Optimal Topology Search for Fast Model Averaging in Decentralized Parallel SGD

Mohsan Jameel, Shayan Jawed, and Lars Schmidt-Thieme

University of Hildesheim, Universittsplatz 1, 31141 Hildesheim, Germany
{mohsan.jameel,shayan,schmidt-thieme}@ismll.uni-hildesheim.de

Abstract. Distributed training of deep learning models on high-latency systems necessitates decentralized parallel SGD solutions. However, existing solutions suffer from slow convergence because of hand-crafted topologies. The question arises, "*for decentralized parallel SGD, is it possible to learn a topology that provides faster model averaging compared to the hand-crafted counterparts?*".
By leveraging spectral properties of the graph, we formulate the objective function for finding the topology that provides fast model averaging. Since direct optimization of the objective function is infeasible, we employ a local search algorithm guided by the objective function. We show through extensive empirical evaluation on image classification tasks that the model averaging based on learned topologies leads to fast convergence. An equally important aspect of the decentralized parallel SGD is the link weights for sparse model averaging. In contrast to setting weights via Metropolis-Hastings, we propose to use Laplacian link weights on the learned topologies, which provide a significant lift in performance.

Keywords: Optimal Decentralize Topology · Fast Model Averaging · Parallel Stochastic Gradient Decent · Deep Learning

1 Introduction

The current trend in learning distributed models on edge devices (factory controllers etc) is gaining traction as it eases the burden of transferring data to a central location, which in a real-world scenario, is limited by slow transfer rates or legal restrictions [13]. The decentralized nature of these devices coupled with high latency networks renders the centralized parallel SGD [21,8] approaches infeasible due to high aggregation cost at the central node. Hence, it is imperative to design distributed algorithms that are well suited for these high latency systems. Decentralized learning algorithms [9,10] operate on local hosts using only the local data partitions but to reduce communication cost, each worker shares information with a small subset of neighbors. As a result, the central bottleneck is effectively eliminated. An example is the hand-crafted virtual ring topology, where each worker only communicates with its two adjacent neighbors.

However, these hand-crafted solutions exhibit slow model averaging properties, which lead to high variance among the individual models learned by workers [20,16]. For training highly non-linear models with non-convex optimization

objectives, this could be problematic as high variance degrades convergence behavior and leads to suboptimal solutions. The fundamental question arises, "*for decentralized parallel SGD, is it possible to learn a topology that provides faster model averaging compared to the hand-crafted counterparts?*". It is well known that the spectral properties of the graph provide a good metrics to measure the connectivity structure of the topology [14]. Based on this, we propose an objective function, optimization of which leads to the topology that provides fast model averaging under constraints such as the number of connections. The direct optimization of the objective function leads to a combinatorial explosion of the search space, thus finding an optimal solution under constraints is intractable. The solution [12] that exists is limited to prime and prime power of the number of direct connections, which hinders its real-world applicability. In this paper, we propose a local search algorithm guided by the objective function to find a topology that leads to fast model averaging.

The topology itself is an incomplete solution for fast model averaging without giving due importance to optimal averaging link weights. The common practice to set weights so far has been based on the Metropolis-Hastings algorithm [14]. However, as we shall show these are not generalizable across learned topologies. We propose to exploit spectral properties of the graph to assign link weights for model averaging, which have been proven to be the optimal choice for constant edge weights [18].

To recap, our contributions are:

- We propose a principled method to find a topology that enables fast model averaging under communication constraints.
- We demonstrate improved convergence behavior with Laplacian link weights in contrast to weights set by the Metropolis-Hastings algorithm.
- We evaluate on a set of image classification tasks that model averaging solutions based on our method converges faster than hand-crafted counterparts.

2 Related Work

Training machine learning models in a distributed setting is a widely studied topic, and becoming more challenging with the increase in data and model complexity. Literature [21,8,3] on centralized approaches relies on a central node for aggregating the updates from distributed workers. Lian et al. [9] have shown that in high latency systems, the central node becomes a bottleneck due to high communication cost. To eliminate this central bottleneck, they proposed a decentralized model averaging scheme, where each worker performs model averaging by communicating with their adjacent neighbors. Their solutions [9,10] use variants of a structured topology based on a Regular Ring Lattice (RRL) graph and link weights are set using the Metropolis-Hastings algorithm [14]. However, these handcrafted topologies exhibit poor spectral properties of the graph, which lead to slow model averaging and degradation in the convergence behavior. Wang et al. [16] through a unified analysis of centralized and decentralized approaches, show that the averaging delay and sparse connectivity can adversely affect the convergence behavior.

