Neighborhood Gradient Clustering: An Efficient Decentralized Learning Method for Non-IID Data

Anonymous authors
Paper under double-blind review

Abstract

Decentralized learning algorithms enable the training of deep learning models over large distributed datasets, without the need for a central server. The current state-of-the-art decentralized algorithms mostly assume the data distributions to be Independent and Identically Distributed (IID). In practical scenarios, the distributed datasets can have significantly different data distributions across the agents. This paper focuses on improving decentralized learning on non-IID data with minimal compute and memory overheads. We propose *Neighborhood Gradient Clustering (NGC)*, a novel decentralized learning algorithm that modifies the local gradients of each agent using self- and cross-gradient information. In particular, the proposed method averages the local gradients with model-variant or data-variant cross-gradients based on the communication budget. Model-variant cross-gradients are derivatives of the received neighbors' model parameters with respect to the local dataset. Data-variant cross-gradient derivatives of the local model with respect to its neighbors' datasets. The data-variant cross-gradients are aggregated through an additional communication round without breaking the privacy constraints of the decentralized setting. We theoretically analyze the convergence characteristics of *NGC* and demonstrate its efficiency on non-IID data sampled from various vision and language datasets. Our experiments demonstrate that the proposed method either remains competitive or outperforms (by $0-6\%$) the existing state-of-the-art (SoTA) decentralized learning algorithm on non-IID data with significantly less compute and memory requirements. Further, we show that the model-variant cross-gradient information available locally at each agent can improve the performance on non-IID data by $1-35\%$ without additional communication cost.

1 Introduction

The remarkable success of deep learning is mainly attributed to the availability of humongous amounts of data and computing power. Large amounts of data are generated on a daily basis at different devices all over the world which could be used to train powerful deep learning models. Collecting such data for centralized processing is not practical because of communication and privacy constraints. To address this concern, a new interest in developing distributed learning algorithms Agarwal & Duchi (2011) has emerged. Federated learning (centralized learning) Konecny et al. (2016) is a popular setting in the distributed machine learning paradigm, where the training data is kept locally at the edge devices and a global shared model is learned by aggregating the locally computed updates through a coordinating central server. Such a setup requires continuous communication with a central server which becomes a potential bottleneck Haghighat et al. (2020). This has motivated the advancements in decentralized machine learning.

Decentralized machine learning is a branch of distributed learning that focuses on learning from data distributed across multiple agents/devices. Unlike Federated learning, these algorithms assume that the agents are connected peer to peer without a central server. It has been demonstrated that decentralized learning algorithms Lian et al. (2017) can perform comparably to centralized algorithms on benchmark vision datasets. Lian et al. (2017) present Decentralised Parallel Stochastic Gradient Descent (D-PSGD) by combining SGD with gossip averaging algorithm Xiao & Boyd (2004). Further, the authors analytically show that the convergence rate of D-PSGD is similar to its centralized counterpart Dean et al. (2012). Decentralized

Momentum Stochastic Gradient Descent (DMSGD) which introduces momentum to D-PSGD was proposed and analyzed in Balu et al. (2021). Assran et al. (2019) introduce Stochastic Gradient Push (SGP) which extends D-PSGD to directed and time-varying graphs. Tang et al. (2019); Koloskova et al. (2019) explore error-compensated compression techniques (Deep-Squeeze and CHOCO-SGD) to reduce the communication cost of D-PSGD significantly while achieving the same convergence rate as centralized algorithms. Aketi et al. (2021) combined Deep-Squeeze with SGP to propose communication-efficient decentralized learning over time-varying and directed graphs. Recently, Koloskova et al. (2020) proposed a unified framework for the analysis of gossip-based decentralized SGD methods and provide the best-known convergence guarantees.

The key assumption to achieve state-of-the-art performance by all the above-mentioned decentralized algorithms is that the data is independent and identically distributed (IID) across the agents. In particular, the data is assumed to be distributed in a uniform and random manner across the agents. This assumption does not hold in most of the real-world applications as the data distributions across the agents are significantly different (non-IID) based on the user pool Hsieh et al. (2020). The effect of non-IID data in a peer-to-peer decentralized setup is a relatively under-studied problem. There are only a few works that try to bridge the performance gap between IID and non-IID data for a decentralized setup. Note that, we mainly focus on a common type of non-IID data, widely used in prior works Tang et al. (2018); Lin et al. (2021); Esfandiari et al. (2021): a skewed distribution of data labels across agents. Tang et al. (2018) proposed D^2 algorithm that extends D-PSGD to non-IID data. However, the algorithm was demonstrated on only a basic LeNet5 LeCun et al. (1998) model and is not scalable to deeper models with normalization layers. SwarmSGD proposed by Nadiradze et al. (2019) leverages random interactions between participating agents in a graph to achieve consensus. Lin et al. (2021) replace local momentum with Quasi-Global Momentum (QGM) and improve the test performance by $1 - 20\%$. But the improvement in accuracy is only $1 - 2\%$ in case of highly skewed data distribution as shown in Aketi et al. (2022). Most recently, Esfandiari et al. (2021) proposed Cross-Gradient Aggregation (CGA) and a compressed version of CGA (CompCGA), claiming state-of-the-art performance for decentralized learning algorithms over completely non-IID data. CGA aggregates *cross-gradient* information, i.e., derivatives of its model with respect to its neighbors' datasets through an additional communication round. It then updates the model using projected gradients based on quadratic programming. CGA and CompCGA require a very slow quadratic programming step Goldfarb & Idnani (1983) after every iteration for gradient projection which is both compute and memory intensive. This work focuses on the following question: *Can we improve the performance of decentralized learning on non-IID data with minimal compute and memory overhead?*

In this paper, we propose *Neighborhood Gradient Clustering* (*NGC*) algorithm with two variants to handle non-IID data in peer-to-peer decentralized learning setups. Note that we do not use any kind of unsupervised clustering and the word 'clustering' is used as a synonym for 'grouping'. Firstly, we classify the gradients at each agent into three types, namely self-gradients, model-variant cross-gradients, and data-variant cross-gradients (see Section 3). The self-gradients (or local gradients) are the derivatives computed at each agent on its model parameters with respect to the local dataset. The model-variant cross-gradients are the derivatives of the received neighbors' model parameters with respect to the local dataset. These gradients are computed locally at each agent after receiving the neighbors' model parameters. Communicating the neighbors' model parameters is a necessary step in any gossip-based decentralized algorithm Lian et al. (2017). The data-variant cross-gradients are the derivatives of the local model with respect to its neighbors' datasets. These gradients are obtained through an additional round of communication. We then cluster the gradients into a) *model-variant cluster* with self-gradients and model-variant cross-gradients, and b) *data-variant cluster* with self-gradients and data-variant cross-gradients. Finally, the local gradients are replaced with the model-variant cluster means in NGC_{mv} where the communication budget is $1\times$, and with the data-variant cluster means in NGC_{dv} where the communication budget is $2\times$. The main motivation behind this modification is to account for the high variation in the computed local gradients (and in turn the model parameters) across the neighbors due to the non-IID nature of the data distribution.

