Journal of Applied and Computational Topology
https://doi.org/10.1007/s41468-023-00127-8

GIST: distributed training for large-scale graph convolutional networks

Cameron R. Wolfe[1] · Jingkang Yang[3] · Fangshuo Liao[1] · Arindam Chowdhury[2] · Chen Dun[1] · Artun Bayer[2] · Santiago Segarra[2] · Anastasios Kyrillidis[1]

Received: 21 June 2022 / Revised: 19 January 2023 / Accepted: 7 June 2023

Abstract
The graph convolutional network (GCN) is a go-to solution for machine learning on graphs, but its training is notoriously difficult to scale both in terms of graph size and the number of model parameters. Although some work has explored training on large-scale graphs, we pioneer efficient training of large-scale GCN models with the proposal of a novel, distributed training framework, called GIST. GIST disjointly partitions the parameters of a GCN model into several, smaller sub-GCNs that are trained independently and in parallel. Compatible with all GCN architectures and existing sampling techniques, GIST (i) improves model performance, (ii) scales to training on arbitrarily large graphs, (iii) decreases wall-clock training time, and (iv) enables the training of markedly overparameterized GCN models. Remarkably, with GIST, we train an astonishgly-wide 32–768-dimensional GraphSAGE model, which exceeds the capacity of a single GPU by a factor of 8×, to SOTA performance on the Amazon2M dataset.

Keywords Graph neural networks · Distributed training · Efficient training · Overparameterization

Mathematics Subject Classification 68T07

Cameron R. Wolfe, Jingkang Yang and Fangshuo Liao have contributed equally to this work.

✉ Cameron R. Wolfe
crw13@rice.edu

[1] Department of Computer Science, Rice University, 6100 Main Street, Houston, TX 77005, USA

[2] Department of Electrical and Computer Engineering, Rice University, 6100 Main Street, Houston, TX 77005, USA

[3] School of Computer Science and Engineering, Nanyang Technology University, 50 Nanyang Avenue, Singapore 639798, Singapore

Published online: 14 August 2023

1 Introduction

Since not all data can be represented in Euclidean space (Bronstein et al. 2017), many applications rely on graph-structured data. For example, social networks can be modeled as graphs by regarding each user as a node and friendship relations as edges (Lusher et al. 2013; Newman et al. 2002). Alternatively, in chemistry, molecules can be modeled as graphs, with nodes representing atoms and edges encoding chemical bonds (Balaban et al. 1985; Benkö et al. 2003).

To better understand graph-structured data, several (deep) learning techniques have been extended to the graph domain (Defferrard et al. 2016; Gori et al. 2005; Masci et al. 2015). Currently, the most popular one is the graph convolutional network (GCN) (Kipf and Welling 2016), a multi-layer architecture that implements a generalization of the convolution operation to graphs. Although the GCN handles node- and graph-level classification, it is notoriously inefficient and unable to support large graphs (Chen et al. 2018a, b; Gao et al. 2018; Huang et al. 2018; You et al. 2020; Zeng et al. 2019), making practical, large-scale applications difficult to handle.

To deal with these issues, node partitioning methodologies have been developed. These schemes can be roughly categorized into neighborhood sampling (Chen et al. 2018b; Hamilton et al. 2017; Zou et al. 2019) and graph partitioning (Chiang et al. 2019; Zeng et al. 2019) approaches. The goal is to partition a large graph into multiple smaller graphs that can be used as mini-batches for training the GCN. In this way, GCNs can handle larger graphs during training, expanding their potential into the realm of big data. However, the size of the underlying model is still limited by available memory capacity, thus placing further constraints on the scale of GCN experimentation.

Although some papers perform large-scale experiments (Chiang et al. 2019; Zeng et al. 2019), the models (and data) used in GCN research remain small in the context of deep learning (Kipf and Welling 2016; Veličković et al. 2017), where the current trend is towards incredibly large models and datasets (Brown et al. 2020; Conneau et al. 2019). Despite the widespread moral questioning of this trend (Hao 2019; Peng and Sarazen 2019; Sharir et al. 2020), the deep learning community continues to push the limits of scale. Overparameterized models yield improvements in tasks like zero/few-shot learning (Brown et al. 2020; Radford et al. 2021), are capable of discovering generalizable solutions (Nakkiran et al. 2019), and even have desirable theoretical properties (Oymak and Soltanolkotabi 2020).

Although deeper GCNs may perform poorly due to oversmoothing (Kipf and Welling 2016; Li et al. 2018), GCNs should similarly benefit from overparameterization, meaning that larger hidden layers may be beneficial. Furthermore, recent work indicates that overparameterization is most impactful on larger datasets (Hoffmann et al. 2022), making overparameterized models essential as GCNs are applied to practical problems at scale. Moving in this direction, *our work provides an efficient training framework for wide, overparameterized GCN models—beyond the memory capacity of a single GPU—of any architecture that is compatible with existing training techniques.*

This paper. Inspired by independent subnetwork training (IST) (Yuan et al. 2019), our methodology randomly partitions the hidden feature space in each layer, decomposing the global GCN model into multiple, narrow sub-GCNs of equal depth. Sub-GCNs

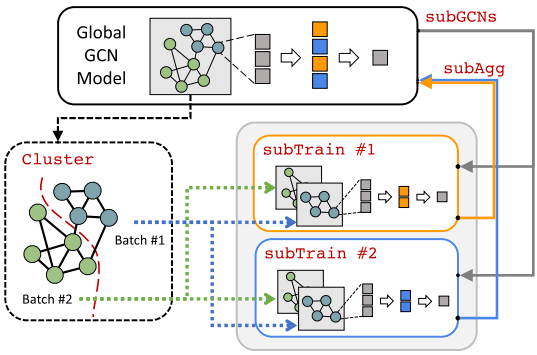

Fig. 1 GIST pipeline: subGCNs divides the global GCN into sub-GCNs. Every sub-GCN is trained by subTrain using mini-batches (smaller sub-graphs) generated by Cluster. Sub-GCN parameters are intermittently aggregated through subAgg

are trained independently for several iterations in parallel prior to having their updates synchronized; see Fig. 1. This process of randomly partitioning, independently training, and synchronizing sub-GCNs is repeated until convergence. We call this method **G**raph **I**ndependent **S**ubnetwork **T**raining (GIST), as it extends the IST framework to the training of GCNs.

Though IST was previously unexplored in this domain, we find that GIST pairs well with any GCN architecture, is compatible with node sampling techniques, can scale to arbitrarily large graphs, and significantly reduces wall-clock training time, allowing larger models and datasets to be explored. In particular, we focus on training "ultra-wide" GCNs (i.e., GCN models with very large hidden layers), as deeper GCNs are prone to oversmoothing (Li et al. 2018) and GISTs model partitioning strategy can mitigate the memory overhead of training these wider GCNs.

The contributions of this work are as follows:

- We develop a novel extension of IST for training GCNs, show that it works well for training GCNs with a variety of architectures, and demonstrate its compatibility with commonly-used GCN training techniques like neighborhood sampling and graph partitioning.
- We show that GIST can be used to reach state-of-the-art performance with reduced training time relative to standard training methodologies. GIST is a compatible addition to GCN training that improves efficiency.
- We propose a novel Graph Independent Subnetwork Training Kernel (GIST-K) that allows a convergence rate to be derived for two-layer GCNs trained with GIST in the infinite width regime. Based on GIST-K, we provide theory that GIST converges linearly –up to an error neighborhood– using distributed gradient descent with local iterations. We show that the radius of the error neighborhood is controlled by the overparameterization parameter, as well as the number of workers in the distributed setting. Such findings reflect practical observations that are made in the experimental section.
- We use GIST to enable the training of markedly overparameterized GCN models. In particular, GIST is used to train a two-layer GraphSAGE model with a hidden dimension of 32 768 on the Amazon2M dataset. *Such a model exceeds the capacity of a single GPU by* 8×.

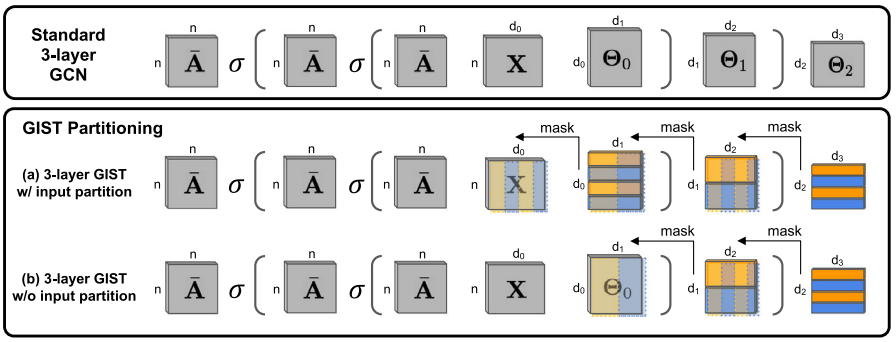

Fig. 2 GCN partition into $m = 2$ sub-GCNs. Orange and blue colors depict different feature partitions. Both hidden dimensions (d_1 and d_2) are partitioned. The output dimension (d_3) is not partitioned. Partitioning the input dimension (d_0) is optional, but we do not partition d_0 in `GIST`

2 What is the `GIST` of this work?

Algorithm 1 `GIST` Algorithm

Parameters: T synchronization iterations, m sub-GCNs,
ζ local iterations, c clusters, $\mathcal{G}$ training graph.

$\Psi_{\mathcal{G}}(\cdot\,; \boldsymbol{\Theta}) \leftarrow$ randomly initialize GCN
$\{\mathcal{G}_{(j)}\}_{j=1}^{c} \leftarrow \texttt{Cluster}(\mathcal{G}, c)$
for $t = 0, \ldots, T-1$ **do**
  $\left\{\Psi_{\mathcal{G}}(\cdot\,; \boldsymbol{\Theta}^{(i)})\right\}_{i=1}^{m} \leftarrow \texttt{subGCNs}(\Psi_{\mathcal{G}}(\cdot\,; \boldsymbol{\Theta}), m)$
  Distribute each $\Psi_{\mathcal{G}}(\cdot\,; \boldsymbol{\Theta}^{(i)})$ to a different worker
  for $i = 1, \ldots, m$ **do**
    for $z = 1, \ldots, \zeta$ **do**
      $\Psi_{\mathcal{G}}(\cdot\,; \boldsymbol{\Theta}^{(i)}) \leftarrow \texttt{subTrain}(\boldsymbol{\Theta}^{(i)}, \{\mathcal{G}_{(j)}\}_{j=1}^{c})$
    end for
  end for
  $\Psi_{\mathcal{G}}(\cdot\,; \boldsymbol{\Theta}) \leftarrow \texttt{subAgg}(\{\boldsymbol{\Theta}^{(i)}\}_{i=1}^{m})$
end for

GCN Architecture. The GCN (Kipf and Welling 2016) is arguably the most widely-used neural network architecture on graphs. Consider a graph $\mathcal{G}$ comprised of n nodes with d-dimensional features $\mathbf{X} \in \mathbb{R}^{n \times d}$. The output $\mathbf{Y} \in \mathbb{R}^{n \times d'}$ of a GCN can be expressed as $\mathbf{Y} = \Psi_{\mathcal{G}}(\mathbf{X}; \boldsymbol{\Theta})$, where $\Psi_{\mathcal{G}}$ is an L-layered architecture with trainable parameters $\boldsymbol{\Theta}$. If we define $\mathbf{H}_0 = \mathbf{X}$, we then have that $\mathbf{Y} = \Psi_{\mathcal{G}}(\mathbf{X}; \boldsymbol{\Theta}) = \mathbf{H}_L$, where an intermediate ℓ-th layer of the GCN is given by

$$\mathbf{H}_{\ell+1} = \sigma(\bar{\mathbf{A}}\, \mathbf{H}_{\ell}\, \boldsymbol{\Theta}_{\ell}). \tag{1}$$

In (1), $\sigma(\cdot)$ is an elementwise activation function (e.g., ReLU), $\bar{\mathbf{A}}$ is the degree-normalized adjacency matrix of $\mathcal{G}$ with added self-loops, and the trainable parameters $\boldsymbol{\Theta} = \{\boldsymbol{\Theta}_{\ell}\}_{\ell=0}^{L-1}$ have dimensions $\boldsymbol{\Theta}_{\ell} \in \mathbb{R}^{d_{\ell} \times d_{\ell+1}}$ with $d_0 = d$ and $d_L = d'$. In

Fig. 2 (top), we illustrate nested GCN layers for $L = 3$, but our methodology extends to arbitrary L. The activation function of the last layer is typically the identity or softmax transformation – we omit this in Fig. 2 for simplicity.

GIST overview. We overview GIST in Algorithm 1 and present a schematic depiction in Fig. 1. We partition our (randomly initialized) global GCN into m smaller, disjoint sub-GCNs with the subGCNs function ($m = 2$ in Figs. 1 and 2) by sampling the feature space at each layer of the GCN; see Sect. 2.1. Each sub-GCN is assigned to a different worker (i.e., a different GPU) for ζ rounds of distributed, independent training through subTrain. Then, newly-learned sub-GCN parameters are aggregated (subAgg) into the global GCN model. This process repeats for T iterations. Our graph domain is partitioned into c sub-graphs through the Cluster function ($c = 2$ in Fig. 1). This operation is only relevant for large graphs ($n > 50\,000$), and we omit it ($c = 1$) for smaller graphs that don't require partitioning.[1]

2.1 subGCNs: constructing sub-GCNs

GIST partitions a global GCN model into several narrower sub-GCNs of equal depth. Formally, consider an arbitrary layer ℓ and a random, disjoint partition of the feature set $[d_\ell] = \{1, 2, \ldots, d_\ell\}$ into m equally-sized blocks $\{\mathcal{D}_\ell^{(i)}\}_{i=1}^m$.[2] Accordingly, we denote by $\mathbf{\Theta}_\ell^{(i)} = [\mathbf{\Theta}_\ell]_{\mathcal{D}_\ell^{(i)} \times \mathcal{D}_{\ell+1}^{(i)}}$ the matrix obtained by selecting from $\mathbf{\Theta}_\ell$ the rows and columns given by the ith blocks in the partitions of $[d_\ell]$ and $[d_{\ell+1}]$, respectively. With this notation in place, we can define m different sub-GCNs $\mathbf{Y}^{(i)} = \Psi_{\mathcal{G}}(\mathbf{X}^{(i)}; \mathbf{\Theta}^{(i)}) = \mathbf{H}_L^{(i)}$ where $\mathbf{H}_0^{(i)} = \mathbf{X}_{[n] \times \mathcal{D}_0^{(i)}}$ and each layer is given by:

$$\mathbf{H}_{\ell+1}^{(i)} = \sigma(\bar{\mathbf{A}}\, \mathbf{H}_\ell^{(i)}\, \mathbf{\Theta}_\ell^{(i)}). \tag{2}$$

Notably, not all parameters within the global GCN model are partitioned to a sub-GCN. However, by randomly re-constructing new groups of sub-GCNs according to a uniform distribution throughout the training process, all parameters have a high likelihood of being updated. In Sect. 4, we provide theoretical guarantees that the partitioning of model parameters to sub-GCNs does not harm training performance.

Sub-GCN partitioning is illustrated in Fig. 2a, where $m = 2$. Partitioning the input features is optional (i.e., (a) vs. (b) in Fig. 2). *We do not partition the input features within GIST* so that sub-GCNs have identical input information (i.e., $\mathbf{X}^{(i)} = \mathbf{X}$ for all i); see Sect. 5.1. Similarly, we do not partition the output feature space to ensure that the sub-GCN output dimension coincides with that of the global model, thus avoiding any need to modify the loss function. This decomposition procedure (subGCNs function in Algorithm 1) extends to arbitrary L.

[1] Though any clustering method can be used, we advocate the use of METIS (Karypis and Kumar 1998a, b) due to its proven efficiency in large-scale graphs.

[2] For example, if $d_\ell = 4$ and $m = 2$, one valid partition would be given by $\mathcal{D}_\ell^{(1)} = \{1, 4\}$ and $\mathcal{D}_\ell^{(2)} = \{2, 3\}$.

2.2 `subTrain`: independently training sub-GCNs

Assume $c = 1$ so that the `Cluster` operation in Algorithm 1 is moot and $\{\mathcal{G}_{(j)}\}_{j=1}^{c} = \mathcal{G}$. Because $\mathbf{Y}^{(i)}$ and $\mathbf{Y}$ share the same dimension, sub-GCNs can be trained to minimize the same global loss function. One application of `subTrain` in Algorithm 1 corresponds to a single step of stochastic gradient descent (SGD). Inspired by local SGD (Lin et al. 2018), multiple, independent applications of `subTrain` are performed in parallel (i.e., on separate GPUs) for each sub-GCN prior to aggregating weight updates. The number of independent training iterations between synchronization rounds, referred to as *local iterations*, is denoted by ζ, and the total amount of training is split across sub-GCNs.[3] Ideally, the number sub-GCNs and local iterations should be increased as much as possible to minimize communication and training costs. In practice, however, such benefits may come at the cost of statistical inefficiency; see Sect. 5.1.

If $c > 1$, `subTrain` first selects one of the c subgraphs in $\{\mathcal{G}_{(j)}\}_{j=1}^{c}$ to use as a mini-batch for SGD. Alternatively, the union of several sub-graphs in $\{\mathcal{G}_{(j)}\}_{j=1}^{c}$ can be used as a mini-batch for training. Aside from using mini-batches for each SGD update instead of the full graph, *the use of graph partitioning does not modify the training approach outlined above.* Some form of node sampling must be adopted to make training tractable when the full graph is too large to fit into memory. However, *both graph partitioning and layer sampling are compatible with* `GIST` (see Sects. 5.2 and 5.4). We adopt graph partitioning in the main experiments due to the ease of implementation. The novelty of our work lies in the feature partitioning strategy of `GIST` for distributed training, which is an orthogonal technique to node sampling; see Fig. 3 and Sect. 2.4.

2.3 `subAgg`: aggregating sub-GCN parameters

After each sub-GCN completes ζ training iterations, their updates are aggregated into the global model (`subAgg` function in Algorithm 1). Within `subAgg`, each worker replaces global parameter entries within $\boldsymbol{\Theta}$ with its own, independently-trained sub-GCN parameters $\boldsymbol{\Theta}^{(i)}$, where no collisions occur due to the disjointness of sub-GCN partitions. Thus, `subAgg` is a basic copy operation that transfers sub-GCN parameters into the global model.

Not every parameter in the global GCN model is updated by `subAgg` because, as previously mentioned, parameters exist that are not partitioned to any sub-GCN by the `subGCNs` operation. For example, focusing on $\boldsymbol{\Theta}_1$ in Fig. 2a, one worker will be assigned $\boldsymbol{\Theta}_1^{(1)}$ (i.e., overlapping orange blocks), while the other worker will be assigned $\boldsymbol{\Theta}_1^{(2)}$ (i.e., overlapping blue blocks). The rest of $\boldsymbol{\Theta}_1$ is not considered within `subAgg`. Nonetheless, since sub-GCN partitions are randomly drawn in each cycle t, one expects all of $\boldsymbol{\Theta}$ to be updated multiple times if T is sufficiently large.

[3] For example, if a global model is trained on a single GPU for 10 epochs, a comparable experiment for `GIST` with two sub-GCNs would train each sub-GCN for only 5 epochs.

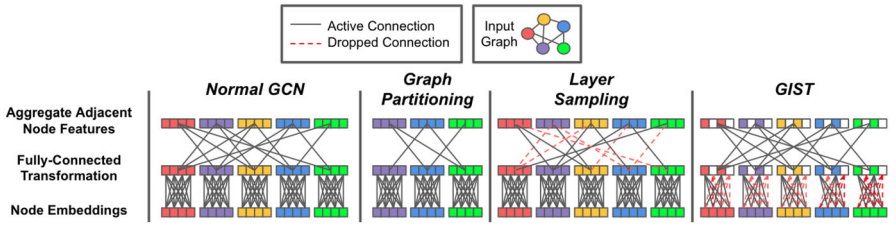

Fig. 3 Illustrates the difference between GIST and node sampling techniques within the forward pass of a single GCN layer (excluding non-linear activation). While graph partitioning and layer sampling remove nodes from the forward pass (i.e., either completely or on a per-layer basis), GIST partitions node feature representations (and, in turn, model parameters) instead of the nodes themselves

2.4 What is the value of GIST?

Architecture-Agnostic Distributed Training. GIST is a generic, distributed training methodology that can be used for any GCN architecture. We implement GIST for vanilla GCN, GraphSAGE, and GAT architectures, but GIST is not limited to these models; see Sect. 5.

Compatibility with Sampling Methods. GIST is **NOT** a replacement for graph or layer sampling. Rather, it is an efficient, distributed training technique that can be used in tandem with node partitioning. As depicted in Fig. 3, GIST partitions node feature representations and model parameters between sub-GCNs, while graph partitioning and layer sampling sub-sample nodes within the graph.

Interestingly, we find that GIST's feature and parameter partitioning strategy is compatible with node partitioning—the two approaches can be combined to yield further efficiency benefits. For example, GIST is combined with graph partitioning strategies in Sect. 5.2 and with layer sampling methodologies in Sect. 5.4. As such, we argue that GIST offers an easy add-on to GCN training that makes larger scale experiments more feasible.

Enabling Ultra-Wide GCN Training. GIST indirectly updates the global GCN through the training of smaller sub-GCNs, enabling models with hidden dimensions that exceed the capacity of a single GPU to be trained; in our experiments, we show results where GIST allows training of models beyond the capacity of a single GPU by a factor of 8×. In this way, GIST allows markedly overparametrized ("ultra-wide") GCN models to be trained on existing hardware. In Sect. 5.2, *we leverage this capability to train a two-layer GCN model with a hidden dimension of* 32 768 *on Amazon2M.*

Overparameterization through width is especially relevant to GCNs because deeper models suffer from oversmoothing (Li et al. 2018). Additionally, the theoretical results provided within Sect. 4 reveal that the performance of GIST is best as the number of neurons within each hidden layer is increased, which further reveals the benefit of wide, overparameterized layers. We do not explore depth-wise partitions of different GCN layers to each worker, but rather focus solely upon partitioning the hidden neurons within each layer.