The main reason for slow convergence of the decentralized model averaging is the consequence of a poor choice of topologies and link weights. The choice of setting link weights for a fixed graph topology are extensively studied in [2,18,11]. The convergence is guaranteed, if the averaging matrix is a doubly stochastic transition matrix and has an eigenvalue of 1 and all other eigenvalues are within a unit circle. Furthermore, convergence is inversely proportional to the magnitude of the second largest eigenvalue [1]. Xiao and Boyd [18] showed for an unweighted graph, the best constant link weights are set based on the spectral properties of the graph. Taking it further, they defined an optimization procedure for finding optimal link weights for fast model averaging.

The literature for finding optimal link weights assumes that the graph topology is already defined, but does not explore the possibility of finding it. In this direction, Kar et al [7] proposed methods for finding 'the regular Ramanujan Graphs', which are known to exhibit better spectral properties but are limited to prime or prime power values of the number of direct connections. To overcome this limitation they provided the R3L method to find a random regular like Ramanujan Graphs by random rewiring of the edges. Their solution has similar principles to a random rewiring approach of Watts and Strogatz's model (WS) [17] to construct small-world graphs, but with better spectral properties. Despite providing solutions superior to RRL approaches, both methods are not studied in the context of decentralized parallel SGD. R3L and WS are based on random edge rewiring heuristics and are not guided by the objective function.

In this paper, we define an optimization objective function, which can be used to find a topology for fast model averaging in a decentralized parallel SGD. Furthermore, we employed R3L and WS based averaging solutions and compare them with RRL approach currently used in the literature. We also analyze the link weights derived from the graph Laplacian and compare it with the usual choice of Metropolis Hastings.

3 Decentralized Stochastic Gradient Descent as Average Consensus Problem

In supervised machine learning, the training dataset $\mathcal{D}$ consists of M training instances each represented by a tuple $(\mathbf{x}, y)$, where $\mathbf{x}$ is a feature vector and y the corresponding label. A model $\hat{y}(\mathbf{x};\theta)$, with model parameters $\theta \in \mathbb{R}^K$ can be learned by minimizing the objective function,

$$\theta := \underset{\theta}{\operatorname{argmin}}\ \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{P}_{\mathcal{D}}(\mathbf{x},y)}[\mathcal{L}(y, \hat{y}(\mathbf{x};\theta))] \tag{1}$$

where $\mathcal{L}(y, \hat{y}(\mathbf{x};\theta))$ is a loss function. Typical loss functions include the cross entropy loss, squared loss, hinge loss etc. Stochastic Gradient Descent based algorithms are most widely used to optimize the objective function in Eq.(1). In a decentralized SGD, distributed workers $\mathcal{V} = \{1, \cdots, N\}$ represent computing resources that hold local partitions of data $\mathcal{D}$. Each worker holds a local copy θ_u and calculates the gradient in Eq.(2) by sampling a mini-batch $\mathcal{B}_u \subset \mathcal{D}_u$ from

the local data partition $\mathcal{D}_u$.

$$\mathbf{g}_u^t = \frac{1}{|\mathcal{B}_u|} \sum_{(\mathbf{x},y)\in\mathcal{B}_u} \nabla\mathcal{L}(y, \hat{y}(\mathbf{x}; \theta_u^t)) \tag{2}$$

The connectivity between workers is defined by an undirected graph topology $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with an edge set of the unordered pairs $\{u, v\} \in \mathcal{E}$. An edge means that a pair of workers u and v can exchange information. The neighborhood of each worker u is a set $\mathcal{N}_u = \{v : \{u, v\} \in \mathcal{E}, v \in \mathcal{V}\}$ and its degree, $\deg(u)$ is the total number of its neighbors i.e. $\deg(u) = |\mathcal{N}_u|$. To update the local model, each worker first averages the local models of the neighborhood and applies the local gradients in Eq.(3).