The proposed NGC_{dv} has two rounds of communication at every iteration to send model parameters and data-variant cross-gradients. This incurs $2\times$ communication cost compared to traditional decentralized algorithms (D-PSGD). To reduce this communication overhead, we propose the compressed version of NGC_{dv} (*CompNGC*) by compressing the additional round of cross-gradient communication. Note that NGC_{mv} has

no communication overhead. We provide a detailed convergence analysis of the proposed NGC_{dv} algorithm and show that the analysis can be trivially extended to NGC_{mv}. We validate the performance of the proposed algorithm on various datasets, model architectures, and graph topologies. We compare the proposed algorithm with D-PSGD, CGA, and CompCGA and show that we can achieve superior performance on non-IID data compared to the current state-of-the-art approach. We also report the order of communication, memory, and compute overheads required for NGC and CGA as compared to D-PSGD.

Contributions: In summary, we make the following contributions.

- We propose the Neighborhood Gradient Clustering (NGC) algorithm with two variations for decentralized learning on non-IID data. NGC_{mv} utilizes self-gradients and model-variant cross-gradients to improve the performance on non-IID data. NGC_{dv} improves this performance further by utilizing self-gradients and data-variant cross-gradients at the cost of $2\times$ communication overhead.
- We theoretically show that the convergence rate of NGC is $\mathcal{O}(\frac{1}{\sqrt{NK}})$, which is consistent with the state-of-the-art decentralized learning algorithms.
- NGC_{mv} outperforms the D-PSGD baseline by $1-35\%$ without any communication overhead.
- NGC_{dv} either remains competitive or outperforms by $0-6\%$ with significantly less compute and memory requirements compared to the current state-of-the-art at iso-communication.

2 Background

In this section, we provide the background on decentralized learning algorithms with peer-to-peer connections.

The main goal of decentralized machine learning is to learn a global model using the knowledge extracted from the locally generated and stored data samples across N edge devices/agents while maintaining privacy constraints. In particular, we solve the optimization problem of minimizing global loss function $\mathcal{F}(x)$ distributed across N agents as given in equation. 1.

$$\begin{aligned} \min_{x\in\mathbb{R}^d} \mathcal{F}(x) &= \frac{1}{N}\sum_{i=1}^{N} f_i(x), \\ \textit{and} \;\; f_i(x) &= \mathbb{E}_{d^i\in D^i}[F_i(x;d^i)] \;\; \forall i \end{aligned} \tag{1}$$

This is typically achieved by combining stochastic gradient descent Bottou (2010) with global consensus-based gossip averaging Xiao & Boyd (2004). The communication topology in this setup is modeled as a graph $G = ([N], E)$ with edges $\{i, j\} \in E$ if and only if agents i and j are connected by a communication link exchanging the messages directly. We represent $\mathcal{N}_i$ as the neighbors of i including itself. It is assumed that the graph G is strongly connected with self-loops i.e., there is a path from every agent to every other agent. The adjacency matrix of the graph G is referred to as a mixing matrix W where w_{ij} is the weight associated with the edge $\{i, j\}$. Note that, weight 0 indicates the absence of a direct edge between the agents. We assume that the mixing matrix is doubly-stochastic and symmetric, similar to all previous works in decentralized learning. For example, in a undirected ring topology, $w_{ij} = \frac{1}{3}$ if $j \in \{i-1, i, i+1\}$. Further, the initial models and all the hyperparameters are synchronized at the beginning of the training. Algorithm. 2 in the appendix describes the flow of D-PSGD with momentum. The convergence of the Algorithm. 2 assumes the data distribution across the agents to be Independent and Identically Distributed (IID).

3 Neighborhood Gradient Clustering

We propose the *Neighborhood Gradient Clustering (NGC)* algorithm and a compressed version of NGC which improve the performance of decentralized learning on non-IID data. We define the concepts of self-gradients and cross-gradients below.

Self-Gradient: For an agent i with the local dataset D^i and model parameters x^i, the self-gradient is the gradient of the loss function f_i with respect to the model parameters x^i, evaluated on mini-batch d^i sampled from dataset D^i.

$$g^{ii} = \nabla_x F_i(x^i; d^i) \tag{2}$$

Cross-Gradient: For an agent i with model parameters x^i connected to neighbor j that has local dataset D^j, the cross-gradient is the gradient of the loss function f_j with respect to the model parameters x^i, evaluated on mini-batch d^j sampled from dataset D^j.

$$g^{ij} = \nabla_x F_j(x^i; d^j) \tag{3}$$

At each agent i, we further divide the cross-gradients into two categories. 1) *Model-variant cross-gradients*: The derivatives that are computed locally using its local data on the neighbors' model parameters (g^{ji}). 2) *Data-variant cross-gradients*: The derivatives (received through communication) of its model parameters on the neighbors' dataset (g^{ij}). Note that an agent i computes the cross-gradients g^{ji} that act as model-variant cross-gradients for i and as data-variant cross-gradients for j. In particular, the data-variant cross-gradient of i i.e., g^{ij} is computed by agent j using its local data d_j after receiving the model parameters x^i from i and is then communicated to agent i if needed.

3.1 The *NGC* algorithm

The flow of the Neighborhood Gradient Clustering (*NGC*) is shown in Algorithm. 1. The overview of the algorithm is illustrated in Figure. 5 of Appendix A.3.

Algorithm 1 Neighborhood Gradient Clustering (*NGC*)

Input: Each agent $i \in [1, N]$ initializes model weights $x^i_{(0)}$, step size η, momentum coefficient β, averaging rate γ, mixing matrix $W = [w_{ij}]_{i,j\in[1,N]}$, *NGC* mixing weight $\alpha \in \{0, 1\}$, and I_{ij} are elements of $N \times N$ identity matrix, $\mathcal{N}_i$ represents neighbors of i including itself.