Improved Model Complexity. Consider a single GCN layer, trained over M machines with input and output dimension of d_{i-1} and d_i, respectively. For one synchro-

nization round, the communication complexity of GIST and standard distributed training is $\mathcal{O}(\frac{1}{M} d_i d_{i-1})$ and $\mathcal{O}(M d_i d_{i-1})$, respectively. GIST reduces communication by only communicating sub-GCN parameters. Existing node partitioning techniques cannot similarly reduce communication complexity because model parameters are never partitioned. Furthermore, the computational complexity of the forward pass for a GCN model trained with GIST and using standard methodology is $\mathcal{O}(\frac{1}{M} N^2 d_i + \frac{1}{M^2} N d_i d_{i-1})$ and $\mathcal{O}(N^2 d_i + N d_i d_{i-1})$, respectively, where N is the number of nodes in the partition being processed.[4] Node partitioning can reduce N by a constant factor but is compatible with GIST.

Relation to IST. Our work extends the IST distributed training framework—originally proposed for fully-connected network architectures (Yuan et al. 2019)—to GCNs. Due to the unique aspects of GCN training (e.g., non-euclidean data and aggregation of node features), it was previously unclear whether IST would work well in this domain. Though IST is applicable to a variety of architectures, we find that it is especially useful for efficiently training GCNs to high accuracy. GIST i) provides speedups and performance benefits, ii) is compatible with other efficient GCN training methods, and iii) enables training of uncharacteristically-wide GCN models, allowing overparameterized GCNs to be explored via greater width. The practical utility of GIST and interplay of the approach with unique aspects of GCN training differentiate our work from the original IST proposal.

3 Related work

GCN training. In spite of their widespread success in several graph related tasks, GCNs often suffer from training inefficiencies (Gao et al. 2018; Huang et al. 2018). Consequently, the research community has focused on developing efficient and scalable algorithms for training GCNs (Chen et al. 2018a, b; Chiang et al. 2019; Hamilton et al. 2017; Zeng et al. 2019; Zou et al. 2019). The resulting approaches can be divided roughly into two areas: *neighborhood sampling* and *graph partitioning*. However, it is important to note that these two broad classes of solutions are not mutually exclusive, and reasonable combinations of the two approaches may be beneficial.

Neighborhood sampling methodologies aim to sub-select neighboring nodes at each layer of the GCN, thus limiting the number of node representations in the forward pass and mitigating the exponential expansion of the GCNs receptive field. VRGCN (Chen et al. 2018a) implements a variance reduction technique to reduce the sample size in each layer, which achieves good performance with smaller graphs. However, it requires to store all the intermediate node embeddings during training, leading to a memory complexity close to full-batch training. GraphSAGE (Hamilton et al. 2017) learns a set of aggregator functions to gather information from a node's local neighborhood. It then concatenates the outputs of these aggregation functions with each node's own representation at each step of the forward pass. FastGCN (Chen et al. 2018b) adopts a Monte Carlo approach to evaluate the GCN's forward pass in practice, which computes each node's hidden representation using a fixed-size, randomly-sampled set of nodes.

[4] We omit the complexity of applying the element-wise activation function for simplicity.

LADIES (Zou et al. 2019) introduces a layer-conditional approach for node sampling, which encourages node connectivity between layers in contrast to FastGCN (Chen et al. 2018b).

Graph partitioning schemes aim to select densely-connected sub-graphs within the training graph, which can be used to form mini-batches during GCN training. Such sub-graph sampling reduces the memory footprint of GCN training, thus allowing larger models to be trained over graphs with many nodes. ClusterGCN (Chiang et al. 2019) produces a very large number of clusters from the global graph, then randomly samples a subset of these clusters and computes their union to form each sub-graph or mini-batch. Similarly, GraphSAINT (Zeng et al. 2019) randomly samples a sub-graph during each GCN forward pass. However, GraphSAINT also considers the bias created by unequal node sampling probabilities during sub-graph construction, and proposes normalization techniques to eliminate this bias.

As explained in Sect. 2, `GIST` also relies on graph partitioning techniques (`Cluster`) to handle large graphs. However, the feature sampling scheme at each layer (`subGCNs`) that leads to parallel and narrower sub-GCNs is a hitherto unexplored framework for efficient GCN training.

Distributed training. Distributed training is a heavily studied topic (Shi et al. 2020; Zhang et al. 2018). Our work focuses on synchronous and distributed training techniques (Lian et al. 2017; Yu et al. 2019; Zhang et al. 2015). Some examples of synchronous, distributed training approaches include data parallel training, parallel SGD (Agarwal and Duchi 2011; Zinkevich et al. 2020), and local SGD (Lin et al. 2018; Stich 2019). Our methodology holds similarities to model parallel training techniques, which have been heavily explored (Ben-Nun and Hoefler 2019; Gholami et al. 2017; Günther et al. 2018; Kirby et al. 2020; Pauloski et al. 2020; Tavarageri et al. 2019; Zhu et al. 2020). More closely, our approach is inspired by IST, explored for feed-forward networks in Yuan et al. (2019). Later work analyzed IST theoretically (Liao and Kyrillidis 2021) and extended its use to more complex ResNet architectures (Dun et al. 2022). We explore the extension of IST to the GCN architecture both theoretically and empirically, finding that IST-based methods are suited well for GCN training. However, the IST framework is applicable to network architectures beyond the GCN.

4 Theoretical results

We draw upon analysis related to neural tangent kernels (NTK) Jacot et al. (2018) to derive a convergence rate for two-layer GCNs using gradient descent—as formulated in (1) and further outlined in "Appendix C.1"—trained with `GIST`. Given the scaled Gram matrix of an infinite-dimensional NTK $\mathbf{H}^{\infty}$, we define the Graph Independent Subnetwork Training Kernel (`GIST-K`) as follows:

$$\mathbf{G}^{\infty} = \bar{\mathbf{A}}\mathbf{H}^{\infty}\bar{\mathbf{A}}.$$

Given the GIST-K, we adopt the following set of assumptions related to the underlying graph; see "Appendix C.3" for more details.

Notations. Let n denote the number of nodes (training samples) in graph of interest, $d = d_0$ be dimension of the feature vector of each node, and m be the number of sub-GCNs in procedure 4. Let $\lambda_0 = \lambda_{\min}(\mathbf{G}^{\infty})$ and $\lambda^* = \lambda_{\max}(\mathbf{G}^{\infty})$ be the minimum and maximum eigenvalue of $\mathbf{G}^{\infty}$, respectively. Lastly, we denote $\mathbb{E}_{[\mathcal{M}_t]}[\cdot] = \mathbb{E}_{\mathcal{M}_0,\ldots,\mathcal{M}_t}[\cdot]$ to denote the total expectation with respect to $\mathcal{M}_0, \ldots, \mathcal{M}_t$.

Assumption 1 Assume $\lambda_{\min}(\bar{\mathbf{A}}) \neq 0$ and there exists $\epsilon \in (0, 1)$ and $p \in \mathbb{Z}_+$ such that $(1-\epsilon)^2 p \leq \mathbf{D}_{ii} \leq (1+\epsilon)^2 p$ for all $i \in [n] = \{1, 2, \ldots, n\}$, where $\mathbf{D}$ is the degree matrix. Additionally, assume that i) input node representations are bounded in norm and not parallel to any other node representation, ii) output node representations are upper bounded, iii) sub-GCN feature partitions are generated at each iteration from a categorical distribution with uniform mean $\frac{1}{m}$.

Given this set of assumptions, we can guarantee that $\lambda_0 > 0$ (a detailed discussion is deferred to Sect. C.5). Under such conditions, we derive the following result for GCN models trained with GIST.

Theorem 1 *Suppose assumptions 2–4 hold. Moreover, suppose in each global iteration the masks are generated from a categorical distribution with uniform mean* $\frac{1}{m}$. *Fix the number of global iterations to* T *and local iterations to* ζ. *Consider a two-layer GCN with parameters* $\boldsymbol{\Theta}$. *If each entry of* $\boldsymbol{\Theta}$ *is initialized I.I.D. from* $\mathcal{N}(0, \kappa^2 \mathbf{I})$, *and the number of hidden neurons satisfies* $d_1 \geq \Omega\left(\frac{T^2\zeta^2 n}{\lambda_0^4(1-\gamma)^2} \max\left\{\frac{n^3}{\delta^2\kappa^2}, \frac{n^2 d}{\delta^2}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}, T^2\lambda^{*2} d\right\}\right)$, *then procedure* (4) *with constant step size* $\eta = \mathcal{O}\left(\frac{\lambda_0}{n\left\|\bar{\mathbf{A}}^2\right\|_{1,1}}\right)$ *converges according to*

$$\mathbb{E}_{[\mathcal{M}_{t-1}]}\left[\left\|\mathbf{y} - \hat{\mathbf{y}}(t)\right\|_2^2\right] \leq \left(\gamma + (1-\gamma)\left(1 - \frac{\eta\lambda_0}{2}\right)^{\zeta}\right)^t \left\|\mathbf{y} - \hat{\mathbf{y}}(0)\right\|_2^2 + \mathcal{O}\left(\frac{\gamma^2 d\kappa^2 \lambda^{*2}}{m^2(1-\gamma)\lambda_0^2}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}\right)$$

with probability at least $1 - \delta$, *where* $\gamma = \left(1 - m^{-1}\right)^{\frac{1}{3}}$.

A full proof of this result is deferred to "Appendix C", but a sketch of the techniques used is as follows:

1. We define the GIST-K and show that it remains positive definite throughout training given our assumptions and sufficient overparameterization.
2. We show that local sub-GCN training converges linearly, given a positive definite GIST-K.
3. We analyze the change in training error when sub-GCNs are sampled (subGCNs), locally trained (subTrain), and aggregated (subAgg).

Table 1 Test accuracy of GCN models trained on small-scale datasets with GIST

m	d_0	d_1	d_2	Cora	Citeseer	Pubmed	OGBN-Arxiv
Baseline				81.52 ± 0.005	75.02 ± 0.018	75.90 ± 0.003	70.85 ± 0.089
2	✓	✓	✓	80.00 ± 0.010	$\mathbf{75.95} \pm 0.007$	76.68 ± 0.011	65.65 ± 0.700
	✓	✓		78.30 ± 0.011	69.34 ± 0.018	75.78 ± 0.015	65.33 ± 0.347
		✓	✓	$\mathbf{80.82} \pm 0.010$	75.82 ± 0.008	$\mathbf{78.02} \pm 0.007$	$\mathbf{70.10} \pm 0.224$
4	✓	✓	✓	76.78 ± 0.017	70.66 ± 0.011	65.67 ± 0.044	54.21 ± 1.360
	✓	✓		66.56 ± 0.061	68.38 ± 0.018	68.44 ± 0.014	52.64 ± 1.988
		✓	✓	$\mathbf{81.18} \pm 0.007$	$\mathbf{76.21} \pm 0.017$	$\mathbf{76.99} \pm 0.006$	$\mathbf{68.69} \pm 0.579$
8	✓	✓	✓	48.32 ± 0.087	45.42 ± 0.092	54.29 ± 0.029	40.26 ± 1.960
	✓	✓		53.60 ± 0.020	54.68 ± 0.030	51.44 ± 0.002	26.84 ± 7.226
		✓	✓	$\mathbf{79.58} \pm 0.006$	$\mathbf{75.39} \pm 0.016$	$\mathbf{76.99} \pm 0.006$	$\mathbf{65.81} \pm 0.378$

We selectively partition each feature dimension within the GCN model, indicated by a check mark. *Partitioning on all hidden layers except the input layer leads to optimal performance*
Bold values indicate the best performance on each dataset given a fixed value of m. A single value is bolded for each combination of dataset and value of m

4. We establish a connection between local and aggregated weight perturbation, showing that network parameters are bounded by a small region centered around the initialization given sufficient overparameterization.

Discussion. Stated intuitively, the result in Theorem 1 shows that, given sufficient width, two-layer GCNs trained using GIST converge to approximately zero training error. The convergence rate is linear and on par with training the full, two-layer GCN model (i.e., without the feature partition utilized in GIST), up to an error neighborhood. Notice choosing a smaller initialization scale κ will result in a smaller size of the error neighborhood but at the same time a larger overparameterization requirement. Such theory shows that the feature partitioning strategy of GIST does not cause the model to diverge in training. Additionally, the theory suggests that wider GCN models should be used to maximize the convergence rate of GIST and minimize the impact of the additive term within Theorem 1. Such findings reflect practical observations that are made within Sect. 5 and reveal that GIST is particularly-suited towards training extremely wide models that cannot be trained using a traditional, centralized approach on a single GPU due to limited memory capacity.

5 Experiments

We use GIST to train different GCN architectures on six public, multi-node classification datasets; see "Appendix A" for details. In most cases, we compare the performance of models trained with GIST to that of models trained with standard methods (i.e., single GPU with node partitioning). Comparisons to models trained with other distributed methodologies are also provided in "Appendix B". Experiments are divided into small and large scale regimes based upon graph size. The goal of GIST

is to (i) train GCN models to state-of-the-art performance, (ii) minimize wall-clock training time, and (iii) enable training of very wide GCN models.

5.1 Small-scale experiments

In this section, we perform experiments over Cora, Citeseer, Pubmed, and OGBN-Arxiv datasets (Sen et al. 2008; Hu et al. 2020). For these small-scale datasets, we train a three-layer, 256-dimensional GCN model (Kipf and Welling 2016) with `GIST`; see "Appendix A.3" for further experimental settings. All reported metrics are averaged across five separate trials. Because these experiments run quickly, we use them to analyze the impact of different design and hyperparameter choices rather than attempting to improve runtime (i.e., speeding up such short experiments is futile).

Which layers should be partitioned? We investigate whether models trained with `GIST` are sensitive to the partitioning of features within certain layers. Although the output dimension d_3 is never partitioned, we selectively partition dimensions d_0, d_1, and d_2 to observe the impact on model performance; see Table 1. Partitioning input features (d_0) significantly degrades test accuracy because sub-GCNs observe only a portion of each node's input features (i.e., this becomes more noticeable with larger m). However, other feature dimensions cause no performance deterioration when partitioned between sub-GCNs, **leading us to partition all feature dimensions other than d_0 and d_L** within the final `GIST` methodology; see Fig. 2b.
How many Sub-GCNs to use? Using more sub-GCNs during `GIST` training typically improves runtime because sub-GCNs (i) become smaller, (ii) are each trained for fewer epochs, and (iii) are trained in parallel. We find that **all models trained with `GIST` perform similarly for practical settings of m**; see Table 1. One may continue increasing the number sub-GCNs used within `GIST` until all GPUs are occupied or model performance begins to decrease.
`GIST` Performance. Models trained with `GIST` often exceed the performance of models trained with standard, single-GPU methodology; see Table 1. Intuitively, we hypothesize that the random feature partitioning within `GIST`, which loosely resembles dropout (Srivastava et al. 2014), provides regularization benefits during training, but some insight into the favorable performance of `GIST` is also provided by the theoretical guarantees outlined in Sect. 4.

5.2 Large-scale experiments

For large-scale experiments on Reddit and Amazon2M, the baseline model is trained on a single GPU and compared to models trained with `GIST` in terms of F1 score and training time. All large-scale graphs are partitioned into 15 000 sub-graphs during training.[5] Graph partitioning is mandatory because the training graphs are too large to fit into memory. One could instead use layer sampling to make training tractable (see

[5] Single-GPU training with graph partitioning via METIS is the same approach adopted by ClusterGCN Chiang et al. (2019), making our single-GPU baseline a ClusterGCN model. We adopt the same number of sub-graphs as proposed in this work.

Table 2 Performance of models trained with GIST on Reddit and Amazon2M

L	m	Reddit dataset					
		GraphSAGE			GAT		
		F1	Time (s)	Speedup	F1	Time (h)	Speedup
2	–	96.09	105.78	1.00×	89.57	1.19	1.00×
	2	96.40	70.29	1.50×	90.28	0.58	2.05×
	4	96.16	68.88	1.54×	90.02	0.31	3.86×
	8	95.46	76.68	1.38×	89.01	0.18	6.70×
3	–	96.32	118.37	1.00×	89.25	2.01	1.00×
	2	96.36	80.46	1.47×	89.63	0.95	2.11×
	4	95.76	78.74	1.50×	88.82	0.48	4.19×
	8	94.39	88.54	(1.34×)	70.38	0.26	(7.67×)
4	–	96.32	120.74	1.00×	88.36	2.77	1.00×
	2	96.01	91.75	1.32×	87.97	1.31	2.11×
	4	95.21	78.74	(1.53×)	78.42	0.66	(4.21×)
	8	92.75	88.71	(1.36×)	66.30	0.35	(7.90×)
L	m	Amazon2M Dataset					
		GraphSAGE ($d_i = 400$)			GraphSAGE ($d_i = 4\,096$)		
		F1	Time (h)	Speedup	F1	Time (h)	Speedup
2	–	89.90	1.81	1.00×	91.25	5.17	1.00×
	2	88.36	1.25	(1.45×)	90.70	1.70	3.05×
	4	86.33	1.11	(1.63×)	89.49	1.13	(4.57×)
	8	84.73	1.13	(1.61×)	88.86	1.11	(4.65×)
3	–	90.36	2.32	1.00×	91.51	9.52	1.00×
	2	88.59	1.56	(1.49×)	91.12	2.12	4.49×
	4	86.46	1.37	(1.70×)	89.21	1.42	(6.72×)
	8	84.76	1.37	(1.69×)	86.97	1.34	(7.12×)
4	–	90.40	3.00	1.00×	91.61	14.20	1.00×
	2	88.56	1.79	(1.68×)	91.02	2.77	5.13×
	4	87.53	1.58	(1.90×)	89.07	1.65	(8.58×)
	8	85.32	1.56	(1.93×)	87.53	1.55	(9.13×)

Parenthesis are placed around speedups achieved at a cost of >1 deterioration in F1 and m ="–" refers to the baseline. *Models trained with GIST train more quickly and achieve comparable F1 score to those trained with standard methodology. The performance benefits of GIST become more pronounced for wider models*

Sect. 5.4), but we adopt graph partitioning in most experiments because the implementation is simple and performs well.

Reddit Dataset. We perform tests with 256-dimensional GraphSAGE (Hamilton et al. 2017) and GAT (Veličković et al. 2017) models with two to four layers on Reddit; see "Appendix A.4" for more details. As shown in Table 2, utilizing GIST significantly accelerates GCN training (i.e., a 1.32× to 7.90× speedup). GIST performs best in

Table 3 Performance of GraphSAGE models of different widths trained with GIST on Amazon2M

L	m	F1 Score (Time in hours)				
		$d_i = 400$	$d_i = 4\,096$	$d_i = 8\,192$	$d_i = 16\,384$	$d_i = 32\,768$
2	-	89.38 (1.81)	90.58 (5.17)	OOM	OOM	OOM
	2	87.48 (1.25)	90.09 (1.70)	90.87 (2.76)	90.94 (9.31)	90.91 (32.31)
	4	84.82 (1.11)	88.79 (1.13)	89.76 (1.49)	90.10 (2.24)	90.17 (5.16)
	8	82.56 (1.13)	87.16 (1.11)	88.31 (1.20)	88.89 (1.39)	89.46 (1.76)
3	-	89.73 (2.32)	90.99 (9.52)	OOM	OOM	OOM
	2	87.79 (1.56)	90.40 (2.12)	90.91 (4.87)	91.05 (17.7)	OOM
	4	85.30 (1.37)	88.51 (1.42)	89.75 (2.07)	90.15 (3.44)	OOM
	8	82.84 (1.37)	86.12 (1.34)	88.38 (1.37)	88.67 (1.88)	88.66 (2.56)
4	-	89.77 (3.00)	91.02 (14.20)	OOM	OOM	OOM
	2	87.75 (1.79)	90.36 (2.77)	91.08 (6.92)	91.09 (26.44)	OOM
	4	85.32 (1.58)	88.50 (1.65)	89.76 (2.36)	90.05 (4.93)	OOM
	8	83.45 (1.56)	86.60 (1.55)	88.13 (1.61)	88.44 (2.30)	OOM

m ="-" refers to the baseline and "OOM" marks experiments that cause out-of-memory errors. *GIST enables training of higher-performing, ultra-wide models*

terms of F1 score with $m = 2$ sub-GCNs (i.e., $m = 4$ yields further speedups but F1 score decreases). Interestingly, *the speedup provided by GIST is more significant for models and datasets with larger compute requirements*. For example, experiments with the GAT architecture, which is more computationally expensive than GraphSAGE, achieve a near-linear speedup with respect to m.

Amazon2M Dataset. Experiments are performed with two, three, and four-layer GraphSAGE models (Hamilton et al. 2017) with hidden dimensions of 400 and 4 096 (we refer to these models as "narrow" and "wide", respectively). We compare the performance (i.e., F1 score and wall-clock training time) of GCN models trained with standard, single-GPU methodology to that of models trained with GIST; see Table 2. Narrow models trained with GIST have a lower F1 score in comparison to the baseline, but training time is significantly reduced. For wider models, GIST provides a more significant speedup (i.e., up to 7.12×) and tends to achieve comparable F1 score in comparison to the baseline, revealing that *GIST works best with wider models*.

Within Table 2, models trained with GIST tend to achieve a wall-clock speedup at the cost of a lower F1 score (i.e., observe the speedups marked with parenthesis in Table 2). When training time is analyzed with respect to a fixed F1 score, we observe that the baseline takes significantly longer than GIST to achieve a fixed F1 score. For example, when $L = 2$, a wide GCN trained with GIST ($m = 8$) reaches an F1 score of 88.86 in $\sim$ 4 000 seconds, *while models trained with standard methodology take* $\sim$ 10 000 *seconds to achieve a comparable F1 score.* As such, GIST significantly accelerates training relative to model performance.