$$\theta_u^{t+1} = \sum_{v\in\{u\}\cup\mathcal{N}_v} W_{u,v}\theta_v^t \ - \eta\mathbf{g}_v^t \tag{3}$$

where $\mathbf{W} \in \mathbb{R}^{N\times N}$ is called averaging/weighting[1] matrix. Eq.(3) is known as an average consensus problem, and the weight matrix $\mathbf{W}$ encodes the averaging scheme for a sparse connectivity. Convergence of Eq.(3) is possible if $\mathbf{W}$ exhibits the property $\lim_{t\to\infty} \mathbf{W}^t = (1/n)\mathbf{1}\mathbf{1}^T$, i.e. for t steps averaging with $\mathbf{W}$ approaches to a global average [18]. This limit only holds if,

$$\rho(\mathbf{W} - (1/n)\mathbf{1}\mathbf{1}^T) < 1 \tag{4}$$

$$\tau = 1/\log(1/\rho) \tag{5}$$

where $\mathbf{W}$ is a doubly stochastic transition matrix, $\mathbf{1}$ denotes vectors of all ones and $\rho(\cdot)$ is the spectral radius[2] of a matrix. The term $\rho(\mathbf{W} - (1/n)\mathbf{1}\mathbf{1}^T)$ is called the asymptotic convergence factor and Eq.(5) defines its convergence speed. Convergence of Eq.(4) is possible if $\mathbf{W}$ has eigenvalue $\lambda_1(\mathbf{W}) = 1$ and $\lambda_2(\mathbf{W}) \geq \lambda_3(\mathbf{W}) \geq \cdots \geq \lambda_N(\mathbf{W})$ lies in a unit circle. The asymptotic convergence is faster for smaller values of the second largest eigenvalue modulus (SLEM) $\lambda_{\text{slem}}(\mathbf{W}) = \max\{|\lambda_2(\mathbf{W})|, |\lambda_N(\mathbf{W})|\}$ [1].

3.1 Relationship between Network Topology and Averaging Matrix

The convergence of Eq.(3) highly depends on the link weights encoded in $\mathbf{W}$ and the structure of the connectivity of workers defined by graph $\mathcal{G}$. The eigenstructure of the graph Laplacian provides an important tool for studying numerous graph invariant properties including connectivity, expandability, diameter, mean distance and so on [14]. For a given graph $\mathcal{G}$, its Laplacian $\mathbf{L} \in \mathbb{R}^{N\times N}$ is defined as,

$$L_{u,v} = \begin{cases} -1 & \{u, v\} \in \mathcal{E}, \\ deg(u) & u = v, \\ 0 & \text{otherwise}, \end{cases}$$

[1] Averaging matrix, weighting matrix and mixing matrix are interchangeably used to refer to $\mathbf{W}$

[2] $\rho(\mathbf{M}) = \max\{|\lambda|, \lambda \text{ eigenvalue of } \mathbf{M}\}$

For a connected graph the eigenvalues are enumerated as $0 = \lambda_1(\mathbf{L}) < \lambda_2(\mathbf{L}) \leq \cdots \leq \lambda_N(\mathbf{L})$, where the second smallest eigenvalue $\lambda_2(\mathbf{L})$ is known as the algebraic connectivity of the graph [14].

Existing approaches for decentralized parallel SGD set link weights using a simple approach based on Metropolis-Hastings algorithm. Hastings approach [2] utilizes the local degree information of two incident nodes of an edge to assign link weights as,

$$W_{u,v} = \frac{1}{\max\{deg(u), deg(v)\}}, \quad \forall u, v \in \mathcal{V} \tag{6}$$

The graph Laplacian captures global information about the connectivity, and can be used to set the constant edge weights for matrix $\mathbf{W}$ as,

$$\mathbf{W} = \mathbf{I} - \alpha\mathbf{L}, \quad \alpha \in \mathbb{R}^+ \tag{7}$$

The eigenstructure of $\mathbf{W}$ can be expressed as $\lambda_i(\mathbf{W}) = 1 - \alpha\lambda_i(\mathbf{L})$. The convergence in Eq.(4) is possible for α values in the interval $(0, 2/\lambda_2(\mathbf{L}))$ and the best constant value [18] is obtained as,

$$\alpha^* = \frac{2}{\lambda_2(\mathbf{L}) + \lambda_N(\mathbf{L})} \tag{8}$$

The Eq.(8) explicitly relates link weights assignment with the spectral properties of the graph i.e. to $\lambda_2(\mathbf{L})$ and $\lambda_N(\mathbf{L})$. Kar et al. [7] have shown that maximizing the ratio $\frac{\lambda_2(\mathbf{L})}{\lambda_N(\mathbf{L})}$ is equivalent to minimizing $\lambda_{\text{slem}}(\mathbf{W})$.

$$\min \lambda_{\text{slem}}(\mathbf{W}) \sim \max \frac{\lambda_2(\mathbf{L})}{\lambda_N(\mathbf{L})} \tag{9}$$