Each agent simultaneously implements the Train( ) procedure
1. **procedure** Train( )
2. **for** k=$0, 1, \ldots, K-1$ **do**
3. $d^i_k \sim D^i$
4. $g^{ii}_k = \nabla_x f_i(d^i_k; x^i_k)$
5. SendReceive(x^i_k)
6. **for** each neighbor $j \in \{\mathcal{N}_i - i\}$ **do**
7. $g^{ji}_k = \nabla_x f_i(d^i_k; x^j_k)$
8. **if** $\alpha \neq 0$: SendReceive(g^{ji}_k)
11. **end**
12. $\widetilde{g}^i_k = \sum_{j\in\mathcal{N}_i}[(1-\alpha)w_{ji}g^{ji}_k + \alpha w_{ij}g^{ij}_k]$
13. $v^i_k = \beta v^i_{(k-1)} - \eta\widetilde{g}^i_k$
14. $\widetilde{x}^i_k = x^i_k + v^i_k$
15. $x^i_{(k+1)} = \widetilde{x}^i_k + \gamma\sum_{j\in\mathcal{N}_i}(w_{ij} - I_{ij})x^j_k$
16. **end**
17. **return**

The main contribution of the proposed *NGC* algorithm is the local gradient manipulation step (line 12 in Algorithm. 1). We first introduce a hyper-parameter α called *NGC mixing weight* which is either set to 0 or 1. In the kth iteration of *NGC*, each agent i calculates its self-gradient g^{ii}. Then, agent i's model parameters are transmitted to all other agents (j) in its neighborhood, and the respective cross-gradients are calculated by the neighbors. These cross-gradients are transmitted back to agent i through an additional communication round if α is set to 1. At every iteration after the first communication round, each agent

i has access to self-gradients (g^{ii}) and model-variant cross-gradients. When α is set to 1, the agents get access to data-variant cross-gradients through a second round of communication. Now, we group these cross-gradients into two clusters: a) *Model-variant cluster* $\{g^{ji} \forall j \in \mathcal{N}_i\}$ that includes self-gradients and model-variant cross-gradients, and b) *Data-variant cluster* $\{g^{ij} \forall j \in \mathcal{N}_i\}$ that includes self-gradients and data-variant cross-gradients. The local gradients at each agent are replaced with either of the clusters' mean based on α value as shown in Equation. 4, which assumes a uniform mixing matrix ($w_{ij} = 1/m; m = |\mathcal{N}_i|$). When α is set to zero, the proposed algorithm modifies the local gradient with model-variant cluster mean and this is referred to as NGC_{mv}. This version of the algorithm does not incur any communication overhead The $\alpha = 1$ version of the algorithm replaces local gradient with data-variant cluster mean and is referred to as NGC_{dv}. Setting α to any non-zero value requires an additional round of communication and we experimentally determine that $\alpha = 1$ gives the best results (refer section. 6 for more details).

$$\widetilde{g}_k^i = (1-\alpha) * \underbrace{\Big[\frac{1}{m}\sum_{j\in\mathcal{N}_i} g_k^{ji}\Big]}_{\textbf{(a) Model-variant cluster mean}} \quad + \quad \alpha * \underbrace{\Big[\frac{1}{m}\sum_{j\in\mathcal{N}_i} g_k^{ij}\Big]}_{\textbf{(b) Data-variant cluster mean}} \tag{4}$$

The motivation for this modification is to reduce the variation of the computed local gradients across the agents. In IID settings, the local gradients should statistically resemble the cross-gradients and hence simple gossip averaging is sufficient to reach convergence. However, in the non-IID case, the local gradients across the agents are significantly different due to the variation in datasets and hence the model parameters on which the gradients are computed. The proposed algorithm reduces this variation in the local gradients as it is equivalent to adding the bias term ϵ for NGC_{mv} and the bias term ω for NGC_{dv} as shown in Equation. 5.

$$\begin{aligned}
\widetilde{g}_k^i &= g_k^{ii} + (1-\alpha) * \underbrace{\Big[\frac{1}{m}\sum_{j\in\mathcal{N}_i} (g_k^{ji} - g_k^{ii})\Big]}_{\text{model variance bias } \epsilon_k^i} + \alpha * \underbrace{\Big[\frac{1}{m}\sum_{j\in\mathcal{N}_i} \frac{1}{m}(g_k^{ij} - g_k^{ii})\Big]}_{\text{data variance bias } \omega_k^i} \\
\epsilon_k^i &= \frac{1}{m} * \sum_{j\in\mathcal{N}_i} \big(\nabla_x F(d_k^i; x_k^j) - \nabla_x F(d_k^i; x_k^i)\big) \\
\omega_k^i &= \frac{1}{m} * \sum_{j\in\mathcal{N}_i} \big(\nabla_x F(d_k^j; x_k^i) - \nabla_x F(d_k^i; x_k^i)\big)
\end{aligned} \tag{5}$$

The bias term ϵ compensates for the difference in a neighborhood's self-gradients caused due to variation in the model parameters across the neighbors. Whereas, the bias term ω compensates for the difference in a neighborhood's self-gradients caused due to variation in the data distribution across the neighbors. We hypothesize and show through our experiments that the addition of one of these bias terms to the local gradients improves the performance of decentralized learning on non-IID data by accelerating global convergence.

3.2 The Compressed *NGC* Algorithm

At every iteration, the NGC_{dv} algorithm involves two rounds of communication with the neighbors: 1) communicate the model parameters, and 2) communicate the cross-gradients. This communication overhead can be a bottleneck in a resource-constrained environment. Hence we propose a compressed version of *NGC* using Error Feedback SGD (EF-SGD) Karimireddy et al. (2019) to compress gradients. We compress the error-compensated self-gradients and cross-gradients from 32 bits (floating point precision of arithmetic operations) to 1 bit by using scaled signed gradients. The error between the compressed and non-compressed gradient of the current iteration is added as feedback to the gradients in the next iteration before compression. The pseudo-code for *CompNGC* is shown in Algorithm. 3 in the Appendix.

4 Convergence Analysis of *NGC*

In this section, we provide a convergence analysis for NGC_{dv}. We assume that the following statements hold:

Assumption 1 - Lipschitz Gradients: Each function $f_i(x)$ is L-smooth.

Assumption 2 - Bounded Variance: The variance of the stochastic gradients is assumed to be bounded.

$$\mathbb{E}_{d\sim D_i}||\nabla F_i(x;d) - \nabla f_i(x)||^2 \leq \sigma^2 \;\; \forall i \in [1, N] \tag{6}$$

$$||\nabla f_i(x) - \nabla \mathcal{F}(x)||^2 \leq \zeta^2 \;\; \forall i \in [1, N] \tag{7}$$

Assumption 3 - Doubly Stochastic Mixing Matrix: The mixing matrix W is a real doubly stochastic matrix with $\lambda_1(W) = 1$ and

$$max\{|\lambda_2(W)|, |\lambda_N(W)|\} \leq \sqrt{\rho} < 1 \tag{8}$$

where $\lambda_i(W)$ is the ith largest eigenvalue of W and ρ is a constant.
The above assumptions are commonly used in most decentralized learning setups.