5.3 Training ultra-wide GCNs

We use GIST to train GraphSAGE models with widths as high as 32 000 (i.e., **8×** beyond the capacity of a single GPU); see Table 3 for results and "Appendix A.5" for more details. Considering $L = 2$, the best-performing, single-GPU GraphSAGE model ($d_i = 4\,096$) achieves an F1 score of 90.58 in 5.2 hours. With GIST ($m = 2$), we achieve a higher F1 score of 90.87 in 2.8 hours (i.e., a 1.86× speedup) using $d_i = 8\,192$, which is beyond single GPU capacity. Similar patterns are observed for deeper models. Furthermore, we find that utilizing larger hidden dimensions yields further performance improvements, revealing the utility of wide, overparameterized GCN models. *GIST, due to its feature partitioning strategy, is unique in its ability to train models of such scale to state-of-the-art performance.*

5.4 GIST with layer sampling

As previously mentioned, some node partitioning approach must be adopted to avoid memory overflow when the underlying training graph is large. Although graph partitioning is used within most experiments (see Sect. 5.2), GIST is also compatible with other node partitioning strategies. To demonstrate this, we perform training on Reddit using GIST combined with a recent layer sampling approach (Zou et al. 2019) (i.e., instead of graph partitioning); see "Appendix A.6" for more details.

As shown in Table 4, combining GIST with layer sampling enables training on large-scale graphs, and the observed speedup actually exceeds that of GIST with graph partitioning. For example, GIST with layer sampling yields a 1.83× speedup when $L = 2$ and $m = 2$, in comparison to a 1.50× speedup when graph partitioning is used within GIST (see Table 2). As the number of sub-GCNs is increased beyond $m = 2$, GIST with layer sampling continues to achieve improvements in wall-clock training time (e.g., speedup increases from 1.83× to 2.90× from $m = 2$ to $m = 4$ for $L = 2$) without significant deterioration to model performance. Thus, although node partitioning is needed to enable training on large-scale graphs, the feature partitioning strategy of GIST is compatible with numerous sampling strategies (i.e., not just graph sampling).

6 Future work

There are a few notable extensions of GIST that may be especially useful to the research community. We leave these extensions as future work, as they go beyond the core focus of our proposal: formulating an easy-to-use, efficient training framework for large-scale experiments with GCNs.

GCNs with Edge Features. Recent work has explored using edge features within the GCN architecture (Gong and Cheng 2019; Jiang et al. 2020; Bergen et al. 2021). Given that GIST can be applied to any GCN architecture, we argue that (i) GIST is similarly compatible with architectural variants that exploit edge features and (ii) using edge features within the graph could yield further performance benefits.

Table 4 Performance of GCN models trained with a combination of GIST and LADIES Zou et al. (2019) on Reddit

L	# Sub-GCNs	GIST + LADIES		
		F1 Score	Time	Speedup
2	Baseline	89.73	3 359.91s	1.00×
	2	89.29	1 834.59s	1.83×
	4	88.42	1 158.51s	2.90×
3	Baseline	89.57	4 803.88s	1.00×
	2	86.52	2 635.18s	1.82×
	4	86.72	1 605.32s	3.00×

Here, the baseline represents models trained with LADIES in a standard, single-GPU manner. *Combining GIST with layer sampling leads to further improvements in wall-clock training time without deteriorating the F1 score*

To understand why such techniques would be compatible, we emphasize that—similar to node partitioning—edge features operate orthogonally to the model partitioning performed by GIST. For example, the EGNN model (Gong and Cheng 2019) injects edge information into the GCN model via the adjacency matrix at each layer, which modifies node representations and their relationships within the graph. As shown in Fig. 3, GIST simply partitions the feature space of each individual node within the hidden layers of the GCN, which has no impact or dependence on node or edge information within the underlying graph.

Deeper GCNs. Our analysis focuses upon the exploration of wide, but not deep, GCNs due to presence of oversmoothing in deep GCNs (Li et al. 2018). However, GIST is applicable to GCN architectures of any depth—the feature partitioning strategy is just applied separately to each layer. To further reduce the memory overhead of deeper GCN models, one could explore extensions of GIST that combine layer and feature partitioning strategies. Such a variant would independently train narrow sub-GCNs that contain only a small fraction of the global model's total layers. Layer partitioning strategies—without feature partitioning—have already been shown to be effective for IST-based training of convolutional neural networks with residual connections (Dun et al. 2022).

More Settings. The analysis of GIST could be extended to alternative tasks (e.g., link prediction) and larger-scale datasets. However, performing experiments over datasets larger than Amazon2M is difficult due to the lack of moderately-large-scale graphs that are available publicly. For example, the only graph larger than Amazon2M provided via the Open Graph Benchmark (Hu et al. 2020) is Papers100M, which requires 256 Gb of CPU RAM to load.

7 Conclusions

We present GIST, a distributed training approach for GCNs that enables the exploration of larger models and datasets. GIST is compatible with existing sampling approaches and leverages a feature-wise partition of model parameters to construct

smaller sub-GCNs that are trained independently and in parallel. We have shown that GIST achieves remarkable speed-ups over large graph datasets and even enables the training of GCN models of unprecedented size. We hope GIST can empower the exploration of larger, more powerful GCN architectures within the graph community.

Supplementary information

All code for this project is publicly-available via github at the following link: https://github.com/wolfecameron/GIST.

Acknowledgements This work is supported by NSF FET: Small No. 1907936, NSF MLWiNS CNS No. 2003137 (in collaboration with Intel), NSF CMMI no. 2037545, NSF CAREER award No. 2145629, NSF CIF No. 2008555 and Rice InterDisciplinary Excellence Award (IDEA).

Author Contributions All authors made substantial contributions to the work and adhere to the guidelines outlined for this publication. The primary authors have been marked on the title page with equal contribution.

Funding Funding for this project is provided by NSF FET: Small No. 1907936, NSF MLWiNS CNS No. 2003137 (in collaboration with Intel), NSF CMMI no. 2037545, NSF CAREER award no. 2145629, NSF CIF No. 2008555, and Rice InterDisciplinary Excellence Award (IDEA).

Data Availability All data used within the publication is openly-available to the public and can be downloaded for free from the internet.

Declarations

Conflict of interest The authors are not aware of any competing interests directly or indirectly related to this work.

Code Availability All code for this project is publicly-available via github at the following link: https://github.com/wolfecameron/GIST.

A Experimental details

A.1 Datasets

The details of the datasets utilized within GIST experiments in Sect. 5 are provided in Table 5. Cora, Citeseer, PubMed and OGBN-Arxiv are considered "small-scale" datasets and are utilized within experiments in Sect. 5.1. Reddit and Amazon2M are considered "large-scale" datasets and are utilized within experiments in Sect. 5.2.

A.2 Implementation details

We provide an implementation of GIST in PyTorch (Paszke et al. 2019) using the NCCL distributed communication package for training GCN (Kipf and Welling 2016), GraphSAGE (Hamilton et al. 2017) and GAT (Veličković et al. 2017) architectures. Our implementation is centralized, meaning that a single process serves as a central parameter server. From this central process, the weights of the global model are maintained and partitioned to different worker processes (including itself) for independent training. Experiments are conducted with 8 NVIDIA Tesla V100-PCIE-32 G GPUs, a 56-core Intel(R) Xeon(R) CPU E5-2680 v4 @ 2.40GHz, and 256 GB of RAM.

A.3 Small-scale experiments

Small-scale experiments in Sect. 5.1 are performed using Cora, Citeseer, Pubmed, and OGBN-Arxiv datasets (Sen et al. 2008; Hu et al. 2020). GIST experiments are performed with two, four, and eight sub-GCNs in all cases. We find that the performance of models trained with GIST is relatively robust to the number of local iterations ζ, but test accuracy decreases slightly as ζ increases; see Fig. 5. Based on the results in Fig. 5, we adopt $\zeta = 20$ for Cora, Citeseer, and Pubmed, as well as $\zeta = 100$ for OGBN-Arxiv.

Experiments are run for 400 epochs with a step learning rate schedule (i.e., $10\times$ decay at 50% and 75% of total epochs). A vanilla GCN model, as described in Kipf and Welling (2016), is used. The model is trained in a full-batch manner using the Adam optimizer (Kingma and Ba 2014). No node sampling techniques are employed because the graph is small enough to fit into memory. All reported results are averaged across five trials with different random seeds. For all models, d_0 and d_L are respectively given by the number of features and output classes in the dataset. The size of all hidden layers is the same, but may vary across experiments.

We first train baseline GCN models of different depths and hidden dimensions using a single GPU to determine the best model depth and hidden dimension to be used in small-scale experiments. The results are shown in Fig. 4. Deeper models do not yield performance improvements for small-scale datasets, but test accuracy improves as the model becomes wider. Based upon the results in Fig. 4, we adopt a three-layer GCN with a hidden dimension of $d_1 = d_2 = 256$ as the underlying model used in small-scale experiments. Though two-layer models seem to perform best, we use a three-layer

Table 5 Details of relevant datasets

Dataset	n	# Edges	# Labels	d
Cora	2 708	5 429	7	1,433
CiteSeer	3 312	4 723	6	3,703
Pubmed	19 717	44 338	3	500
OGBN-Arxiv	169 343	1.2M	40	128
Reddit	232 965	11.6 M	41	602
Amazon2M	2.5 M	61.8 M	47	100

model within Sect. 5.1 to enable more flexibility in examining the partitioning strategy of GIST.

A.4 Large-scale experiments

Reddit Dataset. For experiments on Reddit, we train 256-dimensional GraphSAGE and GAT models using both GIST and standard, single-GPU methodology. During training, the graph is partitioned into 15 000 sub-graphs. Training would be impossible without such partitioning because the graph is too large to fit into memory. The setting for the number of sub-graphs is the optimal setting proposed in previous work (Chiang et al. 2019). Models trained using GIST and standard, single-GPU methodologies are compared in terms of F1 score and training time.

All tests are run for 80 epochs with no weight decay, using the Adam optimizer (Kingma and Ba 2014). We find that $\zeta = 500$ achieves consistently high performance for models trained with GIST on Reddit. We adopt a batch size of 10 sub-graphs throughout the training process, which is the optimal setting proposed in previous work (Chiang et al. 2019).

Amazon2M Dataset. For experiments on Amazon2M, we train two to four layer GraphSAGE models with hidden dimensions of 400 and 4 096 using both GIST and standard, single-GPU methodology. We follow the experimental settings of Chiang et al. (2019). The training graph is partitioned into 15000 sub-graphs and a batch size of 10 sub-graphs is used. We find that using $\zeta = 5000$ performs consistently well.

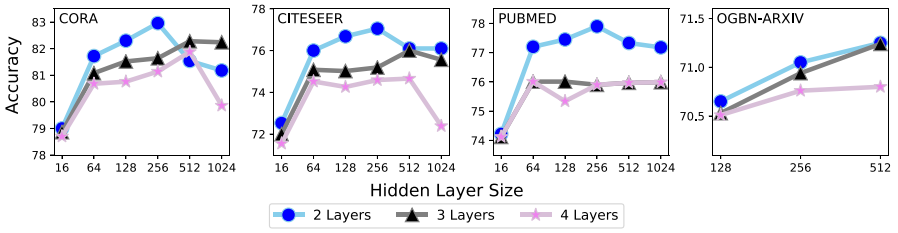

Fig. 4 Test accuracy for different sizes (i.e., varying depth and width) of GCN models trained with standard, single-GPU methodology on small-scale datasets. *We adopt three-layer, 256-dimensional GCN models as our baseline architecture*

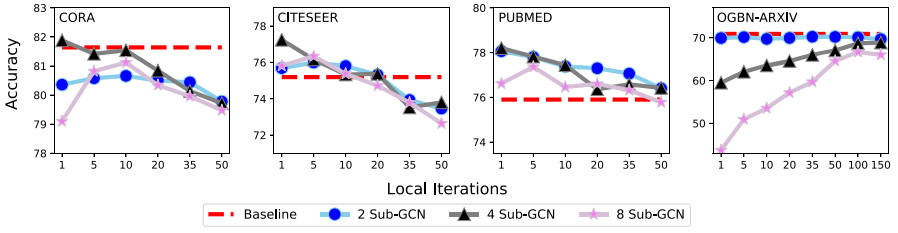

Fig. 5 Test accuracy of GCN models trained on small-scale datasets with GIST using different numbers of local iterations and sub-GCNs. *Models trained with GIST are surprisingly robust to the number of local iterations used during training, no matter the number of sub-GCNs*

Models are trained for 400 total epochs with the Adam optimizer (Kingma and Ba 2014) and no weight decay.

A.5 Training ultra-wide GCNs

All settings for ultra-wide GCN experiments in Sect. 5.3 are adopted from the experimental settings of Sect. 5.2; see "Appendix A.4" for further details. For $d_i > 4096$ evaluation must be performed on graph partitions (not the full graph) to avoid memory overflow. As such, the graph is partitioned into 5000 sub-graphs during testing and F1 score is measured over each partition and averaged. All experiments are performed using a GraphSAGE model, and the hidden dimension of the underlying model is changed between different experiments.

A.6 `GIST` with layer sampling

Experiments in Sect. 5.4 adopt the same experimental settings as Sect. 5.2 for the Reddit dataset; see "Appendix A.4" for further details. Within these experiments, we combine `GIST` with LADIES (Zou et al. 2019), a recent layer sampling approach for efficient GCN training. LADIES is used instead of graph partitioning. Any node sampling approach can be adopted—some sampling approach is just needed to avoid memory overflow.

We train 256-dimensional GCN models with either two or three layers. We utilize a vanilla GCN model within this section (as opposed to GraphSAGE or GAT) to simplify the implementation of `GIST` with LADIES, which creates a disparity in F1 score between the results in Sects. 5.2 and 5.4. Experiments in Sect. 5.4 compare the performance of the same models trained either with `GIST` or using standard, single-GPU methodology. In this case, the single-GPU model is just a GCN trained with LADIES.

B `GIST` vs. other distributed training methods

Although `GIST` has been shown to provide benefits in terms of GCN performance and training efficiency in comparison to standard, single-GPU training, other choices for the distributed training of GCNs exist. Within this section, we compare `GIST` to other natural choices for distributed training, revealing that GCN models trained with `GIST` achieve favorable performance in comparison to those trained with other common distributed training techniques.

B.1 Local SGD

A simple version of local SGD (Lin et al. 2018) can be implemented for distributed training of GCNs by training the full model on each separate worker for a certain number of local iterations and intermittently averaging local updates. In comparison to such a methodology, `GIST` has better computational and communication efficiency

Table 6 Performance of GraphSAGE models trained using local SGD and GIST on Reddit

# Machines	Method	F1 Score	Training time
2	Local SGD	96.37	137.17s
	GIST	**96.40**	**108.67s**
4	Local SGD	95.00	127.63s
	GIST	**96.16**	**116.56s**
8	Local SGD	93.40	129.58s
	GIST	**95.46**	**123.83s**

We adopt settings described in Sect. 5.2, but use 100 local iterations for both GIST and local SGD. *Models trained with GIST outperform those trained with local SGD in terms of test F1 score and wall-clock training time in all cases*

Table 7 Performance of GraphSAGE models trained both with GIST and as ensembles of shallow sub-GCNs on Reddit

# Machines	Method	F1 Score	Inference time
2	Ensemble	96.31	3.59s
	GIST	**96.40**	**1.81s**
4	Ensemble	96.10	6.38s
	GIST	**96.16**	**1.81s**
8	Ensemble	95.28	11.95s
	GIST	**95.46**	**1.81s**

Models trained with *GIST* perform better and do not suffer from increased inference time as the number of sub-GCNs is increased

because *i*) it communicates only a small fraction of model parameters to each machine and *ii*) locally training narrow sub-GCNs is faster than locally training the full model. We perform a direct comparison between local SGD and GIST on the Reddit dataset using a two-layer, 256-dimensional GraphSAGE model; see Table 6. As can be seen, GCN models trained with GIST have lower wall-clock training time and achieve better performance than those trained with local SGD in all cases.

B.2 Sub-GCN ensembles

As previously mentioned, increasing the number of local iterations (i.e., ζ in Algorithm 1) decreases communication requirements given a fixed amount of training. When taken to the extreme (i.e., $\zeta \rightarrow \infty$), one could minimize communication requirements by never aggregating sub-GCN parameters, thus forming an ensemble of independently-trained sub-GCNs. We compare GIST to such a methodology[6] in Table 7 using a two-layer, 256-dimensional GraphSAGE model on the Reddit dataset. Though training ensembles of sub-GCNs minimizes communication, Table 7 reveals that (i) models trained with GIST achieve better performance and (ii) inference time for sub-GCN ensembles becomes burdensome as the number of sub-GCNs is increased.

[6] For each sub-GCN, we measure validation accuracy throughout training and add the highest-performing model into the ensemble.

C Theoretical results

C.1 Formulation of GIST for one-hidden-layer GCNs

In our analysis, we consider a GCN with one hidden-layer and a ReLU activation. Given training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ with input features $\mathbf{x}_i \in \mathbb{R}^d$ and labels $y_i \in \mathbb{R}$, we assume that the GCN outputs a scalar value $\tilde{y}_i$ for each node in the graph. Denoting $\tilde{\mathbf{y}} = [\tilde{y}_1, \ldots, \tilde{y}_n]$, we can write the output of the GCN as

$$\tilde{\mathbf{y}} = \frac{1}{\sqrt{d_1}} \bar{\mathbf{A}} \sigma\left(\bar{\mathbf{A}}\mathbf{X}\boldsymbol{\Theta}\right)\mathbf{a}$$

where $\boldsymbol{\Theta} = [\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_{d_1}] \in \mathbb{R}^{n \times d_1}$ is the weights within the GCN's first layer and $\mathbf{a} = [a_1, \ldots, a_{d_1}] \in \mathbb{R}^{d_1}$ is the weights within the GCN's second layer. To simplify the analysis, we denote $\hat{\mathbf{X}} = \bar{\mathbf{A}}\mathbf{X} = [\hat{\mathbf{x}}_1, \ldots \hat{\mathbf{x}}_n]$. Then, we have $\hat{\mathbf{x}}_i = \sum_{i'=1}^n \bar{\mathbf{A}}_{ii'}\mathbf{x}_{i'}$ and the output of each node within the graph can be written as

$$\tilde{y}_i = \frac{1}{\sqrt{d_1}} \sum_{i'=1}^{n} \sum_{r=1}^{d_1} \bar{\mathbf{A}}_{ii'} a_r \sigma\left(\left\langle \hat{\mathbf{x}}_{i'}, \boldsymbol{\theta}_r \right\rangle\right)$$

As in previous convergence analysis for training neural networks, we assume that second-layer weights $\mathbf{a}$ are fixed and only the first layer weights $\boldsymbol{\Theta}$ are trainable. Following the GIST feature partitioning strategy, we only partition the hidden layer. Specifically, in global iteration t, sub-GCNs are constructed by sampling a set of masks $\mathcal{M}_t \in \mathbb{R}^{m \times d_1}$. We denote the jth column of $\mathcal{M}_t$ as $\mathcal{M}_t^{(j)} \in \mathbb{R}^m$, the rth row of $\mathcal{M}_t$ as $\mathcal{M}_{t,r} \in \mathbb{R}^{d_1}$, and the entry in the rth row and jth column as $\mathcal{M}_{t,r}^{(j)}$. Each $\mathcal{M}_{t,r}^{(j)}$ is a binary values: $\mathcal{M}_{t,r}^{(j)} = 1$ if neuron r is active in sub-GCN j, and $\mathcal{M}_{t,r}^{(j)} = 0$ otherwise. Using this mask notation, the output for node i within sub-GCN j can be written as

$$\hat{y}_i^{(j)}(t,k) = f_{\mathcal{M}_t^{(j)}}\left(\boldsymbol{\Theta}_{t,k}^{(j)}, \mathbf{x}\right)_i = \frac{1}{\sqrt{d_1}} \sum_{i'=1}^{n} \sum_{r=1}^{d_1} \bar{\mathbf{A}}_{ii'} \mathcal{M}_{t,r}^{(j)} a_r \sigma\left(\left\langle \hat{\mathbf{x}_{i'}}, \boldsymbol{\theta}_{t,k,r}^{(j)} \right\rangle\right) \tag{3}$$

t and k denote the current global and local iterations, respectively. We assume that each $\mathcal{M}_{t,r}$ is sampled from a one-hot categorical distribution. We formally define the random variables $\mathcal{M}_{t,r}^{(j)}$ as follows: for each t, let $\{\hat{m}_{t,r}\}_{r=1}^{d_1}$ be a set of I.I.D. uniform random variables taking values on the index set $[m] = \{1, \ldots, m\}$, *i.e.*, $\mathcal{P}\left(\hat{m}_{t,r} = j\right) = \frac{1}{m}$ for $j \in [m]$. Then, we define each mask entry as $\mathcal{M}_{t,r}^{(j)} = \mathbb{I}\left\{\hat{m}_{k,r} = j\right\}$. Masks sampled in such a fashion have the following properties

- $\mathcal{P}\left(\mathcal{M}_{t,r}^{(j)} = 1\right) = \frac{1}{m}$
- $\mathcal{P}\left(\mathcal{M}_{t,r}^{(j)} = 0\right) = 1 - \frac{1}{m}$
- $\sum_{j=1}^m \mathcal{M}_{t,r}^{(j)} = 1$

- $\mathcal{M}_{t,r}^{(j)}\mathcal{M}_{t,r}^{(j')} = 0$ if $j' \neq j$.