Now using the fact that the convergence factor in Eq.(4) depends on $\lambda_{\text{slem}}(\mathbf{W})$, therefore Eq.(9) becomes,

$$\min \rho(\mathbf{W} - (1/n)\mathbf{1}\mathbf{1}^T) \sim \max \frac{\lambda_2(\mathbf{L})}{\lambda_N(\mathbf{L})} \tag{10}$$

3.2 Optimization Problem

Let e be the edge set defining the connectivity between workers and using relationship in Eq.(10), we can define the objective function with connectivity constraints as:

$$\max_e \quad f_\rho(e) := \max_e \frac{\lambda_2(\mathbf{L})}{\lambda_N(\mathbf{L})} \tag{11}$$

$$\text{with} \quad \sum_u e_{u,v} \leq C, \quad C \in \mathbb{R}^+ \tag{12}$$

$$\text{over} \quad e_{u,v} \in \{0, 1\} \quad \forall u, v \in \mathcal{V}, u < v \tag{13}$$

where e is the optimization variable, C is the communication budget. Eq.(12) defines the communication constraint that limits the total amount of information exchange at a given averaging step. For a homogeneous network, also focus of this paper, C could imply the total number of connections allowed in a network, whereas, in a heterogeneous case, each communication link has an associated latency and we can bound the total latency in the system. The direct optimization of Eq.(11) under constraints is infeasible as it leads to a combinatorial explosion of the search space.

4 Optimal Network Topology Through Local Search

As we discussed in the previous section, direct optimization of Eq.(11) is infeasible, therefore, in this section we will provide a local search algorithm guided by the objective function to find a topology that provides fast model averaging.

4.1 Constructing Optimal Network Topology using Local Search (LS-R2L)

We initialize the edge set as a regular ring lattice (RRL) topology, which is constructed by taking a cycle and connecting each vertex with its *hops* $= 1, \ldots, \lfloor C/2 \rfloor$ neighbors on the right and left. The resulting graph is connected and satisfies the constraint in Eq.(12). We propose ***local-search-R2L*** **(LS-R2L)** method that uses the local search for rewiring of edges guided by the objective function Eq.(11). At each step, we search the local neighborhood for a set of possible solutions that improve the objective value, and move to that solution. The local search is performed at lines $5-8$ by rewiring a subset of edges and the effect of each change is calculated. Then at line 12, a solution is picked with probability proportional to the improvement in objective values. These steps are repeated until a local minimum is reached or the maximum number of steps T is reached.

local-search-R2L(LS-R2L)$(\mathcal{V}, C, T, S, \epsilon_0)$:
1 $e :=$ construct-RRL$(\mathcal{V}, C)$
2 for $t := \{1, ..., T\}$
3 $\epsilon_t = \exp(\epsilon_0 \times t)$
4 $f_e = f_\rho(e)$ using $Eq.(11)$
5 for $s := \{1, ..., S\}$
6 $e', (u, u', v, v') :=$ **rewire-edges**$(\mathcal{V}, e)$
7 $\Delta f_{u,u',v,v'} := f_e - f_\rho(e')$
8 $p_{u,u',v,v'} := \max(0, \epsilon_t + \Delta f_{u,u',v,v'})$
9 if all $p_{u,u',v,v'} = 0$
10 return e, local minimum
11 $p :=$ normalize-to-sum-1(p)
12 $e_{u,u'}, e_{v,v'} \sim$ cat$(e \times e, p)$
13 $e_{u,u'} := 0, e_{v,v'} := 0$
14 $e_{u,v} := 1, e_{u',v'} := 1$
15 $f_e := f_e + \Delta f_{u,u',v,v'}$
16 return e, not converged

rewire-edges$(\mathcal{V}, e)$:
1 $\{u, u'\} \sim$ unif$(\{u, u'\} | e_{u,u'} = 1)$
2 $\{v, v'\} \sim$ unif$(\{v, v'\} | e_{v,v'} = 1 \wedge e \setminus \{e_{u,\cdot}, e_{u',\cdot}\})$
3 $e_{u,u'} := 0, e_{v,v'} := 0$
4 $e_{u,v} := 1, e_{u',v'} := 1$
5 return $e, (u, u', v, v')$

LS-R2L method is a simple method that performs a local search in the immediate neighborhood to find a better solution among them. It is to be noted that there are methods like Monte Carlo Random Walk, which perform a more aggressive search. However, the LS-R2L method demonstrates the importance of optimal topology search for fast model averaging and serve the purpose of this paper.