Lemma 1. *Given assumptions 1-3, we define $g^i = \nabla F_i(x;d)$ and the NGC gradient update $\widetilde{g}^i$. For all i, we have:*

$$\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}(\tilde{\mathbf{g}}^i - \mathbf{g}^i)\right\|^2\right] \leq 4(\frac{\sigma^2}{N} + \zeta^2) \tag{9}$$

A complete proof for Lemma 1 can be found in Appendix. A.1. This lemma bounds the difference between self-gradients and the proposed gradients in terms of inter- and intra-gradient variance bounds. The inter-gradient variation bounded by ζ^2 determines the heterogeneity in the data distribution i.e., the non-IIDness. The intra-gradient variation bounded by σ^2 determines the stochastic noise across the mini-batches within an agent. Intuitively, the distance between the self-gradients g^i and the NGC_{dv} gradient update $\widetilde{g}^i$ increases with an increase in the degree of heterogeneity in data distribution which is expected. Note that the convergence rate for NGC_{mv} remains the same as CGA where $\mathbb{E}\left[\left\|(\tilde{\mathbf{g}}^i - \mathbf{g}^i)\right\|^2\right]$ is bounded by a positive constant which is of the order of $\frac{1}{K}$.

Theorem 1 presents the convergence of the proposed NGC_{dv} algorithm and the proof is detailed in Appendix A.2.2

Theorem 1. *(Convergence of NGC_{dv} algorithm) Given assumptions 1-3 and lemma 1, let step size η satisfy the following conditions:*

$$1 - \frac{6\eta^2 L^2}{(1-\beta)^2(1-\sqrt{\rho})} - \frac{4L\eta}{(1-\beta)^2} \geq 0 \tag{10}$$

For all $K \geq 1$, we have

$$\begin{aligned}\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[||\nabla\mathcal{F}(\bar{x}_k)||^2] \leq & \frac{1}{C_1 K}(\mathcal{F}(\bar{x}^0) - \mathcal{F}^*) + \left(10C_2 + 10C_3\frac{\eta^2\beta^2}{(1-\beta)^4} + 4C_4\right)(\frac{\sigma^2}{N} + \zeta^2) + \\ & C_5\frac{14\eta^2(\sigma^2+\zeta^2)}{(1-\beta)^2(1-\sqrt{\rho})^2}\end{aligned} \tag{11}$$

where β = momentum coefficient, $\bar{x}_k = \frac{1}{N}\sum_{i=1}^{N} x_k^i$
$C_1 = \frac{1}{2}(\frac{\eta}{1-\beta} - \frac{(1-\beta)}{\beta L})$,
$C_2 = \left(\frac{\beta L\eta^2}{2(1-\beta)^3} + \frac{\eta^2 L}{(1-\beta)^2}\right)/C_1$, $C_3 = \frac{(1-\beta)L}{2\beta}/C_1$,
$C_4 = \frac{\eta^2\beta L}{2(1-\beta)^3}/C_1$, *and* $C_5 = \frac{\eta L^2}{2(1-\beta)}/C_1$.

The result of the theorem. 1 shows that the magnitude of the average gradient achieved by the consensus model is upper-bounded by the difference between the initial objective function value and the optimal value, the sampling variance, and gradient variations across the agents representing data heterogeneity. A detailed explanation of the constraints on step size is presented in the Appendix. A.2.3. Further, we present a corollary to show the convergence rate of NGC_{dv} in terms of the number of iterations.

Corollary 1. *Suppose that the step size satisfies* $\eta = \mathcal{O}\left(\sqrt{\frac{N}{K}}\right)$ *and that* $\zeta^2 = \mathcal{O}\left(\frac{1}{\sqrt{K}}\right)$. *For a sufficiently large* $K \geq \frac{36NL^2}{r^2}$, *and*
$r = \sqrt{4(1-\sqrt{\rho})^2 + 6(1-\sqrt{\rho})(1-\beta)^2} - 2(1-\sqrt{\rho})$, *we have, for some constant* $C > 0$,

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\left[\|\nabla\mathcal{F}\left(\bar{\mathbf{x}}_k\right)\|^2\right] \leq C\left(\frac{1}{\sqrt{NK}} + \frac{1}{K} + \frac{1}{K^{1.5}}\right).$$

The proof for Corollary. 1 can be found in Appendix. A.2.4. It indicates that the proposed algorithm achieves linear speedup (with a convergence rate of $\mathcal{O}(\frac{1}{\sqrt{NK}})$) when K is sufficiently large. This convergence rate is similar to the well-known best result for decentralized SGD algorithms in the literature as shown in Table. 1.

Table 1: Convergence rate comparison between decentralized learning algorithms.

Method	Rate
D-PSGD	$\mathcal{O}(\frac{1}{\sqrt{NK}} + \frac{1}{K})$
SwarmSGD	$\mathcal{O}(\frac{1}{\sqrt{K}})$
CGA	$\mathcal{O}(\frac{1}{\sqrt{NK}} + \frac{1}{K} + \frac{1}{K^{1.5}} + \frac{1}{K^2})$
NGC_{dv} (ours)	$\mathcal{O}(\frac{1}{\sqrt{NK}} + \frac{1}{K} + \frac{1}{K^{1.5}})$

5 Experiments

In this section, we analyze the performance of the proposed NGC_{mv}, NGC_{dv} and *CompNGC* techniques and compare them with the corresponding baseline i.e., D-PSGD algorithm Lian et al. (2017) and state-of-the-art CGA and CompCGA methods Esfandiari et al. (2021). The source code has been uploaded in the supplementary material.

5.1 Experimental Setup

The efficiency of the proposed method is demonstrated through our experiments on a diverse set of datasets, model architectures, tasks, topologies, and numbers of agents. We present the analysis on – (a) Datasets (Appendix A.5): vision datasets (CIFAR-10, CIFAR-100, Fashion MNIST and Imagenette Husain (2018)) and language datasets (AGNews Zhang et al. (2015)). (b) Model architectures (Appendix A.6): 5-layer CNN, VGG-11, ResNet-20, LeNet-5, MobileNet-V2, ResNet-18, $\text{BERT}_{\text{mini}}$ and $\text{DistilBERT}_{\text{base}}$ (c) Tasks: Image and Text Classification. (d) Topologies: Ring, Chain, and Torus. (e) Number of agents: varying from 4 to 20. Note that we use low-resolution (32×32) images of the Imagenette dataset for the experiments in Table. 3. The results for high resolution (224×224) Imagenette are presented in Table. 4.