Here, the first and second properties guarantee that the expected number of neurons active in each sub-GCN is equal. The third and fourth properties guarantee that each neuron is active in one and only one sub-GCN. Within this setup, we consider the GIST training procedure, described as

$$\begin{aligned} \boldsymbol{\theta}_{t,0,r}^{(j)} &= \boldsymbol{\theta}_{t,r} \\ \boldsymbol{\theta}_{t,k+1,r}^{(j)} &= \boldsymbol{\theta}_{t,k,r}^{(j)} - \eta \frac{\partial L\left(\boldsymbol{\Theta}_{t,k}^{(j)}\right)}{\partial \boldsymbol{\theta}_r} \\ \boldsymbol{\theta}_{t+1,r} &= \boldsymbol{\theta}_{t,r} + \sum_{j=1}^{m}\left(\boldsymbol{\theta}_{t,\zeta,r}^{(j)} - \boldsymbol{\theta}_{t,0,r}^{(j)}\right) \end{aligned} \tag{4}$$

Within this formulation, ζ represents the total number of local iterations performed for each sub-GCN, while $L\left(\boldsymbol{\Theta}_{t,k}^{(j)}\right)$ is the loss on the jth sub-GCN during the t-th global and kth local iteration. We can express $L\left(\boldsymbol{\Theta}_{t,k}^{(j)}\right)$ as

$$L\left(\boldsymbol{\Theta}_{t,k}^{(j)}\right) = \left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2 = \left\|\mathbf{y} - f_{\mathcal{M}_t^{(j)}}\left(\boldsymbol{\Theta}_{t,k}^{(j)}, \mathbf{x}\right)\right\|_2^2$$

and the gradient has the form

$$\frac{\partial L\left(\boldsymbol{\Theta}_{t,k}^{(j)}\right)}{\partial \boldsymbol{\theta}_r} = \frac{1}{\sqrt{d_1}} \sum_{i=1}^{n}\sum_{i'=1}^{n}\left(\hat{y}_i^{(j)}(t,k) - y_i\right)\bar{\mathbf{A}}_{ii'}\mathcal{M}_{t,r}^{(j)} a_r \hat{\mathbf{x}}_{i'} \mathbb{I}\left\{\left\langle \boldsymbol{\theta}_{t,k,r}^{(j)}, \hat{\mathbf{x}}_{i'}\right\rangle \geq 0\right\}$$

C.2 Preliminary and notations

We use bold lower-case letters (e.g. $\mathbf{a}$) to denote vectors, bold upper-case letters (e.g. $\mathbf{A}$) to denote matrices, and standard letters (e.g. a) for scalars. For a vector $\mathbf{a}$, $\|\mathbf{a}\|_1$ and $\|\mathbf{a}\|_1$ denote its ℓ_1 and ℓ_2 norm, respectively. We also use $\texttt{abs}(\mathbf{a}) = [|a_1|, \ldots, |a_n|]$ to denote the vector with entry-wise absolute value of $\mathbf{a}$. For a matrix $\mathbf{A}$, we use $\|\mathbf{A}\|_2$, $\|\mathbf{A}\|_F$, $\|\mathbf{A}\|_{1,1}$ to denote its operator norm, Frobenius norm, and $L_{1,1}$ norm, respectively. We denote $\mathbb{E}_{[\mathcal{M}_t]}[\cdot] = \mathbb{E}_{\mathcal{M}_0,\ldots,\mathcal{M}_t}[\cdot]$ to denote the total expectation with respect to $\mathcal{M}_0, \ldots, \mathcal{M}_t$. For a full description of the symbols used in the proof, please refer to Table 8.

In the proof, we will also utilize the following probability tools

Property 1 (Markov's Inequality) *For a non-negative random variable X, we have*

$$\mathcal{P}\left(X \geq a\right) \leq \frac{1}{a}\mathbb{E}[X]$$

Table 8 Table of Notations

Symbol	Description	Mathematical definition
T	Number of global iterations	$T \in \mathbb{N}_+$
t	Index of global iterations	$t \in \{0\} \cup [T]$
ζ	Number of local iterations	$\zeta \in \mathbb{N}_+$
k	Index of global iterations	$k \in \{0\} \cup [\zeta]$
m	Number of sub-GCNs	$m \in \mathbb{N}_+$
n	Number of training samples	$n \in \mathbb{N}_+$
d	Dimension of input feature	$d \in \mathbb{N}_+$
d_1	Dimension of hidden layer	$d_1 \in \mathbb{N}_+$
$\{\mathbf{x}_i\}_{i=1}^n$	Input training data	$\mathbf{x}_i \in \mathbb{R}^d$
$\{y_i\}_{i=1}^n$	Training labels	$y_i \in \mathbb{R}, \lvert y_i \rvert \leq C$
$\mathcal{M}_t$	Binary Mask in iteration t	$\mathcal{M}_t \in \{0,1\}^{m \times d_1}$
$\mathcal{M}_{t,r}^{(j)}$	The (r, j)th entry of $\mathcal{M}_t$	$\mathcal{M}_{t,r} \in \{0,1\}$
η	Constant step size	$\eta = \mathcal{O}\left(\frac{\lambda_0}{n\lVert\bar{\mathbf{A}}^2\rVert_{1,1}}\right)$
$\hat{y}_i^{(j)}$	Output of node i within sub-GCN j	See Eq. (3)
$\hat{\mathbf{y}}^{(j)}$	Output for all nodes within sub-GCN j	$[\hat{y}_1^{(j)}, \ldots, \hat{y}_n^{(j)}]$
$\hat{y}_i$	Output of node i of the whole GCN	See Eq. (15)
$\hat{\mathbf{y}}$	Output for all nodes of the whole GCN	$[\hat{y}_1, \ldots, \hat{y}_n]$
$\bar{\mathbf{A}}$	Aggregation matrix of the GCN	$\bar{\mathbf{A}} = \mathbf{I} + \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$
$\mathbf{G}^\infty$	Infinite-width `GIST-K` of the GCN	See Eq. (7)
λ_0	Mimimum eigenvalue of $\mathbf{G}^\infty$	$\lambda_0 = \lambda_{\min}\left(\mathbf{G}^\infty\right)$
λ^*	Maximum eigenvalue of $\mathbf{G}^\infty$	$\lambda^* = \lambda_{\max}\left(\mathbf{G}^\infty\right)$

Property 2 (Jensen's Inequality for Expectation) *For a non-negative random variable X, we have*

$$\mathbb{E}[X^{\frac{1}{2}}] \leq \mathbb{E}[X]^{\frac{1}{2}}$$

C.3 Properties of the transformed input

The GCN (Kipf and Welling 2016) uses a first-degree Chebyshev polynomial to approximate a spectral convolution on the graph, which results in an aggregation matrix of the form

$$\bar{\mathbf{A}} = \mathbf{I} + \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}} \tag{5}$$

where $\mathbf{A}$ is the adjacency matrix and $\mathbf{D}$ is the degree matrix with $\mathbf{D}_{ii} = \sum_{j=1}^n \mathbf{A}_{ij}$. This is the same form of the aggregation matrix in Eq. (7) of Kipf and Welling (2016). In practice, the re-normalization trick is applied to control the magnitude of the largest eigenvalue of $\bar{\mathbf{A}}$ (see Eq. (8) of Kipf and Welling (2016)). Here, however, we keep the

original formulation of (5) to facilitate our analysis, and our assumption on the depth of the GCN does not lead to numerical instability even if $\lambda_{\max}(\bar{\mathbf{A}}) > 1$. Moreover, the definition of $\bar{\mathbf{A}}$ connects with the degree-normalized Laplacian $\mathcal{L}$ in the sense that $\bar{\mathbf{A}} = 2\mathbf{I} - \mathcal{L}$. It is a well-known result that $2 = \lambda_{\max}(\bar{\mathbf{A}}) \geq \lambda_{\min}(\bar{\mathbf{A}}) \geq 0$. In particular, the lower bound on the minimum eigenvalue is obtained by considering

$$\mathbf{v}^\top \bar{\mathbf{A}}\mathbf{v} = \sum_{i=1}^{n} v_i^2 + \sum_{(i,j)\in E} \frac{v_i v_j}{\sqrt{\mathbf{D}_{ii}\mathbf{D}_{jj}}} = \sum_{(i,j)\in E} \left(\frac{v_i}{\sqrt{\mathbf{D}_{ii}}} + \frac{v_j}{\sqrt{\mathbf{D}_{jj}}} \right)^2$$

In our analysis, we require the aggregation matrix $\bar{\mathbf{A}}$ to be positive definite. Thus, the following assumption can be made about $\lambda_{\min}(\bar{\mathbf{A}})$.

Assumption 2 $\lambda_{\min}(\bar{\mathbf{A}}) \neq 0$.

Going further, we must make a few more assumptions about the aggregation matrix and the graph itself to satisfy certain properties relevant to the analysis.

Assumption 3 There exists $\epsilon \in (0, 1)$ and $p \in \mathbb{Z}_+$ such that for all $i \in [n]$

$$(1-\epsilon)^2 p \leq \mathbf{D}_{ii} \leq (1+\epsilon)^2 p$$

Assumption 3 implies the following property

Property 3 *For all $i \in [n]$, we have $\left\|\hat{\mathbf{x}}_i\right\|_2 \leq 1$.*

Proof of Property 3 Under assumption 3, we have that for all $i, i' \in [n]$

$$\left(\frac{1-\epsilon}{1+\epsilon}\right)^2 \leq \frac{\mathbf{D}_{ii}}{\mathbf{D}_{i'i'}} \leq \left(\frac{1+\epsilon}{1-\epsilon}\right)^2$$

Therefore, we can write

$$\begin{aligned}
\left\|\hat{\mathbf{x}}_i\right\|_2 &= \left\| \sum_{i'=1}^{n} \bar{\mathbf{A}}_{ii'} \mathbf{x}_i \right\|_2 \\
&= \left\| \sum_{i'=1}^{n} \left(\mathbf{I} + \mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}} \right)_{ii'} \mathbf{x}_i \right\|_2 \\
&= \left\| \mathbf{x}_i + \mathbf{D}_{ii}^{-\frac{1}{2}} \sum_{i' \neq i} \mathbf{A}_{ii'} \mathbf{D}_{i'i'}^{-\frac{1}{2}} \mathbf{x}_i \right\|_2 \\
&\leq \frac{1-\epsilon}{2} + \mathbf{D}_{ii}^{-\frac{1}{2}} \sum_{i' \neq i} \mathbf{A}_{ii'} \mathbf{D}_{i'i'}^{-\frac{1}{2}} \left(\frac{1-\epsilon}{2} \right) \\
&\leq \frac{1-\epsilon}{2} + \mathbf{D}_{ii}^{-\frac{1}{2}} \sum_{i' \neq i} \mathbf{A}_{ii'} \mathbf{D}_{ii}^{-\frac{1}{2}} \left(\frac{1+\epsilon}{1-\epsilon} \right) \left(\frac{1-\epsilon}{2} \right) \\
&= 1
\end{aligned}$$

where the first inequality follows from Assumption 4 and the triangle inequality.

Additionally, we make the following assumption regarding the graph itself

Assumption 4 For all $i \in [n]$, we have $\|\mathbf{x}_i\|_2 \leq \frac{1-\epsilon}{2}$, and $|y_i| \leq C$ for some constant C. Moreover, for all $j \in [n]$ and $j \neq i$, we have $[\bar{\mathbf{A}}\mathbf{X}]_i \not\parallel [\bar{\mathbf{A}}\mathbf{X}]_j$.

C.4 Full statement theorem 1

We now state the full version of Theorem 1 from Sect. 4, which characterizes the convergence properties of one-hidden-layer GCN models trained with GIST.

Theorem 2 *Suppose assumptions 2–4 hold. Moreover, suppose in each global iteration the masks are generated from a categorical distribution with uniform mean $1/m$. Fix the number of global iterations to T and local iterations to ζ. Consider a two-layer GCN with parameters $\boldsymbol{\Theta}$. If each entry of $\boldsymbol{\Theta}$ is initialized I.I.D. from $\mathcal{N}(0, \kappa^2\mathbf{I})$, and the number of hidden neurons satisfies $d_1 \geq \Omega\left(\frac{T^2\zeta^2 n}{\lambda_0^4(1-\gamma)^2} \max\left\{\frac{n^3}{\delta^2\kappa^2}, \frac{n^2 d}{\delta^2}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}, T^2\lambda^{*2}d\right\}\right)$, then procedure (4) with constant step size $\eta = \mathcal{O}\left(\frac{\lambda_0}{n\left\|\bar{\mathbf{A}}^2\right\|_{1,1}}\right)$ converges according to*

$$\mathbb{E}_{[\mathcal{M}_{t-1}]}\left[\left\|\mathbf{y} - \hat{\mathbf{y}}(t)\right\|_2^2\right] \leq \left(\gamma + (1-\gamma)\left(1 - \frac{\eta\lambda_0}{2}\right)^{\zeta}\right)^t \left\|\mathbf{y} - \hat{\mathbf{y}}(0)\right\|_2^2 + \mathcal{O}\left(\frac{\gamma^2 d\kappa^2\lambda^{*2}}{m^2(1-\gamma)\lambda_0^2}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}\right)$$

with probability at least $1-\delta$, where $\gamma = \left(1 - m^{-1}\right)^{\frac{1}{3}}$.

In the following subsections, we will provide a proof for Theorem 2 In Sect. C.5, we first show that the local training of each sub-GCN enjoys linear convergence (Theorem 3). In Sect. C.6, we use this result to show Theorem 2. In Sect. C.7, we provide proof for the auxiliary lemmas.

C.5 GIST and local training progress

In this section, our goal is to show the following theorem

Theorem 3 *Suppose the number of hidden nodes satisfies $d_1 = \Omega\left(\lambda_0^{-1}n^2\log^{Tmn}/_{\delta}\right)$. If for all $r \in [d_1]$ it holds that*

$$\left\|\boldsymbol{\theta}_{t,r} - \boldsymbol{\theta}_{0,r}\right\|_2 + \frac{4T\eta\zeta}{\delta\alpha}\sqrt{\frac{n}{d_1}}\mathbb{E}_{[\mathcal{M}_{t-1}],\mathbf{W}_0,\mathbf{A}}\left[\left\|\mathbf{y} - \hat{\mathbf{y}}(t)\right\|_2^2\right]^{\frac{1}{2}} + (T-t)B \leq R \quad (6)$$

with

$$B = 4\eta\zeta\kappa\sqrt{\frac{Tdn(m-1)}{d_1 m\delta}}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}} + \frac{20T\zeta\gamma\kappa\lambda^*}{m\delta\alpha}\sqrt{\frac{\eta^3 nd}{\lambda_0 d_1}}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}; \quad R \leq \frac{\lambda_0}{192n}$$

then we have

$$\left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t, k+1)\right\|_2^2 \leq \left(1 - \frac{\eta\lambda_0}{2}\right)\left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t, k)\right\|_2^2$$

and for all $r \in [d_1]$, $j \in [m]$ *it holds that*

$$\left\|\boldsymbol{\theta}_{t,\zeta,r}^{(j)} - \boldsymbol{\theta}_{0,r}^{(j)}\right\|_2 \leq \frac{2T\eta\zeta}{\delta}\sqrt{\frac{n}{d_1}}\mathbb{E}_{[\mathcal{M}_{t-1}],\mathbf{W}_0,\mathbf{A}}\left[\left\|\mathbf{y} - \hat{\mathbf{y}}(t)\right\|_2^2\right]^{\frac{1}{2}} + 4\eta\zeta\kappa\sqrt{\frac{Tdn(m-1)}{d_1 m\delta}}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}$$

with probability at least $1 - \frac{\delta}{T}$

For a one-hidden-layer MLP, the analysis often depends on the (scaled) Gram Matrix of the infinite-dimensional NTK

$$\mathbf{H}_{ij}^{\infty} = \frac{1}{d_1 m}\left\langle\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j\right\rangle\mathbb{E}_{\boldsymbol{\theta}\sim\mathcal{N}(0,\mathbf{I})}\left[\mathbb{I}\{\left\langle\hat{\mathbf{x}}_i, \boldsymbol{\theta}\right\rangle \geq 0, \left\langle\hat{\mathbf{x}}_j, \boldsymbol{\theta}\right\rangle \geq 0\}\right] \tag{7}$$

We can extend this definition of the Gram Matrix to an infinite-width, one-hidden-layer GCN as follows

$$\mathbf{G}^{\infty} = \bar{\mathbf{A}}\mathbf{H}^{\infty}\bar{\mathbf{A}}$$

With Assumption 4, prior work (Du et al. 2019) shows that $\lambda_{\min}(\mathbf{H}) > 0$. Since $\bar{\mathbf{A}}$ is also positive definite, we must have that $\mathbf{G}^{\infty}$ is at least positive semidefinite. Moreover, for any $\mathbf{v} \in \mathbb{R}^n$ such that $\mathbf{v} \neq 0$, we must have that $\bar{\mathbf{A}}\mathbf{v} \neq 0$, and thus $\mathbf{H}^{\infty}\bar{\mathbf{A}}\mathbf{v} \neq 0$, which implies that $\bar{\mathbf{A}}\mathbf{H}^{\infty}\bar{\mathbf{A}}\mathbf{v} \neq 0$. Thus, for any $\mathbf{v} \neq 0$, $\mathbf{G}^{\infty}\mathbf{v} \neq 0$. Therefore, we must have that $\lambda_{\min}(\mathbf{G}^{\infty}) > 0$. In our analysis, we define the Graph Independent Subnetwork Tangent Kernel (GIST-K)

$$\mathbf{G}^{(j)}(t, t', k) = \bar{\mathbf{A}}\mathbf{H}(t, t', k)\bar{\mathbf{A}}$$

where $\mathbf{H}(t, t', k)$ is defined as

$$\mathbf{H}(t, t', k) = \frac{1}{d_1}\left\langle\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j\right\rangle\sum_{r=1}^{d_1}\mathcal{M}_{t,r}^{(j)}\mathbb{I}\left\{\left\langle\hat{\mathbf{x}}_i, \boldsymbol{\theta}_{t',k,r}^{(j)}\right\rangle \geq 0, \left\langle\hat{\mathbf{x}}_j, \boldsymbol{\theta}_{t',k,r}^{(j)}\right\rangle \geq 0\right\}$$

for masks $\mathcal{M}_t$ and weights $\mathbf{\Theta}^{(j)}_{t',k}$. Following previous work (Liao and Kyrillidis 2021) on subnetwork theory, we obtained Lemma 3, which shows that, if $d_1 = \Omega\left(\lambda_0^{-1} n^2 \log {}^{Tmn}/_{\delta}\right)$, and for all t, k it holds that $\left\|\boldsymbol{\theta}_{t,k,r} - \boldsymbol{\theta}_{0,r}\right\|_2 \leq R := \frac{\kappa\lambda_0}{192n}$, then with probability at least $1-\delta$, for all $t, t' \in [T]$ we have:

$$\lambda_{\min}(\mathbf{G}^{(j)}(t, t', k)) \geq \tfrac{\lambda_0}{2}$$

The lemma above shows that every `GIST-K` is positive definite. To proceed, the proof for the linear convergence relies on the following quadratic expansion of the loss

$$\begin{aligned}\left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t, k+1)\right\|_2^2 = \left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t, k)\right\|_2^2 + \left\|\hat{\mathbf{y}}^{(j)}(t, k+1) - \hat{\mathbf{y}}^{(j)}(t, k)\right\|_2^2 \\ - 2\left\langle \mathbf{y} - \hat{\mathbf{y}}^{(j)}(t, k), \hat{\mathbf{y}}^{(j)}(t, k+1) - \hat{\mathbf{y}}(t, k)\right\rangle\end{aligned} \tag{8}$$

Thus, it is natural to focus on the change of the output of the sub-GCN in each local iterations $\hat{y}_i^{(j)}(t, k+1) - \hat{y}_i^{(j)}(t, k)$. The following lemma shows that this term is bounded

Analyzing the inner-product term on the right-hand side of Eq. (8) involves a detailed study of the change of activation pattern $\mathbb{I}\left\{\left\langle \boldsymbol{\theta}, \hat{\mathbf{x}}_i\right\rangle \geq 0\right\}$ Then, following Song and Yang (2020), we first fix $R = \frac{\kappa\lambda_0}{192n}$, and denote

$$\begin{aligned} A_{ir} &= \exists \boldsymbol{\theta} : \left\|\boldsymbol{\theta} - \boldsymbol{\theta}_{0,r}\right\|_2 \leq R, \mathbb{I}\{\left\langle \boldsymbol{\theta}, \hat{\mathbf{x}}_i\right\rangle \geq 0\} \neq \mathbb{I}\{\left\langle \boldsymbol{\theta}_{0,r}, \hat{\mathbf{x}}_i\right\rangle \geq 0\} \\ S_i &= \{r \in [m] : \neg A_{ir}\}; \quad S_i^{\perp} = [m] \setminus S_i \end{aligned}$$

Based on the definition of $\hat{\mathbf{y}}(t, \cdot)$, we can write

$$\begin{aligned}&\hat{y}_i^{(j)}(t, k+1) - \hat{y}_i^{(j)}(t, k) \\ &= \frac{1}{\sqrt{d_1}} \sum_{i'=1}^{n} \sum_{r=1}^{d_1} \bar{\mathbf{A}}_{ii'} \mathcal{M}_{t,r}^{(j)} a_r \left(\sigma\left(\left\langle \boldsymbol{\theta}_{t,k+1,r}^{(j)}, \hat{\mathbf{x}}_{i'}\right\rangle\right) - \sigma\left(\left\langle \boldsymbol{\theta}_{t,k,r}^{(j)}, \hat{\mathbf{x}}_{i'}\right\rangle\right)\right)\end{aligned}$$

Thus, we can decompose $\hat{y}_i^{(j)}(t, k+1) - \hat{y}_i^{(j)}(t, k) = I_{i,1}^{(j)}(t, k) + I_{i,2}^{(j)}(t, k)$ with

$$\begin{aligned} I_{i,1}^{(j)}(t, k) &= \frac{1}{\sqrt{d_1}} \sum_{i'=1}^{n} \sum_{r \in S_{i'}} \bar{\mathbf{A}}_{ii'} \mathcal{M}_{t,r}^{(j)} a_r \cdot \\ &\qquad \left(\sigma\left(\left\langle \boldsymbol{\theta}_{t,k+1,r}^{(j)}, \hat{\mathbf{x}}_{i'}\right\rangle\right) - \sigma\left(\left\langle \boldsymbol{\theta}_{t,k,r}^{(j)}, \hat{\mathbf{x}}_{i'}\right\rangle\right)\right) \\ I_{i,2}^{(j)}(t, k) &= \frac{1}{\sqrt{d_1}} \sum_{i'=1}^{n} \sum_{r \in S_{i'}^{\perp}} \bar{\mathbf{A}}_{ii'} \mathcal{M}_{t,r}^{(j)} a_r \cdot \\ &\qquad \left(\sigma\left(\left\langle \boldsymbol{\theta}_{t,k+1,r}^{(j)}, \hat{\mathbf{x}}_{i'}\right\rangle\right) - \sigma\left(\left\langle \boldsymbol{\theta}_{t,k,r}^{(j)}, \hat{\mathbf{x}}_{i'}\right\rangle\right)\right) \end{aligned}$$