4.2 Constructing Network Topology using Random Heuristics

We also like to discuss two more approaches based on random rewiring heuristics that provide better model averaging behavior than RRL, but are interestingly not employed in decentralized SGD. The first solution is proposed by Watts and Strogatz [17] to construct a small-world graph. They proposed a mechanism of converting an RRL to a small-world graph, by rewiring edges based on a given probability $p^r \in (0, 1]$. Their objective is to maintain the high clustering of an RRL, while introducing random long links to decreases the shortest path. If the rewiring probability is low, the resulting topology is closer to an RRL and for higher values it becomes a random graph.

The second method is proposed by Kar et al. [7], which is also based on random rewiring heuristics that converts RRL to a Random Regular Ramanujan Like graph. The algorithm runs for a sufficiently large numbers of steps to obtain the solution.

In this paper, we refer to Watts and Strogatz's model [17] as **WS** and Kar et al. [7] method as **R3L**. Both methods, R3L and WS, show an order of magnitude better convergence properties than RRL, however, none of these methods are employed for distributed SGD so far.

5 Experiments

In this section we address the following research questions to evaluate the proposed techniques.

(RQ1) Do learned topologies exhibit better spectral properties than the existing approaches?
(RQ2) How does the model averaging solutions provided by learned topologies impact the decentralized training of deep learning models?
(RQ3) Do Laplacian based link weights improve upon Metropolis-Hastings set weights?

5.1 Convergence Properties (RQ1)

We assess the quality of topologies obtained form LS-R2L, R3L, WS and RRL on three properties, i.e convergence factor, convergence time and the objective value. Results in Fig.(1(a)) and Fig.(1(b)) are obtained by varying the number of nodes $N = \{32, 48, \ldots, 256\}$ and the node degree $C = \{4, 6, \ldots, 14\}$. LS-R2L performs best among the four methods on three properties, followed closely by

R3L. On the other hand RRL shows an order of magnitude slower convergence properties than the other three methods. We observed that for a constant node degree C, convergence degrades as the number of nodes increases and for a fixed number of nodes N, the convergence factor improves with the increase in the node degree.

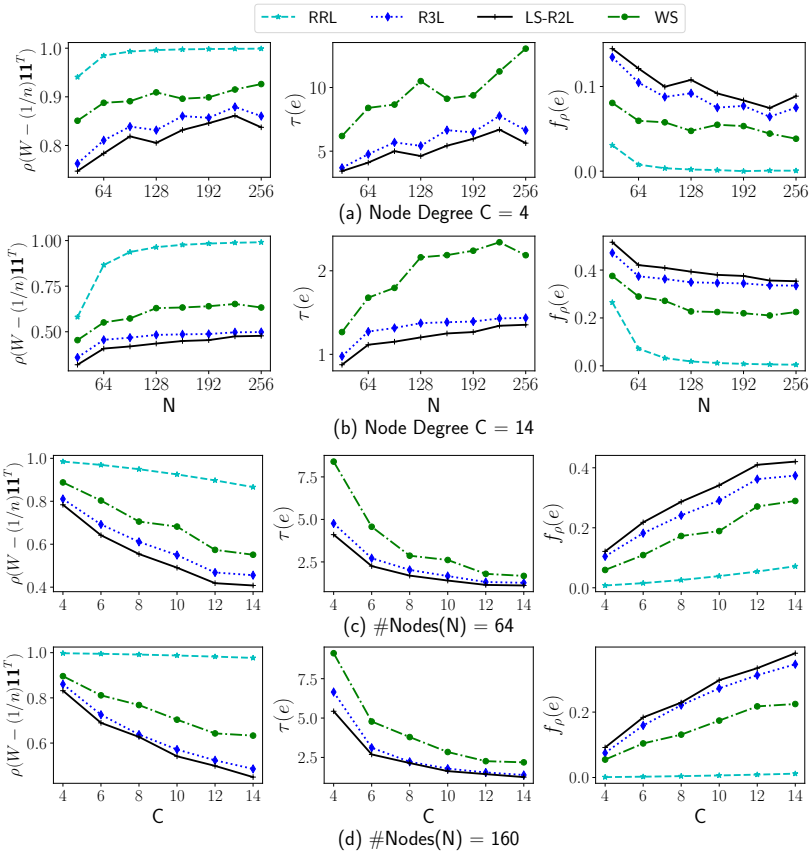

Fig. 1. LS-R2L show better convergence factor, convergence time and the objective values than other methods. We omit the RRL plots for the convergence time τ as values are an order of magnitude worse than others.