We consider an extreme case of non-IID distribution where no two neighboring agents have the same class. This is referred to as complete label-wise skew or 100% label-wise non-IIDness Hsieh et al. (2020). In particular, for a 10-class dataset such as CIFAR-10 - each agent in a 5-agent system has data from 2 distinct classes, and each agent in a 10-agent system has data from a unique class. For a 20-agent system, two agents that are maximally apart share the samples belonging to a class. We report the test accuracy of the consensus model averaged over three randomly chosen seeds. The details of the hyperparameters for all the experiments are present in Appendix. A.10. We compare the proposed method with iso-communication baselines. The experiments on NGC_{mv} are compared with D-PSGD, NGC_{dv} with CGA, and *CompNGC* with CompCGA. The communication cost for each experiment in this section is presented in Appendix A.9.

5.2 Results

Firstly, we evaluate variants of *NGC* and *CompNGC* and compare them with respective baselines in Table. 2, for training different models trained on CIFAR-10 over various graph sizes and topologies. We observe that

Table 2: Average test accuracy comparisons for CIFAR-10 with non-IID data using various architectures and graph topologies. The results are averaged over three seeds where std is indicated.

Method	Agents	5layer CNN Ring	5layer CNN Torus	VGG-11 Ring	ResNet-20 Ring
D-PSGD	5	76.00 ± 1.44	-	67.04 ± 5.36	82.13 ± 0.84
	10	47.68 ± 3.20	55.34 ± 6.32	44.14 ± 3.30	31.66 ± 6.01
	20	44.85 ± 1.94	50.12 ± 1.91	38.92 ± 2.99	31.94 ± 2.91
NGC_{mv} (ours)	5	**82.20 ± 0.34**	-	**79.27 ± 0.39**	**85.88 ± 0.58**
	10	**67.43 ± 1.15**	**73.84 ± 0.33**	**59.92 ± 2.12**	**66.02 ± 2.86**
	20	**58.80 ± 1.30**	**64.55 ± 1.16**	**52.70± 1.63**	**50.74 ± 2.36**
CGA	5	82.20 ± 0.43	-	84.41 ± 0.22	87.52 ± 0.50
	10	72.96 ± 0.40	76.04 ± 0.62	**79.66 ± 0.46**	79.98 ± 1.23
	20	69.88 ± 0.84	73.21 ± 0.27	79.30 ± 0.12	75.13 ± 1.56
NGC_{dv} (ours)	5	**83.36 ± 0.65**	-	**85.15 ± 0.58**	**88.52 ± 0.19**
	10	**75.34 ± 0.30**	**78.53 ± 0.56**	79.55 ± 0.30	**84.02 ± 0.44**
	20	**73.36 ± 0.88**	**75.11 ± 0.07**	**79.43 ± 0.62**	**81.26 ± 0.69**
CompCGA	5	82.00 ± 0.25	-	83.65 ± 0.41	86.73 ± 0.34
	10	71.41 ± 0.94	75.95 ± 0.41	73.96 ± 0.31	73.63 ± 0.55
	20	68.15 ± 0.79	71.71 ± 0.54	73.72 ± 2.74	66.34 ± 0.98
CompNGC (ours)	5	**82.91 ± 0.21**	-	**84.03 ± 0.32**	**87.56 ± 0.34**
	10	**74.36 ± 0.42**	**77.82 ± 0.20**	**77.02 ± 0.14**	**78.50 ± 0.98**
	20	**71.46 ± 0.85**	**73.62 ± 0.74**	**73.76 ± 0.20**	**72.62 ± 0.71**

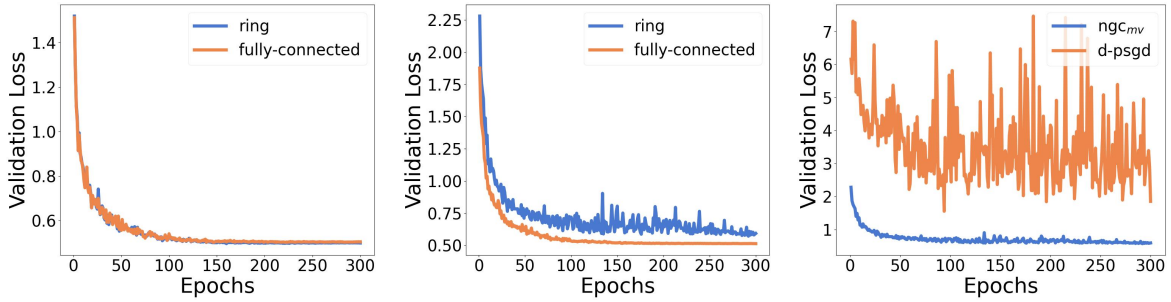
Table 3: Average test accuracy comparisons for various datasets with non-IID sampling trained over undirected ring topology. The results are averaged over three seeds where std is indicated.

Method	Agents	Fashion MNIST (LeNet-5)	CIFAR-100 (ResNet-20)	Imagenette(32x32) (MobileNet-V2)
D-PSGD	5	86.43 ± 0.14	44.66 ± 5.23	47.09 ± 9.20
	10	75.49 ± 0.32	19.03 ± 13.27	32.81 ± 2.18
NGC_{mv} (ours)	5	**88.49 ± 0.18**	**55.96 ± 0.95**	**60.15 ± 2.17**
	10	**82.85 ± 0.24**	**35.34 ± 0.32**	**36.13 ± 1.97**
CGA	5	90.03 ± 0.39	56.43 ± 2.39	72.82 ± 1.25
	10	**87.61 ± 0.30**	53.61 ± 1.07	61.97 ± 0.58
NGC_{dv} (ours)	5	**90.61 ± 0.18**	**56.50 ± 3.23**	**74.49 ± 0.93**
	10	87.24 ± 0.23	**53.77 ± 0.15**	**64.06 ± 1.11**
CompCGA	5	90.45 ± 0.34	55.74 ± 0.33	72.76 ± 0.44
	10	81.62 ± 0.37	38.84 ± 0.54	59.92 ± 0.72
CompNGC (ours)	5	**90.48 ± 0.19**	**57.51 ± 0.48**	**72.91 ± 1.06**
	10	**83.38 ± 0.39**	**43.07 ± 0.32**	**61.91 ± 2.10**

NGC_{mv} consistently outperforms D-PSGD for all models, graph sizes, and topologies with a significant performance gain varying from $3 - 35\%$. Our experiments show the superiority of NGC_{dv} and *CompNGC* over CGA and CompCGA respectively. The performance gains are more pronounced when considering larger graphs (with 20 agents) and compact models such as ResNet-20. We further demonstrate the generalizability of the proposed method by evaluating it on various image datasets such as Fashion MNIST, and Imagenette and on challenging datasets such as CIFAR-100. Table. 3, 4 show that NGC_{mv} outperforms D-PSGD by $2 - 13\%$ across various datasets whereas NGC_{dv} and *CompNGC* remain competitive with an average improvement of $\sim 1\%$.