Let $\mathbf{I}_1^{(j)}(t,k) = \left[I_{1,1}^{(j)}(t,k), \ldots, I_{n,1}^{(j)}(t,k)\right]$ and similarly $\mathbf{I}_2^{(j)}(t,k) = \left[I_{1,2}^{(j)}(t,k), \ldots, I_{n,2}^{(j)}(t,k)\right]$ be the vectorized notation. For $\mathbf{I}_1^{(j)}(t,k)$, we need a more detailed analysis. To start, we define

$$\mathbf{H}^{\perp}(t,t',k) = \frac{1}{d}\left\langle \hat{\mathbf{x}}_i, \hat{\mathbf{x}}_{i'} \right\rangle \sum_{r \in S_i^{\perp}} \mathcal{M}_{t,r}^{(j)} \mathbb{I}\{\left\langle \boldsymbol{\theta}_{t',k,r}^{(j)}, \hat{\mathbf{x}}_i \right\rangle \geq 0; \left\langle \boldsymbol{\theta}_{t',k,r}^{(j)}, \hat{\mathbf{x}}_{i'} \right\rangle \geq 0\}$$

and let $\mathbf{G}^{\perp}(t,t',k) = \bar{\mathbf{A}}\mathbf{H}^{\perp}(t,t',k)\bar{\mathbf{A}}$. Notice that

$$\begin{aligned}
I_{i,1}^{(j)}(t,k) &= \frac{1}{\sqrt{d_1}} \sum_{i'=1}^{n} \sum_{r \in S_{i'}} \bar{\mathbf{A}}_{ii'} \mathcal{M}_{t,r}^{(j)} a_r \left\langle \boldsymbol{\theta}_{t,k+1,r}^{(j)} - \boldsymbol{\theta}_{t,k,r}^{(j)}, \hat{\mathbf{x}}_{i'} \right\rangle \cdot \\
&\qquad \mathbb{I}\left\{ \left\langle \boldsymbol{\theta}_{t,k,r}^{(j)}, \hat{\mathbf{x}}_{i'} \right\rangle \geq 0 \right\} \\
&= -\frac{\eta}{\sqrt{d_1}} \sum_{i'=1}^{n} \sum_{r \in S_{i'}} \bar{\mathbf{A}}_{ii'} \mathcal{M}_{t,r}^{(j)} a_r \left\langle \frac{\partial L\left(\boldsymbol{\Theta}_{t,k}^{(j)}\right)}{\partial \boldsymbol{\theta}_r}, \hat{\mathbf{x}}_{i'} \right\rangle \cdot \\
&\qquad \mathbb{I}\left\{ \left\langle \boldsymbol{\theta}_{t,k,r}^{(j)}, \hat{\mathbf{x}}_{i'} \right\rangle \geq 0 \right\} \\
&= \frac{\eta}{d_1} \sum_{i'=1}^{n} \sum_{i_1=1}^{n} \sum_{i_1'=1}^{n} \sum_{r \in S_i} \bar{\mathbf{A}}_{ii'} \bar{\mathbf{A}}_{i_1 i_1'} \mathcal{M}_{t,r}^{(j)} \left(y_{i_1} - \hat{y}_{i_1}^{(j)}(t,k) \right) \cdot \\
&\qquad \left\langle \hat{\mathbf{x}}_{i_1'}, \hat{\mathbf{x}}_{i'} \right\rangle \mathbb{I}\left\{ \left\langle \boldsymbol{\theta}_{t,k,r}^{(j)}, \hat{\mathbf{x}}_{i'} \right\rangle \geq 0; \left\langle \boldsymbol{\theta}_{t,k,r}^{(j)}, \hat{\mathbf{x}}_{i_1'} \right\rangle \geq 0 \right\} \\
&= \eta \sum_{i'=1}^{n} \sum_{i_1=1}^{n} \sum_{i_1'=1}^{n} \bar{\mathbf{A}}_{ii'} \bar{\mathbf{A}}_{i_1 i_1'} \left(y_{i_1} - \hat{y}_{i_1}^{(j)}(t,k) \right) \cdot \\
&\qquad \left(\mathbf{H}^{(j)}(t,t,k)_{i'i_1'} - \mathbf{H}^{(j)\perp}(t,t,k)_{i'i_1'} \right)
\end{aligned}$$

Thus for $\mathbf{I}_{i,1}^{(j)}(t,k) = [I_{1,1}^{(j)}(t,k), \ldots, I_{n,1}^{(j)}(t,k)]$ we have

$$\begin{aligned}
\mathbf{I}_{i,1}^{(j)}(t,k) &= \eta \bar{\mathbf{A}} \left(\mathbf{H}^{(j)}(t,t,k) - \mathbf{H}^{(j)\perp}(t,t,k) \right) \bar{\mathbf{A}} \left(\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k) \right) \\
&= \eta \left(\mathbf{G}^{(j)}(t,t,k) - \mathbf{G}^{(j)\perp}(t,t,k) \right) \left(\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k) \right)
\end{aligned} \tag{9}$$

which implies that

$$\begin{aligned}
&\left\langle \hat{\mathbf{y}}^{(j)}(t,k) - \mathbf{y}, \mathbf{I}_1^{(j)}(t,k) \right\rangle \\
&\quad = \eta \left\langle \hat{\mathbf{y}}^{(j)}(t,k) - \mathbf{y}, \left(\mathbf{G}^{(j)}(t,t,k) - \mathbf{G}^{(j)\perp}(t,t,k) \right) \left(\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k) \right) \right\rangle \\
&\quad \leq -\eta \left(\lambda_{\min}\left(\mathbf{G}^{(j)}(t,t,k) \right) - \left\| \mathbf{G}^{(j)\perp}(t,t,k) \right\|_2 \right) \left\| \mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k) \right\|_2^2
\end{aligned}$$

Thus, Eq. (8) can be written as

$$\begin{aligned}\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,k+1)\right\|_2^2 \\ \leq \left(1-2\eta\left(\lambda_{\min}\left(\mathbf{G}^{(j)}(t,t,k)\right)-\left\|\mathbf{G}^{(j)\perp}(t,t,k)\right\|_2\right)\right)\cdot \\ \left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2+2\left|\left\langle\hat{\mathbf{y}}^{(j)}(t,k)-\mathbf{y},\mathbf{I}_2^{(j)}(t,k)\right\rangle\right|+ \\ \left\|\hat{\mathbf{y}}^{(j)}(t,k+1)-\hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2\end{aligned} \tag{10}$$

Now, we are ready to prove Theorem 3.

Proof of Theorem 3 We use induction on the following three conditions to prove the theorem.

$$\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,k+1)\right\|_2^2 \leq \left(1-\frac{\eta\lambda_0}{2}\right)\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2 \tag{11}$$

$$\left\|\boldsymbol{\theta}_{t,k,r}^{(j)}-\boldsymbol{\theta}_{0,r}^{(j)}\right\|_2 \leq \frac{2T\eta\zeta}{\delta}\sqrt{\frac{n}{d_1}}\mathbb{E}_{[\mathcal{M}_{t-1}],\mathbf{W}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2\right]^{\frac{1}{2}}+ \eta\zeta n\sqrt{\frac{8(m-1)T}{md_1\delta}} \tag{12}$$

$$\left\|\boldsymbol{\theta}_{t,k,r}^{(j)}-\boldsymbol{\theta}_{0,r}^{(j)}\right\|_2 \leq R \tag{13}$$

Part 1. To start, we show that (13) $\rightarrow$ (11) for all k. As illustrated in Eq. (10), we have

$$\begin{aligned}\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,k+1)\right\|_2^2 \leq \left(1-2\eta\left(\lambda_{\min}\left(\mathbf{G}^{(j)}(t,t,k)\right)-\left\|\mathbf{G}^{(j)\perp}(t,t,k)\right\|_2\right)\right)\cdot \\ \left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2+2\left|\left\langle\hat{\mathbf{y}}^{(j)}(t,k)-\mathbf{y},\mathbf{I}_2^{(j)}(t,k)\right\rangle\right|+ \\ \left\|\hat{\mathbf{y}}^{(j)}(t,k+1)-\hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2\end{aligned}$$

Using Lemmas 3 and 6, we have that

$$\lambda_{\min}\left(\mathbf{G}^{(j)}(t,t,k)\right) \geq \frac{\lambda_0}{2}; \quad \left\|\mathbf{G}^{(j)\perp}\right\|_2 \leq \frac{\lambda_0}{12}$$

which implies that

$$\begin{aligned}\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,k+1)\right\|_2^2 \leq \left(1-\frac{5}{6}\eta\lambda_0\right)\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2+ \\ 2\left|\left\langle\hat{\mathbf{y}}^{(j)}(t,k)-\mathbf{y},\mathbf{I}_2^{(j)}(t,k)\right\rangle\right|+ \\ \left\|\hat{\mathbf{y}}^{(j)}(t,k+1)-\hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2\end{aligned}$$

Choosing $\eta = \frac{\lambda_0}{24n\left\|\tilde{\mathbf{A}}^2\right\|_{1,1}}$, we further use Lemma 5 with $\mathbf{v} = \mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k)$, and Lemma 7 to simplify the second and third term, respective. This gives that

$$\begin{aligned}\left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k+1)\right\|_2^2 &\leq \left(1 - \frac{5}{6}\eta\lambda_0\right)\left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2 + \frac{\eta\lambda_0}{6}\left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2 + \\ &\quad \frac{\eta\lambda_0}{6}\left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2 \\ &= \left(1 - \frac{\eta\lambda_0}{2}\right)\left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2\end{aligned}$$

This shows that (13) → (11).

Part 2. Next, we show that if Eq. (11) hold for all $k = 0, 1, \ldots, k'-1$, then Eq. (12) holds for $k = k'$. Thus, for all $k = 0, 1, \ldots, k'-1$, we have

$$\left\|\mathbf{y} - \hat{\mathbf{y}}(t,k)\right\|_2^2 \leq \left(1 - \frac{\eta\lambda_0}{2}\right)^k \left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2 \tag{14}$$

Lemma 4 gives that

$$\left\|\frac{\partial L(\boldsymbol{\Theta}_{t,k}^{(j)})}{\partial \boldsymbol{\theta}_r}\right\|_2 \leq \sqrt{\frac{n}{d_1}}\left\|\tilde{\mathbf{A}}\right\|_2 \left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k)\right\|_2$$

Using Markov's inequality, we have that with probability at least $1 - \frac{\delta}{2T}$

$$\left\|\mathbf{y} - \hat{\mathbf{y}}(t)\right\|_2^2 \leq \frac{2T}{\delta}\mathbb{E}_{[\mathcal{M}_{t-1}]}\left[\left\|\mathbf{y} - \hat{\mathbf{y}}(t)\right\|_2^2\right]$$

Thus, with probability at least $1 - \frac{\delta}{2T}$, we have

$$\begin{aligned}\left\|\boldsymbol{\theta}_{t,k,r}^{(j)} - \boldsymbol{\theta}_{t,0,r}^{(j)}\right\|_2 &\leq \sum_{k'=0}^{k-1}\left\|\boldsymbol{\theta}_{t,k'+1,r} - \boldsymbol{\theta}_{t,k',r}\right\|_2 \\ &\leq \eta\sum_{k'=0}^{k-1}\left\|\frac{\partial L\left(\boldsymbol{\Theta}_{t,k'}^{(j)}\right)}{\partial \boldsymbol{\theta}_r}\right\|_2 \\ &\leq \eta\sqrt{\frac{n}{d_1}}\sum_{k'=0}^{k-1}\left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k')\right\|_2 \\ &\leq \eta\sqrt{\frac{n}{d_1}}\sum_{k'=0}^{k-1}\left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,0)\right\|_2 \\ &\leq \eta\zeta\sqrt{\frac{n}{d_1}}\left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,0)\right\|_2\end{aligned}$$

$$\leq \eta\zeta\sqrt{\frac{n}{d_1}}\left(\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2+\left\|\hat{\mathbf{y}}(t)-\hat{\mathbf{y}}^{(j)}(t,0)\right\|_2\right)$$
$$\leq \frac{2T\eta\zeta}{\delta}\sqrt{\frac{n}{d_1}}\mathbb{E}_{[\mathcal{M}_{t-1}]}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2\right]^{\frac{1}{2}}+$$
$$4\eta\zeta\kappa\sqrt{\frac{Tdn(m-1)}{d_1m\delta}}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}$$

where in the fourth inequality we use Eq. (14), and in the sixth inequality we use the triangle inequality. This shows that if Eq. (11) hold for all $k = 0, 1, \ldots, k' - 1$, then Eq. (12) holds for $k = k'$.

Part 3. Now, we use induction to show that (13) holds for all k. At $k = 0$, we have that $\boldsymbol{\theta}_{t,k,r} = \boldsymbol{\theta}_{t,r}$. Thus, Eq. (6) implies (13) naturally. Assume (13) holds for all $k = 0, 1, \ldots, k' - 1$. Then we that Eq. (11) holds for all $k = 0, 1, \ldots, k'$. As we have shown in the previous part, Eq. (12) holds for $k = k'$. Since $\alpha < 1 < 2$, we have that

$$\frac{2T\eta\zeta}{\delta}\sqrt{\frac{n}{d_1}} \leq \frac{4T\eta\zeta}{\delta\alpha}\sqrt{\frac{n}{d_1}}$$

Also we have

$$4\eta\zeta\kappa\sqrt{\frac{Tdn(m-1)}{d_1m\delta}}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}} \leq B$$

Therefore, by Eq. (6), and using $\boldsymbol{\theta}_{t,k,r} = \boldsymbol{\theta}_{t,r}$, we have that

$$\left\|\boldsymbol{\theta}_{t,k',r}^{(j)}-\boldsymbol{\theta}_{0,r}^{(j)}\right\|_2 \leq \left\|\boldsymbol{\theta}_{t,r}^{(j)}-\boldsymbol{\theta}_{0,r}^{(j)}\right\|_2+\left\|\boldsymbol{\theta}_{t,k',r}^{(j)}-\boldsymbol{\theta}_{t,0,r}^{(j)}\right\|_2 \leq R$$

Now that Eq. (13) holds for all k, we have Eq. (11) holds and thus Eq. (12) holds for all k.

C.6 Convergence of GIST

We now prove the convergence result for GIST outlined in "Appendix C.4". In showing the convergence of GIST, we care about the regression loss $\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2$ with

$$\hat{\mathbf{y}}(t) = f(\boldsymbol{\Theta}_t, \mathbf{x}) = \frac{1}{m\sqrt{d_1}}\bar{\mathbf{A}}\sigma\left(\bar{\mathbf{A}}\mathbf{x}\boldsymbol{\Theta}_t\right)\mathbf{A} \tag{15}$$

As in previous work (Liao and Kyrillidis 2021), we add the scaling factor $\frac{1}{m}$ to make sure that $\mathbb{E}_{\mathcal{M}_t}[\hat{\mathbf{y}}^{(j)}(t,0)] = \hat{\mathbf{y}}(t)$. Moreover, by properties of the masks $\mathcal{M}_t^{(j)}$, we have

$$f(\boldsymbol{\Theta}, \mathbf{x}) = \sum_{j=1}^{m} f_{\mathcal{M}}^{(j)}(\boldsymbol{\Theta}, \mathbf{x})$$

Thus, we can invoke lemmas 13 and 14 from Liao and Kyrillidis (2021). We state the two key lemmas here in accordance with our own notation.

Lemma 1 *The t-th global step produces squared error satisfying*

$$\begin{aligned}&\left\|\mathbf{y}-\hat{\mathbf{y}}(t+1)\right\|_2^2\\&\quad=\frac{1}{m}\sum_{j=1}^{m}\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,\zeta)\right\|_2^2-\frac{1}{m^2}\sum_{j=1}^{m}\sum_{j'=1}^{j-1}\left\|\hat{\mathbf{y}}^{(j)}(t,\zeta)-\hat{\mathbf{y}}^{(j')}(t,\zeta)\right\|_2^2\end{aligned}$$

Lemma 2 *In the t-th global iteration, the sampled subnetwork's deviation from the whole network is given by*

$$\sum_{j=1}^{m}\left\|\hat{\mathbf{y}}(t+1)-\hat{\mathbf{y}}^{(j)}(t,\zeta)\right\|_2^2=\frac{1}{m}\sum_{j=1}^{m}\sum_{j'=1}^{j-1}\left\|\hat{\mathbf{y}}^{(j)}(t,\zeta)-\hat{\mathbf{y}}^{(j')}(t,\zeta)\right\|_2^2$$

Moreover, Lemma 22 from Liao and Kyrillidis (2021) show that with probability at least $1-2n\exp(-\frac{m}{32})$, for all $R\leq\frac{\kappa}{2}$, it holds that

$$\|\mathbf{\Theta}_0\|_F\leq\kappa\sqrt{2d_1d}-\sqrt{d_1}R$$

For convenience, we assume that such an initialization property holds. Given that for all $r\in[d_1]$, we have $\left\|\boldsymbol{\theta}_{t,r}-\boldsymbol{\theta}_{0,r}\right\|_2\leq R$, we must have that $\|\mathbf{\Theta}_t-\mathbf{\Theta}_0\|_F\leq\kappa\sqrt{d_1}R$. Thus, $\|\mathbf{\Theta}_t\|_F\leq\kappa\sqrt{2d_1d}$ for all t. We point out that, within the proof, we use $R=\frac{\kappa\lambda_0}{192n}$, which satisfies the condition above. Using Lemma 1 to expand the loss at the $(t+1)$th iteration and invoking Theorem 3 gives

$$\begin{aligned}\left\|\mathbf{y}-\hat{\mathbf{y}}(t+1)\right\|_2^2&=\frac{1}{m}\sum_{j=1}^{m}\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,\zeta)\right\|_2^2-\\&\qquad\frac{1}{m^2}\sum_{j=1}^{m}\sum_{j'=1}^{j-1}\left\|\hat{\mathbf{y}}^{(j)}(t,\zeta)-\hat{\mathbf{y}}^{(j')}(t,\zeta)\right\|_2^2\\&\leq\frac{1}{m}\sum_{j=1}^{m}\left(1-\frac{\eta\lambda_0}{2}\right)^{\zeta}\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2-\\&\qquad\frac{1}{m^2}\sum_{j=1}^{m}\sum_{j'=1}^{j-1}\left\|\hat{\mathbf{y}}^{(j)}(t,\zeta)-\hat{\mathbf{y}}^{(j')}(t,\zeta)\right\|_2^2\\&=\frac{1}{m}\sum_{j=1}^{m}\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2-\end{aligned}$$

$$\frac{\eta\lambda_0}{2m}\sum_{k=0}^{\zeta-1}\sum_{j=1}^{m}\left(1-\frac{\eta\lambda_0}{2}\right)^k\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2$$
$$-\frac{1}{m^2}\sum_{j=1}^{m}\sum_{j'=1}^{j-1}\left\|\hat{\mathbf{y}}^{(j)}(t,\zeta)-\hat{\mathbf{y}}^{(j')}(t,\zeta)\right\|_2^2$$

Using the fact that $\mathbb{E}_{\mathcal{M}_t}[\hat{\mathbf{y}}^{(j)}(t,0)]=\hat{\mathbf{y}}(t)$ we have

$$\mathbb{E}_{\mathcal{M}_t}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2\right]=\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2+\mathbb{E}_{\mathcal{M}_t}\left[\left\|\hat{\mathbf{y}}(t)-\hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2\right]$$

Then, using Lemma 2 to rewrite the last term in the equation above and plugging in gives

$$\mathbb{E}_{\mathcal{M}_t}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t+1)\right\|_2^2\right]$$
$$\leq\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2-\frac{\eta\lambda_0}{2m}\sum_{k=0}^{\zeta-1}\sum_{j=1}^{m}\left(1-\frac{\eta\lambda_0}{2}\right)^k\mathbb{E}_{\mathcal{M}_t}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2\right]$$
$$+\frac{1}{m^2}\sum_{j=1}^{m}\sum_{j'=1}^{j-1}\mathbb{E}_{\mathcal{M}_t}\left[\left\|\hat{\mathbf{y}}(t)-\hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2-\left\|\hat{\mathbf{y}}(t+1)-\hat{\mathbf{y}}^{(j)}(t,\zeta)\right\|_2^2\right]$$

We denote the last term within the equation above as ι_t

$$\iota_t=\frac{1}{m^2}\sum_{j=1}^{m}\sum_{j'=1}^{j-1}\mathbb{E}_{\mathcal{M}_t}\left[\left\|\hat{\mathbf{y}}(t)-\hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2-\left\|\hat{\mathbf{y}}(t+1)-\hat{\mathbf{y}}^{(j)}(t,\zeta)\right\|_2^2\right]$$

Lemma 8 shows the bound on ι_t

$$\iota_t\leq\frac{\eta\gamma\lambda_0}{2m}\sum_{j=1}^{m}\sum_{k=0}^{\zeta-1}\mathbb{E}_{\mathcal{M}_t}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t,k)\right\|_2^2\right]+\frac{18\eta\gamma^2 d\lambda^{*2}}{m^2\lambda_0}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}$$

for $\gamma=\left(1-m^{-1}\right)^{\frac{1}{3}}$ Therefore, we can derive the following

$$\mathbb{E}_{\mathcal{M}_t}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t+1)\right\|_2^2\right]$$
$$\leq\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2-\frac{\eta\lambda_0}{2m}\sum_{k=0}^{\zeta-1}\sum_{j=1}^{m}\left(1-\frac{\eta\lambda_0}{2}\right)^k\mathbb{E}_{\mathcal{M}_t}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2\right]$$

$$
\begin{aligned}
&+ \frac{\eta\gamma\lambda_0}{2m}\sum_{j=1}^{n}\sum_{k=0}^{\zeta-1}\mathbb{E}_{\mathcal{M}_t}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2\right] + \frac{18\eta\gamma^2 d\lambda^{*2}}{m^2\lambda_0}\left\|\bar{\mathbf{A}}^2\right\|_{1,1} \\
&\leq \left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2 - \\
&\quad \frac{(1-\gamma)\eta\lambda_0}{2m}\sum_{k=0}^{\zeta-1}\sum_{j=1}^{m}\left(1-\frac{\eta\lambda_0}{2}\right)^k \mathbb{E}_{\mathcal{M}_t}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2\right] \\
&\quad + \frac{18\eta\gamma^2 d\lambda^{*2}}{m^2\lambda_0}\left\|\bar{\mathbf{A}}^2\right\|_{1,1} \\
&\leq \left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2 - \frac{(1-\gamma)\eta\lambda_0}{2}\sum_{k=0}^{\zeta-1}\left(1-\frac{\eta\lambda_0}{2}\right)^k \left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2 \\
&\quad + \frac{18\eta\gamma^2 d\lambda^{*2}}{m^2\lambda_0}\left\|\bar{\mathbf{A}}^2\right\|_{1,1} \\
&= \left(\gamma + (1-\gamma)\left(1-\frac{\eta\lambda_0}{2}\right)^{\zeta}\right)\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2 + \frac{18\eta\gamma^2 d\lambda^{*2}}{m^2\lambda_0}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}
\end{aligned}
$$

