|\mathcal{W}_{t-1,s}^{(l)}\right\|_F\left\|\mathbf{x}_{t-1,s}^{(l-1)}-\mathbf{x}_{t-1}^{(l-1)}\right\|_2+\left\|\mathcal{W}_{t-1,s}^{(l)}-\mathcal{W}_{t-1}^{(l)}\right\|_F\left\|\mathbf{x}_{t-1}^{(l-1)}\right\|_2) \\
&\leq \beta(\frac{c_{res}L}{H\sqrt{m}}(c_{w,0}\sqrt{m}+R'\sqrt{m}+\eta\ell\hat{Q}'(t-1))\eta\ell\hat{Q}'(t-1)C_x^*+\eta\ell\hat{Q}'(t-1)(c_xR'+c_{x,0})) \\
&= \eta\ell\hat{Q}'(t-1)\beta(\frac{c_{res}L}{H\sqrt{m}}(c_{w,0}\sqrt{m}+R'\sqrt{m}+\eta\ell\hat{Q}'(t-1))C_x^*+(c_xR'+c_{x,0})) \\
&\triangleq \eta\ell\hat{Q}'(t-1)\beta C_J^*
\end{aligned}
$$

Thus, according to Lemma G.1 in Du et al. [2019], we have

$$
\begin{aligned}
&\left\|\hat{u}_i'^{(h)}\left(\theta(t)\right)-\hat{u}_i'^{(h)}\left(\theta(t)-s(\mathcal{W}_{t-1}-\mathcal{W}_t)\right)\right\|_F \\
\leq &\frac{c_{res}}{H\sqrt{m}}||\mathbf{x}_{i,t-1,s}^{(h-1)}\cdot\left[\mathbf{a}_{t-1,s}^{\top}\prod_{l=h+1}^{H}\left(\mathbf{I}+\frac{c_{res}}{H\sqrt{m}}\mathbf{J}_{i,t-1,s}^{(l)}\mathcal{W}_{t-1,s}^{(l)}\right)\mathbf{J}_{i,t-1,s}^{(h)}\right]
\end{aligned}
$$

$$
\begin{aligned}
&- \mathbf{x}_{i,t-1}^{(h-1)} \cdot \left[\mathbf{a}_{t-1}^\top \prod_{l=h+1}^{H} \left(\mathbf{I} + \frac{c_{res}}{H\sqrt{m}} \mathbf{J}_{i,t-1}^{(l)} \mathcal{W}_{t-1}^{(l)} \right) \mathbf{J}_{i,t-1}^{(h)} \right] || \\
&\leq \eta \ell \hat{Q}'(t-1) \frac{c_{res}}{H\sqrt{m}} 2c_{x,0} 2a_{2,0} L e^{2Lc_{w,0}} \left(\frac{C_x^*}{2c_{x,0}} + \frac{1}{2a_{2,0}} + \frac{c_{res}}{\sqrt{m}} \beta C_J^* \right)
\end{aligned}
$$

On the other hand

$$
\begin{aligned}
\left\| \hat{u}_i'^{(H+1)} \left(\theta(t)\right) - \hat{u}_i'^{(H+1)} \left(\theta(t) - s(\theta(t+1) - \theta(t))\right) \right\|_2 &\leq \left\| \mathbf{x}_{t-1,s}^{(H)} - \mathbf{x}_{t-1}^{(H)} \right\|_2 \\
&\leq \eta \ell \hat{Q}'(t-1) C_x^*
\end{aligned}
$$

In the end,

$$
\begin{aligned}
\|\mathbf{I}_2(t)\|_2 &\leq \eta \ell \hat{Q}'(t-1) \eta \ell \hat{Q}'(t-1) \left(\frac{c_{res}}{\sqrt{m}} 2c_{x,0} 2a_{2,0} L e^{2Lc_{w,0}} \left(\frac{C_x^*}{2c_{x,0}} + \frac{1}{2a_{2,0}} + \frac{c_{res}}{\sqrt{m}} \beta C_J^* \right) + C_x^* \right) \\
&\leq \eta^2 \ell^2 \hat{Q}'(t-1)^2 \left(\frac{c_{res}}{\sqrt{m}} 2c_{x,0} 2a_{2,0} L e^{2Lc_{w,0}} \left(\frac{C_x^*}{2c_{x,0}} + \frac{1}{2a_{2,0}} + \frac{c_{res}}{\sqrt{m}} \beta C_J^* \right) + C_x^* \right) \\
&\leq \eta^2 C_{I_2}^* \|\mathbf{y} - \hat{\mathbf{u}}(t)\|_2 \\
&\leq \frac{1}{16} \eta \lambda_0 \|\mathbf{y} - \hat{\mathbf{u}}(t)\|_2
\end{aligned}
$$

□

Lemma B.11. *If Condition B.1 holds for $t' = 0, \ldots, t-1$ and $\eta \leq c\lambda_0 H^2 n^{-2} \ell^{-2} S^{-1}$ for some small constant c, we have $\|\hat{\mathbf{u}}(t+1) - \hat{\mathbf{u}}(t)\|_2^2 \leq \frac{1}{16} \eta \lambda_0 \|\mathbf{y} - \hat{\mathbf{u}}(k)\|_2^2$.*

Proof of Lemma B.11.

$$
\begin{aligned}
\|\hat{\mathbf{u}}(t+1) - \hat{\mathbf{u}}(t)\|_2^2 &= \sum_{i=1}^{n} \left(\mathbf{a}_{t+1}^\top \mathbf{x}_{i,t+1}^{(H)} - \mathbf{a}_t^\top \mathbf{x}_{i,t}^{(H)} \right)^2 \\
&= \sum_{i=1}^{n} \left(\left[\mathbf{a}_{t+1} - \mathbf{a}_t \right]^\top \mathbf{x}_{i,t+1}^{(H)} + \mathbf{a}_t^\top \left[\mathbf{x}_{i,t+1}^{(H)} - \mathbf{x}_{i,t}^{(H)} \right] \right)^2 \\
&\leq 2 \|\mathbf{a}_{t+1} - \mathbf{a}_t\|_2^2 \sum_{i=1}^{n} \left\| \mathbf{x}_{i,t+1}^{(H)} \right\|_2^2 + 2 \|\mathbf{a}_t\|_2^2 \sum_{i=1}^{n} \left\| \mathbf{x}_{i,t+1}^{(H)} - \mathbf{x}_{i,t}^{(H)} \right\|_2^2 \\
&\leq 18 n \eta^2 \ell^2 c_{x,0}^2 Q'(t)^2 + 4n \left(\eta \ell a_{2,0} c_x Q'(t) \right)^2 \\
&\leq \frac{1}{8} \eta \lambda_0 \|\mathbf{y} - \hat{\mathbf{u}}(t)\|_2^2 .
\end{aligned}
$$

□