Table 4: Average test accuracy comparisons for AGNews dataset (left side of the table) and full resolution (224×224) Imagenette dataset (right side of the table). The results are averaged over three seeds where std is indicated.

Method	AGNews-BERT$_{\text{mini}}$		AGNews-DistilBERT$_{\text{base}}$	Imagenette(224x224)-ResNet-18	
	Agents = 4	Agents = 8	Agents = 4	Agents = 5	
	Ring Topology		Ring Topology	Ring Topology	Chain Topology
D-PSGD	89.21 ± 0.41	85.48 ± 0.71	91.54 ± 0.07	65.43 ± 4.60	42.02 ± 1.25
NGC_{mv} (ours)	**89.40 ± 0.13**	**87.58 ± 0.07**	**91.70 ± 0.11**	**73.15 ± 0.38**	**47.87 ± 0.99**
CGA	91.43 ± 0.11	**89.15 ± 0.45**	93.42 ± 0.04	85.00 ± 0.67	65.96 ± 1.84
NGC_{dv} (ours)	**92.24 ± 0.29**	89.02 ± 0.39	**94.11 ± 0.01**	**85.85 ± 0.60**	**67.77 ± 1.76**
CompCGA	91.05 ± 0.29	88.91 ± 0.25	**93.54 ± 0.03**	84.65 ± 0.57	**62.93 ± 1.33**
CompNGC (ours)	**91.24 ± 0.43**	**89.01 ± 0.13**	93.50 ± 0.16	**85.44 ± 0.10**	62.64 ± 0.85



(a) NGC_{mv} on IID data (b) NGC_{mv} on Non-IID data (c) Non-IID, ring topology

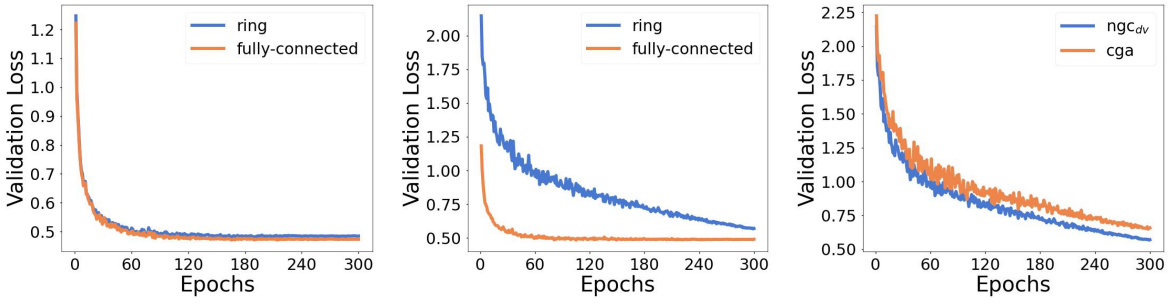
Figure 1: Average Validation loss of NGC_{mv} during training CIFAR-10 on a 5-layer CNN with 5 agents.

To show the effectiveness of the proposed method across different modalities, we present results on the text classification task in Table 4. We train on the BERT$_{\text{mini}}$ model with the AGNews dataset distributed over 4 and 8 agents and DistilBert$_{\text{base}}$ distributed over 4 agents. For NGC_{mv}, we see a maximum improvement of 2.1% over the baseline D-PSGD algorithm. Even for the text classification task, we observe NGC_{dv} and *CompNGC* to be competitive with CGA and CompCGA methods. These observations are consistent with the results of the image classification tasks. Finally, through this exhaustive set of experiments, we demonstrate that the NGC_{dv} can serve as a simple plugin to boost the performance of decentralized learning over label-wise non-IID data without significant memory and compute overheads. Further, NGC_{mv} demonstrates that locally available model-variant cross-gradient information at each agent can be efficiently utilized to improve decentralized learning with no communication overhead.

5.3 Analysis

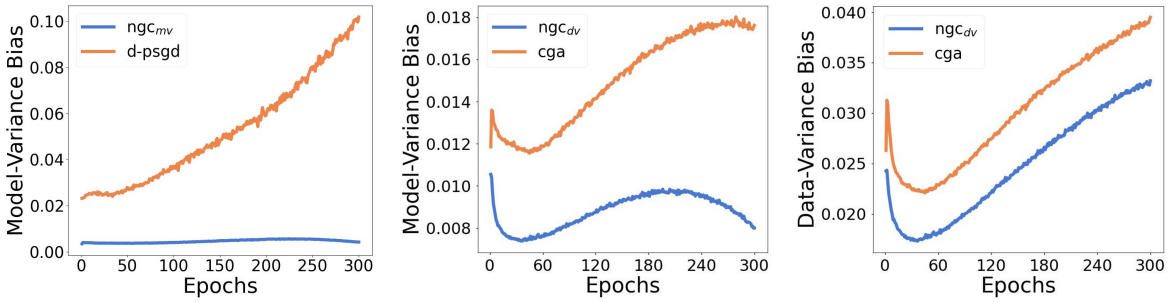
We empirically show the convergence characteristics of the proposed algorithm over IID and Non-IID data in Figures. 1(a),2(a), and Figures. 1(b), 2(b) respectively. For Non-IID data, we observe that there is a slight difference in convergence (as expected) with a slower rate for sparser topology (ring graph) compared to a denser counterpart (fully connected graph). Figure. 1(c), 2(c) shows the comparison of the convergence characteristics of the NGC_{mv} and NGC_{dv} algorithm with the respective baselines. For the same decentralized setup, We observe that NGC_{mv} and NGC_{dv} have lower validation loss than D-PSGD and CGA respectively. Analysis for 10 agents is presented in Appendix A.7.

Then we proceed to analyze the model variance and data variance bias terms when training with various decentralized methods. Figure. 3(a) shows that the model variance bias of NGC_{mv} is much lower than of the D-PSGD baseline resulting in the better performance of NGC_{mv}. We then compare the model variance and data variance bias terms of NGC_{dv} with CGA as shown in Figure. 3(b), and 3(c) respectively. We observe



(a) NGC_{dv} on IID data (b) NGC_{dv} on Non-IID data (c) Non-IID, ring topology

Figure 2: Average Validation loss of NGC_{dv} during training CIFAR-10 on a 5-layer CNN with 5 agents.