Starting from here, we use α to denote the global convergence rate

$$
\alpha = (1-\gamma)\left(1-\left(1-\frac{\eta\lambda_0}{2}\right)^{\zeta}\right)
$$

Since $\zeta \geq 1$, we have that $\alpha \geq \frac{\eta\lambda_0}{2}(1-\gamma)$. Then, the convergence rate above yields the following

$$
\mathbb{E}_{[\mathcal{M}_{t-1}]}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2\right] \leq (1-\alpha)^t\left\|\mathbf{y}-\hat{\mathbf{y}}(0)\right\|_2^2 + \tag{16}
$$

$$
\mathcal{O}\left(\frac{\gamma^2 d\lambda^{*2}}{m^2(1-\gamma)\lambda_0^2}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}\right) \tag{17}
$$

Lastly, we provide a bound on weight perturbation using overparameterization. In particular, we can show that Eq. (6) holds for iteration $t+1$

$$
\left\|\boldsymbol{\theta}_{t+1,r}-\boldsymbol{\theta}_{0,r}\right\|_2 + \frac{4T\eta\zeta}{\delta\alpha}\sqrt{\frac{n}{d_1}}\mathbb{E}_{[\mathcal{M}_t],\boldsymbol{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t+1)\right\|_2^2\right]^{\frac{1}{2}} + (T-t-1)B \leq R
$$

under the assumption that it holds in iteration t

$$
\left\|\boldsymbol{\theta}_{t,r}-\boldsymbol{\theta}_{0,r}\right\|_2 + \frac{4T\eta\zeta}{\delta\alpha}\sqrt{\frac{n}{d_1}}\mathbb{E}_{[\mathcal{M}_{t-1}],\boldsymbol{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2\right]^{\frac{1}{2}} + (T-t)B \leq R
$$

and given the global convergence result

$$\mathbb{E}_{\mathcal{M}_t}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t+1)\right\|_2^2\right] \leq (1-\alpha)\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2 + \frac{18\eta\gamma^2 d\lambda^{*2}}{m^2\lambda_0}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}$$

Thus, it suffices to show that

$$\begin{aligned}
&\left\|\boldsymbol{\theta}_{t+1,r}-\boldsymbol{\theta}_{0,r}\right\|_2 - \left\|\boldsymbol{\theta}_{t,r}-\boldsymbol{\theta}_{0,r}\right\|_2 \\
&\quad\leq \left(\mathbb{E}_{[\mathcal{M}_{t-1}],\boldsymbol{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2\right]^{\frac{1}{2}} - \mathbb{E}_{[\mathcal{M}_t],\boldsymbol{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t+1)\right\|_2^2\right]^{\frac{1}{2}}\right) \\
&\qquad\cdot \frac{4T\eta\zeta}{\delta\alpha}\sqrt{\frac{n}{d_1}} + B
\end{aligned}$$

Using the sub-additivity of the square root function, we derive the following

$$\begin{aligned}
&\mathbb{E}_{[\mathcal{M}_t],\boldsymbol{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t+1)\right\|_2^2\right]^{\frac{1}{2}} \\
&\quad\leq \left((1-\alpha)\,\mathbb{E}_{[\mathcal{M}_{t-1}],\boldsymbol{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2\right] + \frac{18\eta\gamma^2 d\kappa^2\lambda^{*2}}{m^2\lambda_0}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}\right)^{\frac{1}{2}} \\
&\quad\leq \left(1-\frac{\alpha}{2}\right)\mathbb{E}_{[\mathcal{M}_{t-1}],\boldsymbol{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2\right]^{\frac{1}{2}} + \frac{5\gamma\kappa\lambda^*}{m}\sqrt{\frac{\eta d}{\lambda_0}}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}
\end{aligned}$$

It then suffices to show that

$$\begin{aligned}
\left\|\boldsymbol{\theta}_{t+1,r}-\boldsymbol{\theta}_{0,r}\right\|_2 - \left\|\boldsymbol{\theta}_{t,r}-\boldsymbol{\theta}_{0,r}\right\|_2 &\leq \frac{2T\eta\zeta}{\delta}\sqrt{\frac{n}{d_1}}\,\mathbb{E}_{[\mathcal{M}_t],\boldsymbol{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t+1)\right\|_2^2\right]^{\frac{1}{2}} \\
&\quad + B - \frac{20T\zeta\gamma\kappa\lambda^*}{m\delta\alpha}\sqrt{\frac{\eta^3 nd}{\lambda_0 d_1}}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}} \\
&= \frac{2T\eta\zeta}{\delta}\sqrt{\frac{n}{d_1}}\,\mathbb{E}_{[\mathcal{M}_t],\boldsymbol{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t+1)\right\|_2^2\right]^{\frac{1}{2}} \\
&\quad + 4\eta\zeta\kappa\sqrt{\frac{Tdn(m-1)}{d_1 m\delta}}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}
\end{aligned}$$

By Lemma 4, we have

$$\begin{aligned}\left\|\frac{\partial L(\boldsymbol{\Theta}_{t,k}^{(j)})}{\partial \boldsymbol{\theta}_r}\right\|_2 &\leq \sqrt{\frac{n}{d_1}}\left\|\bar{\mathbf{A}}\right\|_2\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,k)\right\|_2\\ &\leq 2\sqrt{\frac{n}{d_1}}\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,0)\right\|_2\\ &\leq 2\sqrt{\frac{n}{d_1}}\left(\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2+\left\|\hat{\mathbf{y}}(t)-\hat{\mathbf{y}}^{(j)}(t,0)\right\|\right)\end{aligned}$$

Lemma 10 gives that

$$\mathbb{E}_{\mathcal{M}_t}\left[\left\|\hat{\mathbf{y}}(t)-\hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2\right]\leq\frac{2d\kappa^2(m-1)}{m^2}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}$$

Apply Markov's inequality gives that with probability at least $1-\frac{\delta}{2T}$, it holds that

$$\begin{aligned}\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2 &\leq \frac{2T}{\delta}\mathbb{E}_{[\mathcal{M}_{t-1}],\boldsymbol{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2\right]^{\frac{1}{2}}\\ \left\|\hat{\mathbf{y}}(t)-\hat{\mathbf{y}}^{(j)}(t,0)\right\|_2 &\leq 2\kappa\sqrt{\frac{Td(m-1)}{m\delta}}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}\end{aligned}$$

Thus, with probability at least $1-\frac{\delta}{2T}$, it holds for all k and j that

$$\begin{aligned}\left\|\frac{\partial L(\boldsymbol{\Theta}_{t,k}^{(j)})}{\partial \boldsymbol{\theta}_r}\right\|_2 \leq \frac{2T}{\delta}\sqrt{\frac{n}{d_1}}\mathbb{E}_{[\mathcal{M}_{t-1}],\boldsymbol{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2\right]^{\frac{1}{2}}+\\ 4\kappa\sqrt{\frac{Tdn(m-1)}{d_1m\delta}}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}\end{aligned}$$

Fix $r\in[d_1]$ and let $\hat{j}$ be the index of the sub-GCN in which r is active. Then with probability at least $1-\frac{\delta}{2T}$, we have that

$$\begin{aligned}\left\|\boldsymbol{\theta}_{t,\zeta,r}^{(\hat{j})}-\boldsymbol{\theta}_{t,r}\right\|_2 &= \left\|\sum_{k=0}^{\zeta-1}\left(\boldsymbol{\theta}_{t,k+1,r}-\boldsymbol{\theta}_{t,k,r}\right)\right\|_2\\ &\leq \sum_{k=0}^{\zeta-1}\left\|\boldsymbol{\theta}_{t,k+1,r}-\boldsymbol{\theta}_{t,k,r}\right\|_2\\ &= \eta\sum_{k=0}^{\zeta-1}\left\|\frac{\partial L(\boldsymbol{\Theta}_{t,k}^{(j)})}{\partial \boldsymbol{\theta}_r}\right\|_2\end{aligned}$$

$$\leq \frac{2T\eta\zeta}{\delta}\sqrt{\frac{n}{d_1}}\mathbb{E}_{[\mathcal{M}_{t-1}],\boldsymbol{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2\right]^{\frac{1}{2}} + 4\eta\zeta\kappa\sqrt{\frac{Tdn(m-1)}{d_1 m\delta}}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}$$

Then, it is easy to see that we indeed have

$$\begin{aligned}\left\|\boldsymbol{\theta}_{t+1,r}-\boldsymbol{\theta}_{0,r}\right\|_2 &\leq \left\|\boldsymbol{\theta}_{t,r}-\boldsymbol{\theta}_{0,r}\right\|_2 + \left\|\boldsymbol{\theta}_{t+1,r}-\boldsymbol{\theta}_{0,r}\right\|_2\\ &= \left\|\boldsymbol{\theta}_{t,r}-\boldsymbol{\theta}_{0,r}\right\|_2 + \left\|\boldsymbol{\theta}_{t,\zeta,r}^{(\hat{j})}-\boldsymbol{\theta}_{t,r}\right\|_2\\ &\leq \left\|\boldsymbol{\theta}_{t,r}-\boldsymbol{\theta}_{0,r}\right\|_2 + \frac{2T\eta\zeta}{\delta}\sqrt{\frac{n}{d_1}}\mathbb{E}_{[\mathcal{M}_{t-1}],\boldsymbol{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t)\right\|_2^2\right]^{\frac{1}{2}}\\ &\quad + 4\eta\zeta\kappa\sqrt{\frac{Tdn(m-1)}{d_1 m\delta}}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}\end{aligned}$$

What remains is to prove Eq. (6) in Theorem 3 for $t = 0$. In that case, we need

$$\begin{aligned}R \geq\ & \frac{4T\eta\zeta}{\delta\alpha}\sqrt{\frac{n}{d_1}}\mathbb{E}_{\boldsymbol{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(0)\right\|_2^2\right]^{\frac{1}{2}} +\\ & 4T\kappa\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}\left(\eta\zeta\sqrt{\frac{Tdn(m-1)}{d_1 m\delta}} + \frac{5T\zeta\gamma\lambda^*}{m\delta\alpha}\sqrt{\frac{\eta^3 nd}{\lambda_0 d_1}}\right)\end{aligned} \tag{18}$$

Using Lemma 11 and using $\alpha \geq \frac{\eta\lambda_0}{2}(1-\gamma)$ to solve for d_1 gives

$$d_1 \geq \Omega\left(\frac{T^2\zeta^2 n}{\lambda_0^4(1-\gamma)^2}\max\left\{\frac{n^3}{\delta^2\kappa^2}, \frac{n^2 d}{\delta^2}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}, T^2\lambda^{*2}d\right\}\right)$$

A detailed computation of the form of d_1 is provided in Sect. C.8.

C.7 Auxiliary lemmas

We now provide all proofs for the major properties and lemmas utilized in deriving the convergence results for `GIST`.

Lemma 3 *Suppose the number of hidden nodes satisfies* $d_1 = \Omega\left(\lambda_0^{-1}n^2\log^{Tmn}/_{\delta}\right)$. *If for all* t, k *it holds that* $\left\|\boldsymbol{\theta}_{t,k,r}-\boldsymbol{\theta}_{0,r}\right\|_2 \leq R := \frac{\kappa\lambda_0}{192n}$, *then with probability at least* $1-\delta$, *for all* $t, t' \in [T]$ *we have:*

$$\lambda_{\min}\left(\mathbf{G}^{(j)}(t,t',k)\right) \geq \frac{\lambda_0}{2}$$

Proof of Lemma 3 Fix some $R > 0$. Following Theorem 2 by Liao and Kyrillidis (2021), we have that with probability at least $1 - 2n^2 e^{-2d_1 t^2}$ it holds that

$$\left\| \mathbf{H}^{(j)}(t,0,0) - \mathbf{H}^{\infty} \right\|_2 \leq nt$$

and with probability at least $1 - n^2 e^{-\frac{d_1 R}{10m}}$ it holds that

$$\left\| \mathbf{H}^{(j)}(t,t',k) - \mathbf{H}^{(j)}(k,0,0) \right\|_2 \leq \frac{3nR}{m}$$

Choosing $t = \frac{\lambda_0}{16n}$ and $R = \frac{\kappa\lambda_0}{192n}$ gives

$$\left\| \mathbf{G}^{(j)}(t,t',k) - \mathbf{G}^{\infty} \right\|_2 \leq \left\| \bar{\mathbf{A}} \right\|_2^2 \left\| \mathbf{H}^{(j)}(t,t',k) - \mathbf{H}^{\infty} \right\|_2 \leq \left\| \bar{\mathbf{A}} \right\|^2 \cdot \frac{\lambda_0}{8} \leq \frac{\lambda_0}{2}$$

with probability at least $1 - n^2 \left(2\exp\left(-\frac{d_1 \lambda_0^2}{128 n^2} \right) + \exp\left(-\frac{d_1 \lambda_0}{480 mn} \right) \right)$. Taking a union bound over all values of t' and j, then plugging in the requirement $d_1 = \Omega\left(\lambda_0^{-1} n^2 \log {}^{Tmn}/_{\delta} \right)$ gives the desired result. □

Lemma 4 *For all $j \in [m], r \in [d_1]$, and t, k, the norm of the sub-GCN gradient is bounded by*

$$\left\| \frac{\partial L\left(\mathbf{\Theta}_{t,k}^{(j)} \right)}{\partial \boldsymbol{\theta}_r} \right\|_2 \leq \sqrt{\frac{n}{d_1}} \left\| \bar{\mathbf{A}} \right\|_2 \left\| \mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k) \right\|_2$$

Proof Recall that

$$\frac{\partial L(\mathbf{\Theta}_{t,k}^{(j)})}{\partial \boldsymbol{\theta}_r} = \frac{1}{\sqrt{d_1}} \sum_{i=1}^{n} \sum_{i'=1}^{n} \left(\hat{y}_i^{(j)}(t,k) - y_i \right) \bar{\mathbf{A}}_{ii'} \mathcal{M}_{t,r}^{(j)} a_r \hat{\mathbf{x}}_{i'} \mathbb{I}\left\{ \left\langle \boldsymbol{\theta}_{t,k,r}^{(j)}, \hat{\mathbf{x}}_{i'} \right\rangle \geq 0 \right\}$$

Since $\left| \mathcal{M}_{t,r}^{(j)} \right| \leq 1$, $|a_r| \leq 1$, $\left\| \hat{\mathbf{x}}_{i'} \right\| \leq 1$, and $\mathbb{I}\left\{ \left\langle \boldsymbol{\theta}_{t,k,r}^{(j)}, \hat{\mathbf{x}}_{i'} \right\rangle \geq 0 \right\} \leq 1$, we must have that

$$\left\| \mathcal{M}_{t,r}^{(j)} a_r \hat{\mathbf{x}}_{i'} \mathbb{I}\left\{ \left\langle \boldsymbol{\theta}_{t,k,r}^{(j)}, \hat{\mathbf{x}}_{i'} \right\rangle \geq 0 \right\} \right\|_2 \leq 1$$

Thus, applying triangle inequality for the double-summation gives

$$\begin{aligned}\left\|\frac{\partial L\left(\boldsymbol{\Theta}_{t,k}^{(j)}\right)}{\partial \boldsymbol{\theta}_r}\right\|_2 &\leq \frac{1}{\sqrt{d_1}}\sum_{i=1}^{n}\sum_{i'=1}^{n}\bar{\mathbf{A}}_{ii'}\left|\hat{\mathbf{y}}_i^{(j)}(t,k)-\mathbf{y}_i\right| \\ &= \frac{1}{\sqrt{d_1}}\left\|\bar{\mathbf{A}}\mathtt{abs}\left(\hat{\mathbf{y}}_i^{(j)}(t,k)-\mathbf{y}_i\right)\right\|_1 \\ &\leq \sqrt{\frac{n}{d_1}}\left\|\bar{\mathbf{A}}\mathtt{abs}\left(\hat{\mathbf{y}}_i^{(j)}(t,k)-\mathbf{y}_i\right)\right\|_2 \\ &\leq \sqrt{\frac{n}{d_1}}\left\|\bar{\mathbf{A}}\right\|_2\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,k)\right\|_2\end{aligned}$$

where in the last inequality we use $\|\mathtt{abs}(\mathbf{v})\|_2 = \|\mathbf{v}\|_2$. □

Lemma 5 *Suppose* $\left\|\boldsymbol{\theta}_{t,k,r}-\boldsymbol{\theta}_{0,r}\right\| \leq R := \frac{\kappa\lambda_0}{192n}$. *Then we have that with probability at least* $1-n\exp\left(d_1\kappa^{-1}R\right)$, *for all* $\mathbf{v}\in\mathbb{R}^n$

$$\left|\left\langle \mathbf{v}, \mathbf{I}_2^{(j)}(t,k)\right\rangle\right| \leq \frac{\eta\lambda_0}{12}\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,k)\right\|_2\|\mathbf{v}\|_2$$

Proof of Lemma 5 Using 1-Lipschitzness of $\sigma(\cdot)$, and that $\left\|\hat{\mathbf{x}}_i\right\|_2 = 1$, we have that for all i, t, k, r

$$\begin{aligned}\left|\sigma\left(\left\langle\boldsymbol{\theta}_{t,k+1,r}^{(j)},\hat{\mathbf{x}}_i\right\rangle\right)-\sigma\left(\left\langle\boldsymbol{\theta}_{t,k,r}^{(j)},\hat{\mathbf{x}}_i\right\rangle\right)\right| &\leq \left|\left\langle\boldsymbol{\theta}_{t,k+1,r}^{(j)}-\boldsymbol{\theta}_{t,k,r}^{(j)},\hat{\mathbf{x}}_i\right\rangle\right| \\ &\leq \left\|\boldsymbol{\theta}_{t,k+1,r}^{(j)}-\boldsymbol{\theta}_{t,k,r}^{(j)}\right\|_2 \\ &= \eta\left\|\frac{\partial L\left(\boldsymbol{\Theta}_{t,k}^{(j)}\right)}{\partial\boldsymbol{\theta}_r}\right\|_2\end{aligned}$$

Lemma 16 from Liao and Kyrillidis (2021) shows that

$$\mathcal{P}\left(\left|S_i^{\perp}\right| \leq 4d_1R\right) \geq \exp(-d_1R)$$

Thus, with probability at least $1-n\exp\left(d_1\kappa^{-1}R\right)$, it holds that

$$\max_{i\in[n]}\left|S_i^{\perp}\right| \leq 4d_1\kappa^{-1}R$$

Thus, for $I_{i,2}^{(j)}(t,k)$, we use Lemma 4 to have

$$\begin{aligned}\left|I_{i,2}^{(j)}(t,k)\right| &\leq \frac{1}{\sqrt{d_1}} \sum_{i'=1}^{n} \sum_{r \in S_{i'}^{\perp}} \bar{\mathbf{A}}_{ii'} \left|\sigma\left(\left\langle \boldsymbol{\theta}_{t,k+1,r}^{(j)}, \hat{\mathbf{x}}_{i'} \right\rangle\right) - \sigma\left(\left\langle \boldsymbol{\theta}_{t,k,r}^{(j)}, \hat{\mathbf{x}}_{i'} \right\rangle\right)\right| \\ &\leq \frac{\eta}{\sqrt{d_1}} \sum_{i'=1}^{n} \sum_{r \in S_{i'}^{\perp}} \bar{\mathbf{A}}_{ii'} \left\| \frac{\partial L\left(\boldsymbol{\Theta}_{t,k}^{(j)}\right)}{\partial \boldsymbol{\theta}_r} \right\|_2 \\ &\leq 4\eta\sqrt{n}\kappa^{-1} R \left\|\bar{\mathbf{A}}\right\|_2 \left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k)\right\|_2 \sum_{i'=1}^{n} \bar{\mathbf{A}}_{ii'}\end{aligned}$$

which yields the following

$$\begin{aligned}\left|\left\langle \mathbf{v}, \mathbf{I}_{i,2}^{(j)}(t,k)\right\rangle\right| &\leq \sum_{i=1}^{n} |v_i| \cdot \left|I_{i,2}^{(j)}(t,k)\right| \\ &\leq 4\eta\sqrt{n}\kappa^{-1} R \left\|\bar{\mathbf{A}}\right\|_2 \left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k)\right\|_2 \sum_{i,i'=1}^{n} \bar{\mathbf{A}}_{ii'} |v_i| \\ &= 4\eta\sqrt{n}\kappa^{-1} R \left\|\bar{\mathbf{A}}\right\|_2 \left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k)\right\|_2 \left\|\bar{\mathbf{A}}\mathtt{abs}(\mathbf{v})\right\|_1 \\ &\leq 4\eta n\kappa^{-1} R \left\|\bar{\mathbf{A}}\right\|_2^2 \left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k)\right\|_2 \|\mathbf{v}\|_2 \\ &\leq 16\eta n\kappa^{-1} R \left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k)\right\|_2 \|\mathbf{v}\|_2\end{aligned}$$

Plugging in $R = \frac{\kappa\lambda_0}{192n}$ gives the desired result. □

Lemma 6 *Under the same condition as Lemma 5, with probability at least* $1 - n\exp\left(-d_1\kappa^{-1}R\right)$*, we have*

$$\left\|\mathbf{G}(t,t',k)^{\perp}\right\|_2 \leq \frac{\lambda_0}{12}$$

Proof of Lemma 6 Lemma 16 from Liao and Kyrillidis (2021) shows that

$$\mathcal{P}\left(\left|S_i^{\perp}\right| \leq 4d_1\kappa^{-1}R\right) \geq \exp(-d_1\kappa^{-1}R)$$

Thus, with probability at least $1 - n\exp\left(-d_1 R\right)$, it holds that

$$\max_{i\in[n]} \left|S_i^{\perp}\right| \leq 4d_1\kappa^{-1}R$$

Recall the definition of $\mathbf{H}^{\perp}$

$$\mathbf{H}^{\perp}(t,t',k) = \frac{1}{d}\left\langle \hat{\mathbf{x}}_i, \hat{\mathbf{x}}_{i'} \right\rangle \sum_{r \in S_i^{\perp}} \mathcal{M}_{t,r}^{(j)} \mathbb{I}\{\left\langle \boldsymbol{\theta}_{t',k,r}^{(j)}, \hat{\mathbf{x}}_i \right\rangle \geq 0; \left\langle \boldsymbol{\theta}_{t',k,r}^{(j)}, \hat{\mathbf{x}}_{i'} \right\rangle \geq 0\}$$

Since $\left|\left\langle \hat{\mathbf{x}}_i, \hat{\mathbf{x}}_{i'} \right\rangle\right| \leq \left\|\hat{\mathbf{x}}_i\right\|_2 \left\|\hat{\mathbf{x}}_{i'}\right\|_2 \leq 1$, we have that

$$\begin{aligned}
\left\|\mathbf{H}^{\perp}(t,t',k)\right\|_2^2 &\leq \left\|\mathbf{H}^{\perp}(t,t',k)\right\|_F^2 \\
&= \sum_{i,i'=1}^{n} \left(\mathbf{H}^{\perp}(t,t',k)_{ij}\right)^2 \\
&\leq \frac{1}{d_1^2} \sum_{i,i'=1}^{n} \sum_{r,r' \in S_i^{\perp}} \mathbb{I}\left\{\left\langle \boldsymbol{\theta}_{t',k,r}^{(j)}, \hat{\mathbf{x}}_i \right\rangle \geq 0; \left\langle \boldsymbol{\theta}_{t',k,r}^{(j)}, \hat{\mathbf{x}}_{i'} \right\rangle \geq 0\right\} \cdot \\
&\qquad \mathbb{I}\left\{\left\langle \boldsymbol{\theta}_{t',k,r'}^{(j)}, \hat{\mathbf{x}}_i \right\rangle \geq 0; \left\langle \boldsymbol{\theta}_{t',k,r'}^{(j)}, \hat{\mathbf{x}}_{i'} \right\rangle \geq 0\right\} \\
&\leq \frac{n^2}{d_1^2} \left(\max_{i \in [n]} \left|S_i^{\perp}\right|\right)^2 = 16 n^2 \kappa^{-2} R^2
\end{aligned}$$

which yields the following

$$\left\|\mathbf{G}^{(j)\perp}(t,t',k)\right\| \leq \left\|\bar{\mathbf{A}}\right\|_2^2 \left\|\mathbf{H}^{(j)\perp}(t,t',k)\right\|_2 = 16 n \kappa^{-1} R$$

Plugging in $R = \frac{\kappa \lambda_0}{192 n}$ gives the desired result.