5.2 Performance Comparison (RQ2)

In this section, we empirically investigate the effect of learning an optimal topology on the decentralized parallel SGD. For this purpose, we choose image classification tasks and perform our evaluation on well-known CIFAR10 and CIFAR100, which consists of 32x32 color images with 10 and 100 classes respectively and split into 50K train-set and 10K test-set. The deep learning models and hyper-parameters for our experiments are summarized in Table 1. Implementation is

based on PyTorch and mpi4py. The experimental setup consists of "n1-standard-32" instances on the Google Cloud Platform, where each instance has 32 vCPUs, 120 GB of memory, 250GB SSD storage, and 4 Nvidia P100 GPUs. The nodes are connected through a 10Gbit/s Ethernet interconnect.

Table 1. Hyperparameters for Experiments

Dataset	Model	batch_size[3]	η	η_schedule	η_decay	size
CIFAR10	Resnet20 [5]	32	0.1	{81, 122}	0.1	1MB
	VGG16 [15]	64	0.1	{25, 50, 75, 100}	0.5	60MB
CIFAR100	DensNet-40-12 [6]	64	0.1	{150, 225}	0.1	1MB
	WideResnet-28-10 [19]	64	0.1	{60, 120, 160}	0.2	146MB

Methods: We compare solutions for decentralized SGD based on the following methods:

- **Complete** is a reference topology for the global averaging $(1/n)\mathbf{1}\mathbf{1}^T$, which has worse communication cost (grows in $O(N)$).
- **RRL**: Baseline method based on a ring topology [9].
- **RingRandom**: Baseline method proposed in [10], where workers communicate with a random neighbor within a ring.
- **WS**: Our proposed method based on Watts and Strogatz model [17].
- **R3L**: Our proposed method based on Kar et al. model [7].
- **LS-R2L**: Our proposed method based on the local search guided by Eq.(11).

Figures 2(a-b) summarize the results on the CIFAR10 datasets for Resnet20 and VGG16 respectively. Figures 2(c-d) summarize the results on the CIFAR100 datasets for DensNet-40-12 and WideResnet-28-10 respectively. The LS-R2L and R3L based solution consistently show faster epoch-wise convergence for training deep learning models. The WS based solution provides slightly degraded convergence behavior. Structured topologies, such as RRL and RingRandom, perform worse among all the methods and the differences become more pronounced when increasing the number of workers from 32 to 64. This shows that the averaging matrix generated by topologies exhibiting better spectral properties, provides faster convergence on training deep learning models. The final test accuracies achieved in the minimum number of iterations are listed in Table 2, which shows superior performance of LS-R2L and R3L compared to other methods. LS-R2L and R3L in most cases, reached best final accuracy values in the minimum number of iterations. The communication requirements for all the topologies are same, as the latency[4] grows in $O(C) \ll O(N)$.

[3] the warmup learning rate scaling technique as described in [4] is employed for stabilizing the learning process for large batch sizes i.e $B_{\text{global}} = N \times B$.

[4] Communication in parallel SGD is dominated by the number of handshakes (latency) required for transferring data, whereas the amount of data transfer (bandwidth requirements [9]) remains the same.

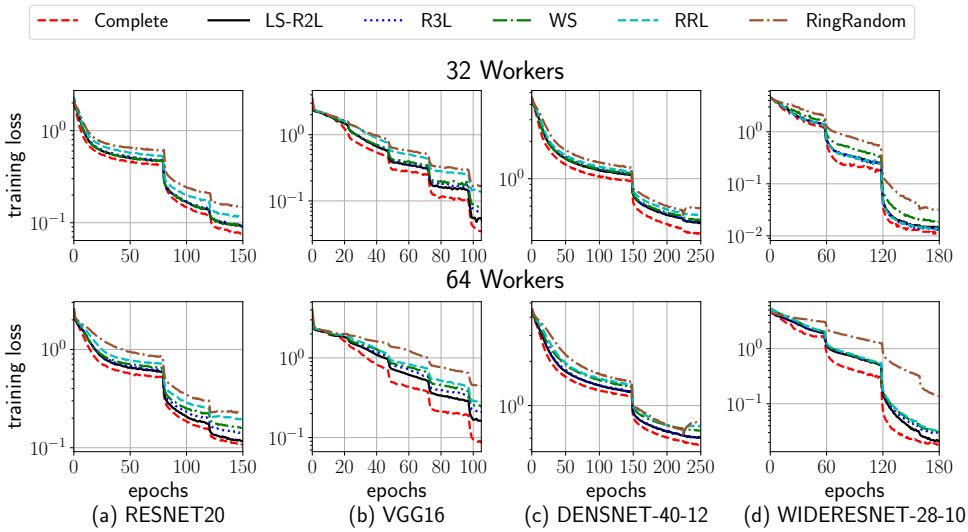

Fig. 2. Epoch-wise convergence behavior of different topologies on the CIFAR10 (a-b) and CIFAR100 (c-d) training using 32 and 64 workers.