(a) Model variance bias of NGC_{mv} (b) Model variance bias of NGC_{dv} (c) Data variance bias of NGC_{dv}

Figure 3: Ablation study: CIFAR-10 dataset trained on a ring graph with 5 layer CNN.

that both the model variance and the data variance bias for NGC_{dv} are significantly lower than CGA. This is because CGA gives more importance to self-gradients as it updates in the descent direction that is close to self-gradients and is positively correlated to data-variant cross-gradients. In contrast, *NGC* accounts for the biases directly and gives equal importance to self-gradients and data-variant cross-gradients, thereby achieving superior performance.

5.4 Hardware Benefits

The proposed NGC_{dv} algorithm is superior in terms of memory and compute efficiency (see Table. 5 and Appendix. A.8) while having equal communication cost as compared to *CGA*. NGC_{dv} replaces the complicated quadratic projection step of CGA with a simple weighted averaging step. Since NGC_{dv} involves weighted averaging, an additional buffer to store the cross-gradients is not required. Weighted cross-gradients can be added to the self-gradient buffer. CGA stores all the cross-gradients in a matrix form for quadratic

Table 5: Comparison of communication, memory, and compute overheads per mini-batch compared to D-PSGD. m_s: model size, $\mathcal{N}_i$: number of neighbors, b: floating point precision, Q: compute for Quadratic Programming, B: compute for Backward Pass. Note that $\mathcal{O}(Q) >> \mathcal{O}(\mathcal{N}_i m_s)$ Nesterov & Todd (1997)

Method	Comm.	Memory	Compute
CGA	$\mathcal{O}(\mathcal{N}_i m_s)$	$\mathcal{O}(\mathcal{N}_i m_s)$	$\mathcal{O}(\mathcal{N}_i B + Q)$
CompCGA	$\mathcal{O}(\frac{\mathcal{N}_i m_s}{b})$	$\mathcal{O}(\mathcal{N}_i m_s)$	$\mathcal{O}(\mathcal{N}_i B + Q)$
NGC_{mv}	0	0	$\mathcal{O}(\mathcal{N}_i B + \mathcal{N}_i m_s)$
NGC_{dv}	$\mathcal{O}(\mathcal{N}_i m_s)$	0	$\mathcal{O}(\mathcal{N}_i B + \mathcal{N}_i m_s)$
CompNGC	$\mathcal{O}(\frac{\mathcal{N}_i m_s}{b})$	$\mathcal{O}(\mathcal{N}_i m_s)$	$\mathcal{O}(\mathcal{N}_i B + \mathcal{N}_i m_s)$

programming projection of the local gradient. Therefore, *NGC* has no memory overhead compared to the baseline D-PSGD algorithm, while CGA requires additional memory equivalent to the number of neighbors times model size. Moreover, the quadratic programming projection step Goldfarb & Idnani (1983) in CGA is much more expensive in compute and latency than the weighted averaging step of cross-gradients in *NGC*. Our evaluation clearly shows that *NGC* is superior to CGA in terms of test accuracy, memory efficiency, compute efficiency, and latency.

6 Discussion and Limitations

As mentioned in Section. 3.1, the *NGC* mixing weight α is either set to 0 or 1 resulting in NGC_{mv} or NGC_{dv} respectively. This implies that NGC_{mv} utilizes model-variant cross-gradients whereas, NGC_{dv} utilizes data-variant cross-gradients. However, the NGC_{dv} variant of the algorithm has access to both model-variant and data-variant cross-gradients. So, we conduct an ablation study mixing these two cross gradients by varying *NGC* mixing weight α in the range of $[0, 1]$. Figure. 4 depicts the change in average validation accuracy with α and we observe that $\alpha = 1$ performs best. Further, Figure. 3(b) shows that the model variant bias reduces for NGC_{dv} even though it does not explicitly utilize the model-variant cross-gradients. We hypothesize that, in NGC_{dv}, the use of data-variant cross-gradients for local updates followed by gossip averaging of model parameters inherently adds model-variant cross-gradients reducing model variance. Hence, explicitly mixing the two sets of cross-gradients is not necessary.

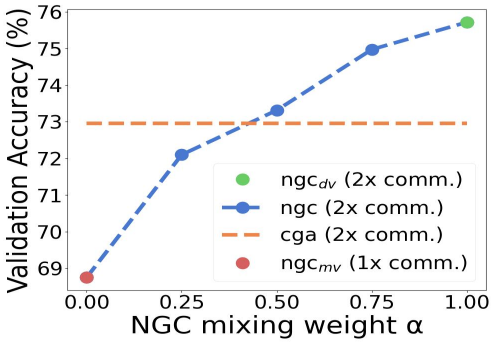


Figure 4: Varying the value of α while training a 5-layer CNN on CIFAR-10 distributed in a non-IID fashion across 10 agents in a ring topology.

In summary, for 1× communication constraints i.e., when agents do not have access to data-variant cross-gradients, NGC_{mv} effectively utilizes locally available model-variant cross-gradients to improve the accuracy. On the contrary, when 2× communication is allowed (non-zero α), the best performance is achieved by only utilizing the data-variant cross-gradients.

NGC_{mv} has one potential limitation compared to D-PSGD i.e., compute overhead. NGC_{mv} has to compute model-variant cross-gradients which requires every agent i to compute $(\mathcal{N}_i - 1)$ additional forward and backward passes. However, NGC_{dv} is significantly better in terms of latency, compute, and memory requirements compared to CGA as it simplifies the complicated quadratic projection step. Running the CGA algorithm on larger datasets like ImageNet is impractical due to the significant latency associated with the quadratic projection step. Table. 6 reports the accuracy and latency of training ImageNet in a non-IID fashion over a ring topology of 10 agents. We observe that CGA is $\sim 20\times$ slower than NGC_{dv} for the ImageNet dataset on ResNet-18. We estimate the training time for one simulation of CGA on ImageNet (90 epochs with 10-agent ring topology) to be around 26 days on 4 Nvidia A100s (80 GB) with Intel(R) Xeon(R) Gold 6338 CPU. Further, we would like to explore efficient methods to compute cross-gradients for *NGC* as a future research direction.

Table 6: Test Accuracy and Latency of ImageNet (non-IID) trained on ResNet-18 aver 10-agent Ring.