Lemma 7 *Choosing* $\eta = \frac{\lambda_0}{24 n \left\|\bar{\mathbf{A}}^2\right\|_{1,1}}$, *we have that*

$$\left\|\hat{\mathbf{y}}^{(j)}(t,k+1) - \hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2 \leq \frac{\eta \lambda_0}{6} \left\|\mathbf{y} - \hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2$$

Proof of Lemma 7 Using 1-Lipschitzness of $\sigma(\cdot)$, and that $\left\|\hat{\mathbf{x}}_i\right\|_2 = 1$, we have that for all i, t, k, r

$$\begin{aligned}
\left|\sigma\left(\left\langle \boldsymbol{\theta}_{t,k+1,r}^{(j)}, \hat{\mathbf{x}}_i \right\rangle\right) - \sigma\left(\left\langle \boldsymbol{\theta}_{t,k,r}^{(j)}, \hat{\mathbf{x}}_i \right\rangle\right)\right| &\leq \left|\left\langle \boldsymbol{\theta}_{t,k+1,r}^{(j)} - \boldsymbol{\theta}_{t,k,r}^{(j)}, \hat{\mathbf{x}}_i \right\rangle\right| \\
&\leq \left\|\boldsymbol{\theta}_{t,k+1,r}^{(j)} - \boldsymbol{\theta}_{t,k,r}^{(j)}\right\|_2 \\
&= \eta \left\|\frac{\partial L\left(\boldsymbol{\Theta}_{t,k}^{(j)}\right)}{\partial \boldsymbol{\theta}_r}\right\|_2
\end{aligned}$$

Using the fact that $\left|\mathcal{M}_{t,r}^{(j)}\right| \leq 1$, and $|a_r| \leq 1$, we have

$$\begin{aligned}
&\left(\hat{y}_i^{(j)}(t,k+1) - \hat{y}_i^{(j)}(t,k)\right)^2 \\
&\quad\leq \frac{1}{d_1}\left(\sum_{i'=1}^{n}\sum_{r=1}^{d_1}\bar{\mathbf{A}}_{ii'}\left|\sigma\left(\left\langle\boldsymbol{\theta}_{t,k+1,r}^{(j)},\hat{\mathbf{x}}_i\right\rangle\right)-\sigma\left(\left\langle\boldsymbol{\theta}_{t,k,r}^{(j)},\hat{\mathbf{x}}_i\right\rangle\right)\right|\right)^2 \\
&\quad\leq \frac{\eta^2}{d_1}\left(\sum_{i'=1}^{n}\sum_{r=1}^{d_1}\bar{\mathbf{A}}_{ii'}\left\|\frac{\partial L\left(\boldsymbol{\Theta}_{t,k}^{(j)}\right)}{\partial\boldsymbol{\theta}_r}\right\|_2\right)^2 \\
&\quad\leq \frac{\eta^2}{d_1}\sum_{i'=1}^{n}\sum_{i''=1}^{n}\sum_{r=1}^{d_1}\sum_{r'=1}^{d_1}\bar{\mathbf{A}}_{ii'}\bar{\mathbf{A}}_{ii''}\left\|\frac{\partial L\left(\boldsymbol{\Theta}_{t,k}^{(j)}\right)}{\partial\boldsymbol{\theta}_r}\right\|_2\cdot\left\|\frac{\partial L\left(\boldsymbol{\Theta}_{t,k}^{(j)}\right)}{\partial\boldsymbol{\theta}_{r'}}\right\|_2
\end{aligned}$$

Applying Lemma 4 to bound the norm of the gradient gives

$$\begin{aligned}
&\left(\hat{y}_i^{(j)}(t,k+1) - \hat{y}_i^{(j)}(t,k)\right)^2 \\
&\quad\leq \frac{\eta^2 n}{d_1^2}\left\|\bar{\mathbf{A}}\right\|_2^2\sum_{i'=1}^{n}\sum_{i''=1}^{n}\sum_{r=1}^{d_1}\sum_{r'=1}^{d_1}\bar{\mathbf{A}}_{ii'}\bar{\mathbf{A}}_{ii''}\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2 \\
&\quad\leq \eta^2 n\left\|\bar{\mathbf{A}}\right\|_2^2\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2\sum_{i'=1}^{n}\sum_{i''=1}^{n}\bar{\mathbf{A}}_{ii'}\bar{\mathbf{A}}_{ii''}
\end{aligned}$$

Therefore

$$\begin{aligned}
\left\|\hat{\mathbf{y}}^{(j)}(t,k+1)-\hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2 &= \sum_{i=1}^{n}\left(\hat{y}_i^{(j)}(t,k+1)-\hat{y}_i^{(j)}(t,k)\right)^2 \\
&= \eta^2 n\left\|\bar{\mathbf{A}}\right\|_2^2\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2\sum_{i=1}^{n}\sum_{i'=1}^{n}\sum_{i''=1}^{n}\bar{\mathbf{A}}_{ii'}\bar{\mathbf{A}}_{ii''} \\
&= 4\eta^2 n\left\|\bar{\mathbf{A}}^2\right\|_{1,1}\left\|\mathbf{y}-\hat{\mathbf{y}}^{(j)}(t,k)\right\|_2^2
\end{aligned}$$

Choosing $\eta = \frac{\lambda_0}{24n\left\|\bar{\mathbf{A}}^2\right\|_{1,1}}$ gives the desired result. □

Lemma 8 *As long as* $\left\|\boldsymbol{\theta}_{t,k,r}^{(j)} - \boldsymbol{\theta}_{0,r}\right\|_2 \leq R$ *for all* t, k, j, *and the initialization satisfies* $\|\boldsymbol{\Theta}_0\|_F \leq \kappa\sqrt{2d_1 d} - \sqrt{d_1}R$, *then we have*

$$\iota_t \leq \frac{\eta\gamma\lambda_0}{2m}\sum_{j=1}^{m}\sum_{k=0}^{\zeta-1}\mathbb{E}_{\mathcal{M}_t}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t,k)\right\|_2^2\right] + \frac{18\eta\gamma^2 d\kappa^2\lambda^{*2}}{m^2\lambda_0}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}$$

for $\gamma = \left(1 - m^{-1}\right)^{\frac{1}{3}}$

Proof of Lemma (8) Recall the definition of ι_k

$$\iota_t = \frac{1}{m}\sum_{j=1}^{m} \mathbb{E}_{\mathcal{M}_t}\left[\left\|\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2 - \left\|\hat{\mathbf{y}}(t+1) - \hat{\mathbf{y}}^{(j)}(t,\zeta)\right\|_2^2\right]$$

Notice that

$$\begin{aligned}
&\left\|\hat{\mathbf{y}}(t+1) - \hat{\mathbf{y}}^{(j)}(t,\zeta)\right\|_2^2 \\
&\quad= \left\|\left(\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right) + \left(\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0) - \hat{\mathbf{y}}(t+1) + \hat{\mathbf{y}}^{(j)}(t,\zeta)\right)\right\|_2^2 \\
&\quad= \left\|\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2 + \left\|\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0) - \hat{\mathbf{y}}(t+1) + \hat{\mathbf{y}}^{(j)}(t,\zeta)\right\|_2^2 - \\
&\qquad 2\left\langle\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0) - \hat{\mathbf{y}}(t+1) + \hat{\mathbf{y}}^{(j)}(t,\zeta), \hat{\mathbf{y}}^{(j)}(t,0) - \hat{\mathbf{y}}(t)\right\rangle \\
&\quad\geq \left\|\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2 - 2\left\langle\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}(t+1), \hat{\mathbf{y}}^{(j)}(t,0) - \hat{\mathbf{y}}(t)\right\rangle - \\
&\qquad 2\left\langle\hat{\mathbf{y}}^{(j)}(t,0) - \hat{\mathbf{y}}^{(j)}(t,\zeta), \hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right\rangle
\end{aligned}$$

Thus,

$$\begin{aligned}
&\left\|\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2 - \left\|\hat{\mathbf{y}}(t+1) - \hat{\mathbf{y}}^{(j)}(t,\zeta)\right\|_2^2 \\
&\quad\leq 2\left\langle\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}(t+1), \hat{\mathbf{y}}^{(j)}(t,0) - \mathbf{y}\hat{(t)}\right\rangle + \\
&\qquad 2\left\langle\hat{\mathbf{y}}^{(j)}(t,0) - \hat{\mathbf{y}}^{(j)}(t,\zeta), \hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right\rangle
\end{aligned}$$

Also notice that

$$\begin{aligned}
\sum_{j=1}^{m}\left\langle\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}(t+1), \hat{\mathbf{y}}^{(j)}(t,0) - \mathbf{y}\hat{(t)}\right\rangle &= \left\langle\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}(t+1), \sum_{j=1}^{m}\hat{\mathbf{y}}^{(j)}(t,0) - m\mathbf{y}\hat{(t)}\right\rangle \\
&= 0
\end{aligned}$$

Therefore

$$\begin{aligned}
&\sum_{j=1}^{m}\left(\left\|\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2 - \left\|\hat{\mathbf{y}}(t+1) - \hat{\mathbf{y}}^{(j)}(t,\zeta)\right\|_2^2\right) \\
&\qquad\leq 2\sum_{j=1}^{m}\left\langle\hat{\mathbf{y}}^{(j)}(t,0) - \hat{\mathbf{y}}^{(j)}(t,\zeta), \hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right\rangle
\end{aligned}$$

$$= 2\sum_{j=1}^{m}\sum_{k=0}^{\zeta-1}\left\langle \hat{\mathbf{y}}^{(j)}(t,k) - \hat{\mathbf{y}}^{(j)}(t,k+1), \hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right\rangle$$
$$= 2\sum_{j=1}^{m}\sum_{k=0}^{\zeta-1}\left\langle \mathbf{I}_1^{(j)}(t,k) + \mathbf{I}_2^{(j)}(t,k), \hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right\rangle$$

if we recall the definition of $\mathbf{I}_1^{(j)}(t,k)$ and $\mathbf{I}_2^{(j)}(t,k)$. Thus, we can bound ι_t as

$$\iota_t \leq \frac{2}{m}\sum_{j=1}^{m}\sum_{k=0}^{\zeta-1}\mathbb{E}_{\mathcal{M}_t}\left[\left\langle \mathbf{I}_1^{(j)}(t,k) + \mathbf{I}_2^{(j)}(t,k), \hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right\rangle\right]$$
$$\leq \frac{2}{m}\sum_{j=1}^{m}\sum_{k=0}^{\zeta-1}\mathbb{E}_{\mathcal{M}_t}\left[\left\|\mathbf{I}_1^{(j)}(t,k)\right\| \cdot \left\|\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right\|\right] +$$
$$\mathbb{E}_{\mathcal{M}_t}\left[\left|\left\langle \mathbf{I}_2^{(j)}(t,k), \hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right\rangle\right|\right]$$

Using Lemma (5) with $\mathbf{v} = \hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)$, we have that for all $j \in [m]$

$$\left|\left\langle \mathbf{I}_2^{(j)}(t,k), \hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right\rangle\right| \leq \frac{\eta\lambda_0}{12}\left\|\mathbf{y} - \hat{\mathbf{y}}(t,k)\right\|_2\left\|\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right\|_2$$
$$\leq \eta\lambda^*\left\|\mathbf{y} - \hat{\mathbf{y}}(t,k)\right\|_2\left\|\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right\|_2$$

Furthermore, Lemma 9 gives a bound on $\left\|\mathbf{I}_1^{(j)}(t,k)\right\|_2$

$$\left\|\mathbf{I}_1^{(j)}(t,k)\right\|_2 \leq 2\eta\lambda^*\left\|\mathbf{y} - \hat{\mathbf{y}}(t,k)\right\|_2$$

Therefore,

$$\iota_t \leq \frac{6\eta\lambda^*}{m}\sum_{j=1}^{m}\sum_{k=0}^{\zeta-1}\mathbb{E}_{\mathcal{M}_t}\left[\left\|\mathbf{y} - \hat{\mathbf{y}}(t,k)\right\|_2 \cdot \left\|\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0)\right\|_2\right]$$
$$\leq \frac{3\eta\lambda^*}{m}\sum_{j=1}^{m}\sum_{k=0}^{\zeta-1}\mathbb{E}_{\mathcal{M}_t}\left[\frac{\lambda_0\left(1-m^{-1}\right)^{\frac{1}{3}}}{6\lambda_*}\left\|\mathbf{y} - \hat{\mathbf{y}}(t,k)\right\|_2^2\right]$$

$$
\begin{aligned}
&+\mathbb{E}_{\mathcal{M}_t}\left[\frac{6\lambda_*}{\lambda_0\left(1-m^{-1}\right)^{\frac{1}{3}}}\left\|\hat{\mathbf{y}}(t)-\hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2\right]\\
&=\frac{\eta\lambda_0}{2m}\left(1-m^{-1}\right)^{\frac{1}{3}}\sum_{j=1}^{m}\sum_{k=0}^{\zeta-1}\mathbb{E}_{\mathcal{M}_t}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t,k)\right\|_2^2\right]+\\
&\frac{18\eta\lambda^{*2}}{m\lambda_0\left(1-m^{-1}\right)^{\frac{1}{3}}}\mathbb{E}_{\mathcal{M}_t}\left[\left\|\hat{\mathbf{y}}(t)-\hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2\right] \qquad (19)
\end{aligned}
$$

Substituting Lemma (10) to the last step of Eq. (19) gives the desired result

$$
\begin{aligned}
\iota_t \leq \frac{\eta\lambda_0}{2m}\left(1-m^{-1}\right)^{\frac{1}{3}}\sum_{j=1}^{m}\sum_{k=0}^{\zeta-1}\mathbb{E}_{\mathcal{M}_t}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(t,k)\right\|_2^2\right]+\\
\frac{18\eta\left(1-m^{-1}\right)^{\frac{2}{3}}d\kappa^2\lambda^{*2}}{m^2\lambda_0}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}
\end{aligned}
$$

□

Lemma 9 *Suppose, then we have*

$$
\left\|\mathbf{I}_1^{(j)}(t,k)\right\| \leq 2\eta\lambda^*\left\|\mathbf{y}-\hat{\mathbf{y}}(t,k)\right\|_2
$$

Lemma 10 *Suppose* $\|\boldsymbol{\Theta}_t\|_F \leq 2\kappa\sqrt{d_1 d}$. *Then we have that for all* $j \in [m]$

$$
\mathbb{E}_{\mathcal{M}_t}\left[\left\|\hat{\mathbf{y}}(t)-\hat{\mathbf{y}}^{(j)}(t,0)\right\|_2^2\right] \leq \frac{2d\kappa^2(m-1)}{m^2}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}
$$

Proof Recall that

$$
\begin{aligned}
\hat{y}_i^{(j)}(t,0) &= \frac{1}{\sqrt{d_1}}\sum_{i'=1}^{n}\sum_{r=1}^{d_1}\bar{\mathbf{A}}_{ii'}\mathcal{M}_{t,r}^{(j)}a_r\sigma\left(\left\langle\hat{\mathbf{x}_{i'}},\boldsymbol{\theta}_{t,r}\right\rangle\right)\\
\hat{y}_i(t) &= \frac{1}{m\sqrt{d_1}}\sum_{i'=1}^{n}\sum_{r=1}^{d_1}\bar{\mathbf{A}}_{ii'}\mathcal{M}_{t,r}^{(j)}a_r\sigma\left(\left\langle\hat{\mathbf{x}_{i'}},\boldsymbol{\theta}_{t,r}\right\rangle\right)
\end{aligned}
$$

Thus

$$
\hat{y}_i^{(j)}(t,0)-\hat{y}_i(t) = \frac{1}{\sqrt{d_1}}\sum_{i'=1}^{n}\sum_{r=1}^{d_1}\bar{\mathbf{A}}_{ii'}\left(\mathcal{M}_{t,r}^{(j)}-m^{-1}\right)a_r\sigma\left(\left\langle\hat{\mathbf{x}}_{i'},\boldsymbol{\theta}_{t,r}\right\rangle\right)
$$

Notice that, by independence of $\mathcal{M}_{t,r}^{(j)}$ and $\mathcal{M}_{t,r'}^{(j)}$ for $r \neq r'$, we have

$$\mathbb{E}_{\mathcal{M}_t}\left[\left(\mathcal{M}_{t,r}^{(j)} - m^{-1}\right)\left(\mathcal{M}_{t,r'}^{(j)} - m^{-1}\right)\right] = \begin{cases} \frac{m-1}{m^2} & \text{if } r = r' \\ 0 & \text{if } r \neq r' \end{cases}$$

Therefore

$$\begin{aligned}
\mathbb{E}_{\mathcal{M}_t}\left[\left(\hat{y}_i^{(j)}(t,0) - \hat{y}_i(t)\right)^2\right] &= \frac{m-1}{d_1 m^2} \sum_{i_1', i_2'=1}^{n} \sum_{r=1}^{d_1} \bar{\mathbf{A}}_{ii_1'} \bar{\mathbf{A}}_{ii_2'} \cdot \\
&\qquad \sigma\left(\left\langle \hat{\mathbf{x}}_{i_1'}, \boldsymbol{\theta}_{t,r} \right\rangle\right) \sigma\left(\left\langle \hat{\mathbf{x}}_{i_2'}, \boldsymbol{\theta}_{t,r} \right\rangle\right) \\
&\leq \frac{m-1}{d_1 m^2} \sum_{i_1', i_2'=1}^{n} \sum_{r=1}^{d_1} \bar{\mathbf{A}}_{ii_1'} \bar{\mathbf{A}}_{ii_2'} \left\| \boldsymbol{\theta}_{t,r} \right\|_2^2 \\
&= \frac{m-1}{d_1 m^2} \left\| \boldsymbol{\Theta}_t \right\|_F^2 \sum_{i_1', i_2'=1}^{n} \bar{\mathbf{A}}_{ii_1'} \bar{\mathbf{A}}_{ii_2'} \\
&\leq \frac{2d\kappa^2(m-1)}{m^2} \sum_{i_1', i_2'=1}^{n} \bar{\mathbf{A}}_{ii_1'} \bar{\mathbf{A}}_{ii_2'}
\end{aligned}$$

Thus,

$$\begin{aligned}
\mathbb{E}_{\mathcal{M}_t}\left[\left\| \hat{\mathbf{y}}(t) - \hat{\mathbf{y}}^{(j)}(t,0) \right\|_2^2\right] &= \sum_{i=1}^{n} \mathbb{E}_{\mathcal{M}_t}\left[\left(\hat{y}_i^{(j)}(t,0) - \hat{y}_i(t)\right)^2\right] \\
&\leq \frac{2d\kappa^2(m-1)}{m^2} \sum_{i=1}^{n} \sum_{i_1', i_2'=1}^{n} \bar{\mathbf{A}}_{ii_1'} \bar{\mathbf{A}}_{ii_2'} \\
&= \frac{2d\kappa^2(m-1)}{m^2} \left\| \bar{\mathbf{A}}^2 \right\|_{1,1}
\end{aligned}$$

Lemma 11 *It holds that* □

$$\mathbb{E}\left[\left\| \mathbf{y} - \hat{\mathbf{y}}(0) \right\|_2^2\right] \leq C^2 n + \frac{d}{m^2} \left\| \bar{\mathbf{A}}^2 \right\|_{1,1}$$

where C is defined in Assumption 4.