5.3 Edge Weights (RQ3)

In this section, we empirically evaluate the impact of setting link weights based on the Laplacian approach in Eq.(7) and compare the results with Metropolis-Hastings approach in Eq.(6). Fig.(3) provides a relative comparison between Hastings and Laplacian edge weights. We trained deep learning models on 64 workers using Hastings and Laplacian link weights. The figures show relative performance in the form of relative gain, which is the ratio of training loss for Hastings over the training loss for Laplacian weights. The values less than 1 means Laplacian weights are better than Hastings weights and vise versa for values greater than 1. The Laplacian edge weights provide faster convergence than Hastings edge weights for LS-R2L and R3L graphs, whereas WS and RRL Hastings edge weights provide faster convergence. Overall LS-R2L and R3L with Laplacian edge weights show faster convergence than WS and RRL with Hastings edge weights, as shown in the previous section.

6 Conclusion

In this paper, we addressed the shortcomings in the choice of topology for model averaging in a decentralized parallel SGD. The existing literature on decentralized parallel SGD employs hand-crafted structured topologies. We show through experiments that these solutions exhibit worse convergence properties that lead to a suboptimal model averaging solution. We defined an objective function based on the spectral properties of the graph Laplacian for finding an optimal averaging scheme under constraints. As the objective function is intractable, we

Table 2. Comparison of test accuracy for the CIFAR10/100 experiments, where the numbers in () show the minimum iterations required to reach this accuracy. The best accuracy reached in the minimum number of iterations is highlighted with †.

		Reference	*Proposed*			*Baselines*	
Model	**N**	**Complete**	**LS-R2L**	**R3L**	**WS**	**RRL**	**RingRandom**
Resnet20	32	91.73 (145)	91.70 (132)	† 91.35 (125)	91.70 (132)	91.14 (141)	90.85 (150)
	64	90.90 (141)	† 90.88 (126)	90.88 (138)	90.15 (126)	89.80 (142)	89.44 (150)
VGG16	32	91.77 (98)	91.62 (99)	† 91.43 (98)	91.40 (99)	91.03 (99)	90.19 (100)
	64	91.47 (98)	† 91.35 (98)	91.01 (99)	90.84 (98)	89.77 (98)	88.74 (98)
DensNet	32	71.42 (153)	71.33 (160)	71.37 (158)	† 71.21 (154)	70.55 (226)	69.65 (228)
	64	71.25 (229)	† 71.23 (226)	70.87 (227)	70.65 (226)	70.01 (240)	68.70 (246)
Wide-	32	78.26 (133)	78.06 (138)	78.05 (130)	77.81 (171)	78.05 (150)	77.18 (169)
Resnet	64	78.15 (160)	† 78.01 (162)	77.35 (178)	77.21 (171)	77.45 (178)	76.94 (178)

provided a local search based solution (LS-R2L) for finding a good solution. The averaging matrix obtained through LS-R2L provides the best convergence behavior on image classification benchmark datasets. As a future direction, it would be interesting to apply these methods in heterogeneous systems, where each link between a pair of workers has a different communication cost.

References

1. Bijral, A.S., Sarwate, A.D., Srebro, N.: Data-dependent convergence for consensus stochastic optimization. IEEE Transactions on Automatic Control **62**(9), 4483–4498 (2017)
2. Boyd, S., Diaconis, P., Xiao, L.: Fastest mixing markov chain on a graph. SIAM Rev. **46**(4), 667–689 (2004)
3. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends Mach. Learn. **3**(1), 1–122 (Jan 2011)
4. Goyal, P., Dollár, P., Girshick, R.B., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y., He, K.: Accurate, large minibatch SGD: training imagenet in 1 hour. CoRR **abs/1706.02677** (2017)
5. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR'2016. pp. 770–778 (2016)
6. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: CVPR'2017. pp. 2261–2269 (2017)
7. Kar, S., Aldosari, S., Moura, J.M.F.: Topology for distributed inference on graphs. IEEE Transactions on Signal Processing **56**(6), 2609–2613 (2008)
8. Li, M., Andersen, D.G., Smola, A.J., Yu, K.: Communication Efficient Distributed Machine Learning with the Parameter Server. In: NIPS'2014, pp. 19–27 (2014)
9. Lian, X., Zhang, C., Zhang, H., Hsieh, C.J., Zhang, W., Liu, J.: Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent. In: NIPS'2017, pp. 5330–5340 (2017)
10. Lian, X., Zhang, W., Zhang, C., Liu, J.: Asynchronous decentralized parallel stochastic gradient descent. In: ICML'2018. pp. 3049–3058 (2018)