Proof of Lemma 11 Note that

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(0)\right\|_2^2\right] &= \sum_{i=1}^{n}\mathbb{E}_{\boldsymbol{\Theta}_0,\mathbf{A}}\left[(y_i-\hat{y}_i(0))^2\right]\\ &= y_i^2 - 2y_i\mathbb{E}_{\boldsymbol{\Theta}_0,\mathbf{A}}\left[\hat{y}_i(0)\right]+\mathbb{E}_{\boldsymbol{\Theta}_0,\mathbf{A}}\left[\hat{y}_i(0)^2\right]\\ &\leq C^2+\mathbb{E}_{\boldsymbol{\Theta}_0,\mathbf{A}}\left[\hat{y}_i(0)^2\right]\end{aligned}$$

where the last inequality follows from the bound on $|y_i|$ and the fact that $\mathbb{E}_{\boldsymbol{\Theta}_0,\mathbf{A}}\left[\hat{y}_i(0)\right]=0$. Moreover, we have

$$\begin{aligned}&\mathbb{E}_{\boldsymbol{\Theta}_0,\mathbf{A}}\left[\hat{y}_i(0)^2\right]\\ &\quad=\frac{1}{m^2d_1}\sum_{i_1=1}^{n}\sum_{i_2=1}^{n}\sum_{r_1=1}^{d_1}\sum_{r_2=1}^{d_1}\bar{\mathbf{A}}_{ii_1}\bar{\mathbf{A}}_{ii_2}\mathbb{E}_{\mathbf{A}}[a_{r_1}a_{r_2}]\cdot\\ &\qquad\mathbb{E}_{\boldsymbol{\Theta}_0}\left[\sigma\left(\left\langle\boldsymbol{\theta}_{0,r_1},\hat{\mathbf{x}}_{i_1}\right\rangle\right)\sigma\left(\left\langle\boldsymbol{\theta}_{0,r_2},\hat{\mathbf{x}}_{i_2}\right\rangle\right)\right]\\ &\quad=\frac{1}{m^2d_1}\sum_{i_1=1}^{n}\sum_{i_2=1}^{n}\sum_{r=1}^{d_1}\bar{\mathbf{A}}_{ii_1}\bar{\mathbf{A}}_{ii_2}\mathbb{E}_{\boldsymbol{\Theta}_0}\left[\sigma\left(\left\langle\boldsymbol{\theta}_{0,r},\hat{\mathbf{x}}_{i_1}\right\rangle\right)\sigma\left(\left\langle\boldsymbol{\theta}_{0,r},\hat{\mathbf{x}}_{i_2}\right\rangle\right)\right]\\ &\quad\leq\frac{1}{m^2d_1}\sum_{i_1=1}^{n}\sum_{i_2=1}^{n}\sum_{r=1}^{d_1}\bar{\mathbf{A}}_{ii_1}\bar{\mathbf{A}}_{ii_2}\mathbb{E}_{\boldsymbol{\Theta}_0}\left[\left\|\boldsymbol{\theta}_{0,r}\right\|_2^2\right]\\ &\quad=\frac{d}{m^2}\sum_{i_1=1}^{n}\sum_{i_2=1}^{n}\bar{\mathbf{A}}_{ii_1}\bar{\mathbf{A}}_{ii_2}\end{aligned}$$

Thus

$$\mathbb{E}_{\boldsymbol{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(0)\right\|_2^2\right]\leq\sum_{i=1}^{n}\mathbb{E}_{\boldsymbol{\Theta}_0,\mathbf{A}}\left[\hat{y}_i(0)^2\right]\leq C^2n+\frac{d}{m^2}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}$$

□

C.8 Computation of overparameterization

In this section, we provide the computation of the exact overparameterization requirement such that Theorem 2 holds. Recall from the end of Sect. C.6, Eq. (18) that we need to satisfy the following requirement

$$\begin{aligned}R\geq\frac{4T\eta\zeta}{\delta\alpha}\sqrt{\frac{n}{d_1}}\mathbb{E}_{\boldsymbol{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(0)\right\|_2^2\right]^{\frac{1}{2}}+\\ 4T\kappa\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}\left(\eta\zeta\sqrt{\frac{Tdn(m-1)}{d_1m\delta}}+\frac{5T\zeta\gamma\lambda^*}{m\delta\alpha}\sqrt{\frac{\eta^3nd}{\lambda_0d_1}}\right)\end{aligned}$$

To start, Lemma 11 gives that

$$\mathbb{E}_{\mathbf{\Theta}_0,\mathbf{A}}\left[\left\|\mathbf{y}-\hat{\mathbf{y}}(0)\right\|_2^2\right]^{\frac{1}{2}} \leq \left(C^2 n + \frac{d}{m^2}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}\right) \leq C\sqrt{n} + \frac{\sqrt{d}}{m}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}$$

Plugging this into Eq. (18) and using $O(\cdot)$ to hide constants, we have

$$\mathcal{O}\left(\frac{\kappa\lambda_0}{n}\right) \geq \frac{T\eta\zeta}{\delta\alpha}\sqrt{\frac{n}{d_1}}\left(\sqrt{n} + \frac{\sqrt{d}}{m}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}\right) + \tag{20}$$

$$T\kappa\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}\left(\eta\zeta\sqrt{\frac{Tdn(m-1)}{d_1 m\delta}} + \frac{T\zeta\gamma\lambda^*}{m\delta\alpha}\sqrt{\frac{\eta^3 nd}{\lambda_0 d_1}}\right) \tag{21}$$

Opening the parenthesis and plugging in $\alpha \geq \frac{\eta\lambda_0}{2}(1-\gamma)$ gives

$$\mathcal{O}\left(\frac{\kappa\lambda_0}{n}\right) \geq \frac{T\zeta n}{\delta\lambda_0(1-\gamma)\sqrt{d_1}} + \frac{T\zeta}{\delta\lambda_0(1-\gamma)m}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}\sqrt{\frac{nd}{d_1}} + T\kappa\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}\eta\zeta\sqrt{\frac{Tdn(m-1)}{d_1 m\delta}} + T^2\kappa\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}\frac{\zeta\gamma\lambda^*}{m(1-\gamma)}\sqrt{\frac{\eta d}{\lambda_0^3 d_1}}$$

Further plugging in $\eta = \frac{\lambda_0}{24n\left\|\bar{\mathbf{A}}^2\right\|_{1,1}}$ gives

$$\mathcal{O}\left(\frac{\kappa\lambda_0}{n}\right) \geq \frac{T\zeta n}{\delta\lambda_0(1-\gamma)\sqrt{d_1}} + \frac{T\zeta}{\delta\lambda_0(1-\gamma)m}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}\sqrt{\frac{nd}{d_1}} + T\lambda_0\kappa\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{-\frac{1}{2}}\zeta\sqrt{\frac{Td(m-1)}{d_1 nm\delta}} + T^2\kappa\frac{\zeta\gamma\lambda^*}{\lambda_0 m(1-\gamma)}\sqrt{\frac{d}{nd_1}}$$

Equation (20) is satisfied if each term on the left-hand side is bounded by the right-hand side when hiding constants. Thus, it holds as long as the following holds

$$\frac{T\zeta n}{\delta\lambda_0(1-\gamma)\sqrt{d_1}} \leq \mathcal{O}\left(\frac{\kappa\lambda_0}{n}\right) \tag{22}$$

$$\frac{T\zeta}{\delta\lambda_0(1-\gamma)m}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}\sqrt{\frac{nd}{d_1}} \leq \mathcal{O}\left(\frac{\kappa\lambda_0}{n}\right) \tag{23}$$

$$T\lambda_0\kappa\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{-\frac{1}{2}}\zeta\sqrt{\frac{Td(m-1)}{d_1 nm\delta}} \leq \mathcal{O}\left(\frac{\kappa\lambda_0}{n}\right) \tag{24}$$

$$T^2\kappa\frac{\zeta\gamma\lambda^*}{\lambda_0 m(1-\gamma)}\sqrt{\frac{d}{nd_1}} \leq \mathcal{O}\left(\frac{\kappa\lambda_0}{n}\right) \tag{25}$$

where in requirement of Eq. (22) we use the simplification $\alpha \geq \frac{\eta\lambda_0}{2}(1-\gamma)$. Notice that d_1 appears in the denominator on the left-hand side for Eqs. (22)–(25). Thus, as long as d_1 is large enough, the above requirements can be satisfied. Next, we solve each of these requirements individually. We move $\sqrt{d_1}$ to the right-hand side of Eqs. (22)–(25) and $\frac{\kappa\lambda_0}{n}$ to the left-hand side to get that

$$\sqrt{d_1} \geq \Omega\left(\frac{T\zeta n^2}{\delta\lambda_0^2(1-\gamma)\kappa}\right)$$
$$\sqrt{d_1} \geq \Omega\left(\frac{T\zeta\sqrt{n^3 d}}{\delta\lambda_0^2(1-\gamma)m\kappa}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{\frac{1}{2}}\right)$$
$$\sqrt{d_1} \geq \Omega\left(T\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{-\frac{1}{2}}\zeta\sqrt{\frac{Td(m-1)n}{m\delta}}\right)$$
$$\sqrt{d_1} \geq \Omega\left(\frac{T^2\zeta\gamma\lambda^*}{\lambda_0^2 m(1-\gamma)}\sqrt{nd}\right)$$

Squaring both sides gives

$$d_1 \geq \Omega\left(\frac{T^2\zeta^2 n^4}{\delta^2\lambda_0^4(1-\gamma)^2\kappa^2}\right) \tag{26}$$
$$d_1 \geq \Omega\left(\frac{T^2\zeta n^3 d}{\delta^2\lambda_0^4(1-\gamma)^2 m^2\kappa^2}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}\right) \tag{27}$$
$$d_1 \geq \Omega\left(\frac{T^3\zeta^2 nd(m-1)}{m\delta}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}^{-1}\right) \tag{28}$$
$$d_1 \geq \Omega\left(\frac{T^4\zeta^2\gamma^2\lambda^{*2}nd}{\lambda_0^4 m^2(1-\gamma)^2}\right) \tag{29}$$

First, notice that

$$\left\|\bar{\mathbf{A}}^2\right\|_{1,1} \geq n\left\|\bar{\mathbf{A}}^2\right\|_F \geq n$$

Substituting this and $\frac{m-1}{m} = \gamma^3$ into Eq. 28 to get

$$d_1 \geq \Omega\left(\frac{T^2\zeta^2 n^4}{\delta^2\lambda_0^4(1-\gamma)^2\kappa^2}\right) \tag{30}$$
$$d_1 \geq \Omega\left(\frac{T^2\zeta n^3 d}{\delta^2\lambda_0^4(1-\gamma)^2 m^2\kappa^2}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}\right) \tag{31}$$
$$d_1 \geq \Omega\left(\frac{T^3\zeta^2\gamma^3 d}{\delta}\right) \tag{32}$$

$$d_1 \geq \Omega\left(\frac{T^4\zeta^2\gamma^2\lambda^{*2}nd}{\lambda_0^4 m^2(1-\gamma)^2}\right) \tag{33}$$

Notice that $\gamma \leq 0$ and $\lambda_0 \leq 0$, and treat δ as some constants. Thus Eq. (33) implies Eq. (32). Noticing that $m \geq 1$, $\zeta \geq 1$ and $\gamma \leq 1$. Factoring out $\frac{T^2\zeta^2 n}{\lambda_0^4(1-\gamma)^2}$ for Eq. (30), (31), and (33), the requirements boils down to

$$d_1 \geq \Omega\left(\frac{T^2\zeta^2 n}{\lambda_0^4(1-\gamma)^2} \cdot \frac{n^3}{\delta^2\kappa^2}\right) \tag{34}$$

$$d_1 \geq \Omega\left(\frac{T^2\zeta^2 n}{\lambda_0^4(1-\gamma)^2} \cdot \frac{n^2 d}{\delta^2}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}\right) \tag{35}$$

$$d_1 \geq \Omega\left(\frac{T^2\zeta^2 n}{\lambda_0^4(1-\gamma)^2} \cdot T^2\lambda^{*2}d\right) \tag{36}$$

Taking a maximum over Eqs. (34)–(36) gives the final requirement

$$d_1 \geq \Omega\left(\frac{T^2\zeta^2 n}{\lambda_0^4(1-\gamma)^2} \max\left\{\frac{n^3}{\delta^2\kappa^2}, \frac{n^2 d}{\delta^2}\left\|\bar{\mathbf{A}}^2\right\|_{1,1}, T^2\lambda^{*2}d\right\}\right)$$

References

Agarwal, A., Duchi, J.C.: Distributed delayed stochastic optimization. In: Proceedings of Advances in Neural Information Processing Systems (NeurIPS) (2011)

Balaban, A.T.: Applications of graph theory in chemistry. J. Chem. Inf. Comput. Sci. (1985)

Benkö, G., Flamm, C., Stadler, P.F.: A graph-based toy model of chemistry. J. Chem. Inf. Comput. Sci. (2003)

Ben-Nun, T., Hoefler, T.: Demystifying Parallel and Distributed Deep Learning: An In-Depth Concurrency Analysis. ACM Computing Surveys (CSUR) (2019)

Bergen, L., O'Donnell, T., Bahdanau, D.: Systematic generalization with edge transformers. Adv. Neural. Inf. Process. Syst. **34**, 1390–1402 (2021)

Bronstein, M.M., Bruna, J., LeCun, Y., Szlam, A., Vandergheynst, P.: Geometric deep learning: going beyond Euclidean data. IEEE Signal Process Magazine (2017)

Brown, T.B., et al.: Language models are few-shot learners. arXiv preprint arXiv:2005.14165 (2020)

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. Adv. Neural. Inf. Process. Syst. **33**, 1877–1901 (2020)

Chen, J., Ma, T., Xiao, C.: FastGCN: fast learning with graph convolutional networks via importance sampling. In: Proceedings of the International Conference on Learning Representations (ICLR) (2018)

Chen, J., Zhu, J., Song, L.: Stochastic training of graph convolutional networks with variance reduction. In: Proceedings of the International Conference on Machine Learning (ICML) (2018)

Chiang, W.-L., Liu, X., Si, S., Li, Y., Bengio, S., Hsieh, C.-J.: Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In: Proceedings of International Conference on Knowledge Discovery & Data Mining (KDD) (2019)

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., Stoyanov, V.: Unsupervised Cross-lingual Representation Learning at Scale. arXiv preprint arXiv:1911.02116 (2019)

Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. arXiv preprint arXiv:1606.09375 (2016)
Du, S.S., Zhai, X., Poczos, B., Singh, A.: Gradient Descent Provably Optimizes Over-parameterized Neural Networks (2019)
Dun, C., Wolfe, C.R., Jermaine, C.M., Kyrillidis, A.: Resist: Layer-wise decomposition of resents for distributed training. In: Uncertainty in Artificial Intelligence, pp. 610–620 (2022). PMLR
Gao, H., Wang, Z., Ji, S.: Large-scale learnable graph convolutional networks. arXiv preprint arXiv:1808.03965 (2018)
Gholami, A., Azad, A., Jin, P., Keutzer, K., Buluc, A.: Integrated Model, Batch and Domain Parallelism in Training Neural Networks. arXiv preprint arXiv:1712.04432 (2017)
Gong, L., Cheng, Q.: Exploiting edge features for graph neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9211–9219 (2019)
Gori, M., Monfardini, G., Scarselli, F.: A new model for learning in graph domains. In: Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN) (2005)
Günther, S., Ruthotto, L., Schroder, J.B., Cyr, E.C., Gauger, N.R.: Layer-Parallel Training of Deep Residual Neural Networks. arXiv preprint arXiv:1812.04352 (2018)
Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Proceedings of Advances in Neural Information Processing Systems (NeurIPS) (2017)
Hao, K.: Training a single AI model can emit as much carbon as five cars in their lifetimes (2019)
Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D.d.L., Hendricks, L.A., Welbl, J., Clark, A., et al.: Training compute-optimal large language models. arXiv preprint arXiv:2203.15556 (2022)
Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., Leskovec, J.: Open Graph Benchmark: Datasets for Machine Learning on Graphs. arXiv e-prints, 2005–00687 (2020) arXiv:2005.00687 [cs.LG]
Huang, W., Zhang, T., Rong, Y., Huang, J.: Adaptive sampling towards fast graph representation learning. arXiv preprint arXiv:1809.05343 (2018)
Jacot, A., Gabriel, F., Hongler, C.: Neural tangent kernel: convergence and generalization in neural networks. arXiv preprint arXiv:1806.07572 (2018)
Jiang, X., Zhu, R., Li, S., Ji, P.: Co-embedding of nodes and edges with graph neural networks. IEEE Trans. Pattern Anal. Mach. Intell. (2020)
Karypis, G., Kumar, V.: A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM J. Sci. Comput. (1998)
Karypis, G., Kumar, V.: Multilevelk-way partitioning scheme for irregular graphs. J. Parallel Distribut. Comput. (1998)
Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. arXiv preprint arXiv:1412.6980 (2014)
Kipf, T.N., Welling, M.: Semi-Supervised Classification with Graph Convolutional Networks. arXiv preprint arXiv:1609.02907 (2016)
Kirby, A.C., Samsi, S., Jones, M., Reuther, A., Kepner, J., Gadepally, V.: Layer-Parallel Training with GPU Concurrency of Deep Residual Neural Networks via Nonlinear Multigrid. arXiv preprint arXiv:2007.07336 (2020)
Li, Q., Han, Z., Wu, X.-M.: Deeper insights into graph convolutional networks for semi-supervised learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
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: Proceedings of Advances in Neural Information Processing Systems (NeurIPS) (2017)
Liao, F., Kyrillidis, A.: On the Convergence of Shallow Neural Network Training with Randomly Masked Neurons (2021)
Lin, T., Stich, S.U., Kshitij Patel, K., Jaggi, M.: Don't Use Large Mini-Batches, Use Local SGD. arXiv preprint arXiv:1808.07217 (2018)
Lusher, D., Koskinen, J., Robins, G.: Exponential Random Graph Models for Social Networks: Theory, Methods, and Applications. Cambridge University Press (2013)
Masci, J., Boscaini, D., Bronstein, M., Vandergheynst, P.: Geodesic convolutional neural networks on riemannian manifolds. In: Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCVW) (2015)
Nakkiran, P., Kaplun, G., Bansal, Y., Yang, T., Barak, B., Sutskever, I.: Deep Double Descent: Where Bigger Models and More Data Hurt. arXiv preprint arXiv:1912.02292 (2019)

Newman, M.E., Watts, D.J., Strogatz, S.H.: Random graph models of social networks. Proc. Natl. Acad. Sci. (2002)
Oymak, S., Soltanolkotabi, M.: Toward moderate overparameterization: Global convergence guarantees for training shallow neural networks. IEEE J. Select. Areas Inf. Theory **1**(1), 84–105 (2020)
Paszke, A., et al.: Pytorch: An imperative style, high-performance deep learning library. In: Proceedings of Advances in Neural Information Processing Systems (NeurIPS) (2019)
Pauloski, J.G., Zhang, Z., Huang, L., Xu, W., Foster, I.T.: Convolutional Neural Network Training with Distributed K-FAC. arXiv preprint arXiv:2007.00784 (2020)
Peng, T., Sarazen, M.: The Staggering Cost of Training SOTA AI Models (2019)
Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning, pp. 8748–8763 (2021). PMLR
Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., Eliassi-Rad, T.: Collective classification in network data. AI Mag. **29**, 93 (2008)
Sharir, O., Peleg, B., Shoham, Y.: The cost of training nlp models: a concise overview. arXiv preprint arXiv:2004.08900 (2020)
Shi, S., Tang, Z., Chu, X., Liu, C., Wang, W., Li, B.: A Quantitative Survey of Communication Optimizations in Distributed Deep Learning. arXiv preprint arXiv:2005.13247 (2020)
Song, Z., Yang, X.: Quadratic Suffices for Over-parametrization via Matrix Chernoff Bound (2020)
Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. (2014)
Stich, S.U.: Local SGD converges fast and communicates little. In: Proceedings of the International Conference on Learning Representations (ICLR) (2019)
Tavarageri, S., Sridharan, S., Kaul, B.: Automatic Model Parallelism for Deep Neural Networks with Compiler and Hardware Support. arXiv preprint arXiv:1906.08168 (2019)
Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
You, Y., Chen, T., Wang, Z., Shen, Y.: L2-gcn: Layer-wise and learned efficient training of graph convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2127–2135 (2020)
Yu, K., Flynn, T., Yoo, S., D'Imperio, N.: Layered sgd: A decentralized and synchronous sgd algorithm for scalable deep neural network training. arXiv preprint arXiv:1906.05936 (2019)
Yuan, B., Kyrillidis, A., Jermaine, C.M.: Distributed Learning of Deep Neural Networks using Independent Subnet Training. arXiv preprint arXiv:1810.01392 (2019)
Zeng, H., Zhou, H., Srivastava, A., Kannan, R., Prasanna, V.: GraphSAINT: Graph Sampling Based Inductive Learning Method. arXiv preprint arXiv:1907.04931 (2019)
Zhang, S., Choromanska, A.E., LeCun, Y.: Deep learning with elastic averaging sgd. In: Proceedings of Advances in Neural Information Processing Systems (NeurIPS) (2015)
Zhang, Z., Yin, L., Peng, Y., Li, D.: A quick survey on large scale distributed deep learning systems. In: 2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS) (2018)
Zhu, W., Zhao, C., Li, W., Roth, H., Xu, Z., Xu, D.: LAMP: Large Deep Nets with Automated Model Parallelism for Image Segmentation. arXiv preprint arXiv:2006.12575 (2020)
Zinkevich, M., Weimer, M., Li, L., Smola, A.J.: Parallelized stochastic gradient descent. In: Proceedings of Advances in Neural Information Processing Systems (NeurIPS), pp. 2595–2603 (2010)
Zou, D., Hu, Z., Wang, Y., Jiang, S., Sun, Y., Gu, Q.: Layer-Dependent Importance Sampling for Training Deep and Large Graph Convolutional Networks. arXiv preprint arXiv:1911.07323 (2019)