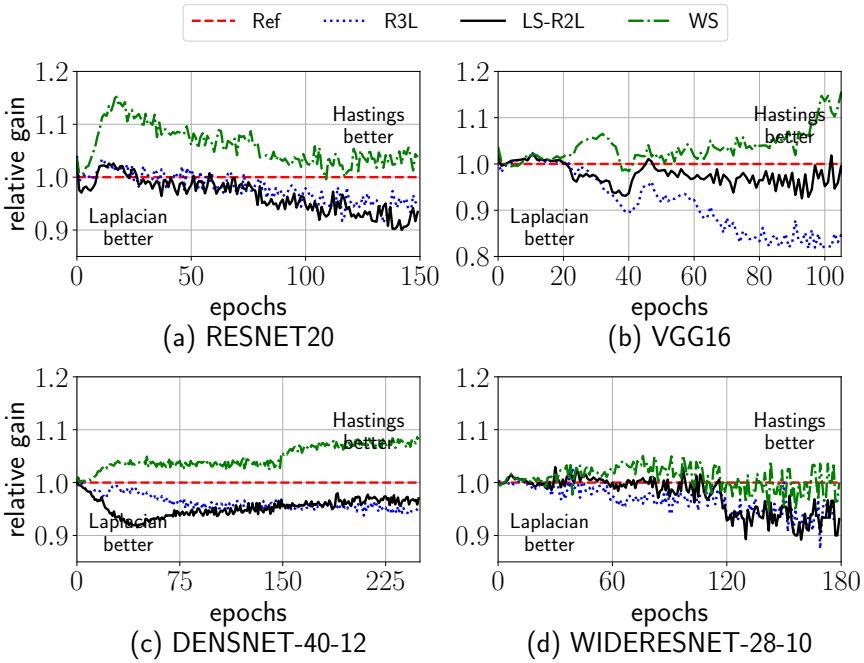

Fig. 3. Comparison of Laplacian edge weights vs hastings edge weights for LS-R2L, R3L and WS methods.

11. Liu, J., Morse, A.S.: Accelerated linear iterations for distributed averaging. Annual Reviews in Control **35**(2), 160–165 (2011)
12. Lubotzky, A., Phillips, R., Sarnak, P.: Ramanujan graphs. Combinatorica **8**(3), 261–277 (1988)
13. Lubotzky, A., Phillips, R., Sarnak, P.: European commission: General data protection regulation (GDPR) (2018), https://ec.europa.eu/commission/priorities/justice-and-fundamental-rights/dataprotection/2018-reform-eu-data-protection-rules
14. Mohar, B.: The laplacian spectrum of graphs. In: Graph Theory, Combinatorics, and Applications. pp. 871–898. Wiley (1991)
15. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR **abs/1409.1556** (2014)
16. Wang, J., Joshi, G.: Cooperative SGD: A unified framework for the design and analysis of communication-efficient SGD algorithms. CoRR **abs/1808.07576** (2018)
17. Watts, D.J., Strogatz, S.H.: Collective dynamics of small-world networks. Nature **393**(6684), 440 (1998)
18. Xiao, L., Boyd, S.: Fast linear iterations for distributed averaging. Systems and Control Letters **53**, 65–78 (2004)
19. Zagoruyko, S., Komodakis, N.: Wide residual networks. In: Proceedings of the British Machine Vision Conference (BMVC). pp. 87.1–87.12 (2016)
20. Zhang, J., Sa, C.D., Mitliagkas, I., Ré , C.: Parallel sgd: When does averaging help. In: Optimization in Machine Learning Workshop ICML (2016)
21. Zinkevich, M., Weimer, M., Li, L., Smola, A.J.: Parallelized stochastic gradient descent. In: NIPS'2010, pp. 2595–2603 (2010)