Trained for 60 epochs	D-PSGD	NGC_{mv}	CGA	NGC_{dv}
Accuracy (%)	25.02	42.66	-	49.42
Epoch Time (min)	7.6	18.6	406	21.5

7 Conclusion

Enabling decentralized training on non-IID data is key for ML applications to efficiently leverage the humongous amounts of user-generated private data. In this paper, we propose the *Neighborhood Gradient Clustering* (NGC) algorithm with two variants that improve decentralized learning on non-IID data. NGC_{mv} improves the performance of decentralized learning utilizing model-variant cross-gradients without any communication overhead. NGC_{dv} further increases these gains by operating on data-variant cross-gradients. Additionally, we present a compressed version of our algorithm ($CompNGC$) to reduce the communication overhead associated with data-variant cross-gradients of NGC_{dv}. We theoretically analyze the convergence characteristic and empirically validate the performance of the proposed techniques over different model architectures, datasets, graph sizes, and topologies. Finally, we compare the proposed algorithms with the current state-of-the-art decentralized learning algorithm on non-IID data and show superior performance with significantly less compute and memory requirements setting the new state-of-the-art for decentralized learning on non-IID data.

References

Alekh Agarwal and John C Duchi. Distributed delayed stochastic optimization. *Advances in neural information processing systems*, 24, 2011.

Sai Aparna Aketi, Amandeep Singh, and Jan Rabaey. Sparse-push: Communication-& energy-efficient decentralized distributed learning over directed & time-varying graphs with non-iid datasets. *arXiv preprint arXiv:2102.05715*, 2021.

Sai Aparna Aketi, Sangamesh Kodge, and Kaushik Roy. Low precision decentralized distributed training over iid and non-iid data. *Neural Networks*, 2022. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2022.08.032.

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

Aditya Balu, Zhanhong Jiang, Sin Yong Tan, Chinmay Hedge, Young M Lee, and Soumik Sarkar. Decentralized deep learning using momentum-accelerated consensus. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3675–3679. IEEE, 2021.

Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pp. 177–186. Springer, 2010.

Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc'aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, et al. Large scale distributed deep networks. *Advances in neural information processing systems*, 25, 2012.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Yasaman Esfandiari, Sin Yong Tan, Zhanhong Jiang, Aditya Balu, Ethan Herron, Chinmay Hegde, and Soumik Sarkar. Cross-gradient aggregation for decentralized learning from non-iid data. In *International Conference on Machine Learning*, pp. 3036–3046. PMLR, 2021.

Donald Goldfarb and Ashok Idnani. A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical programming*, 27(1):1–33, 1983.

Arya Ketabchi Haghighat, Varsha Ravichandra-Mouli, Pranamesh Chakraborty, Yasaman Esfandiari, Saeed Arabi, and Anuj Sharma. Applications of deep learning in intelligent transportation systems. *Journal of Big Data Analytics in Transportation*, 2(2):115–145, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Phillip Gibbons. The non-IID data quagmire of decentralized machine learning. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 4387–4398. PMLR, 13–18 Jul 2020.

Hamel Husain. Imagenette - a subset of 10 easily classified classes from the imagenet dataset. *https://github.com/fastai/imagenette*, 2018.

Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian Stich, and Martin Jaggi. Error feedback fixes signsgd and other gradient compression schemes. In *International Conference on Machine Learning*, pp. 3252–3261. PMLR, 2019.

Anastasia Koloskova, Tao Lin, Sebastian U Stich, and Martin Jaggi. Decentralized deep learning with arbitrary communication compression. *arXiv preprint arXiv:1907.09356*, 2019.

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

Jakub Konecny, H Brendan McMahan, Daniel Ramage, and Peter Richtarik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar (canadian institute for advanced research). *http://www.cs.toronto.edu/ kriz/cifar.html*, 2014.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. *Advances in Neural Information Processing Systems*, 30, 2017.

Tao Lin, Sai Praneeth Karimireddy, Sebastian Stich, and Martin Jaggi. Quasi-global momentum: Accelerating decentralized deep learning on heterogeneous data. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6654–6665. PMLR, 18–24 Jul 2021.

Hanxiao Liu, Andy Brock, Karen Simonyan, and Quoc Le. Evolving normalization-activation layers. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 13539–13550. Curran Associates, Inc., 2020.

Giorgi Nadiradze, Amirmojtaba Sabour, Dan Alistarh, Aditya Sharma, Ilia Markov, and Vitaly Aksenov. Swarmsgd: Scalable decentralized sgd with local updates. *arXiv preprint arXiv:1910.12308*, 2019.

Yu E Nesterov and Michael J Todd. Self-scaled barriers and interior-point methods for convex programming. *Mathematics of Operations research*, 22(1):1–42, 1997.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4510–4520, 2018.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Hanlin Tang, Xiangru Lian, Ming Yan, Ce Zhang, and Ji Liu. d^2: Decentralized training over decentralized data. In *International Conference on Machine Learning*, pp. 4848–4856. PMLR, 2018.

Hanlin Tang, Xiangru Lian, Shuang Qiu, Lei Yuan, Ce Zhang, Tong Zhang, and Ji Liu. Deepsqueeze: Decentralization meets error-compensated compression. *arXiv preprint arXiv:1907.07346*, 2019.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Lin Xiao and Stephen Boyd. Fast linear iterations for distributed averaging. *Systems & Control Letters*, 53 (1):65–78, 2004.

Hao Yu, Rong Jin, and Sen Yang. On the linear speedup analysis of communication efficient momentum sgd for distributed non-convex optimization. *arXiv preprint arXiv:1905.03817*, 2019.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

A Appendix

A.1 Proof of Lemma. 1

This section presents detailed proof for Lemma. 1. The *NGC* algorithm modifies the local gradients as follows

$$\widetilde{g}^i = (1-\alpha)\sum_{j\in\mathcal{N}_i} w_{ji}g^{ji} + \alpha\sum_{j\in\mathcal{N}_i} w_{ij}g^{ij} = g^i + \sum_{j\in\mathcal{N}_i} w_{ij}(g^{ij}-g^i) \quad (\text{NGC}_{\text{dv}} \text{ uses } \alpha = 1)$$
$$\widetilde{g}^i - g^i = \sum_{j\in\mathcal{N}_i} w_{ij}(\nabla F_j(x,d^j) - \nabla F_i(x,d^i))$$

Now we prove the lemma. 1 by applying expectation to the above inequality

$$\begin{aligned}
\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}(\tilde{\mathbf{g}}^i - \mathbf{g}^i)\right\|^2\right] &= \mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i} w_{ij}(\nabla F_j(x,d^j) - \nabla F_i(x,d^i))\right\|^2\right] \\
&\overset{(a)}{=} \mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i} w_{ij}(\nabla F_j(x,d^j) - \nabla F_i(x,d^i) - \mathbb{E}[\nabla F_j(x,d^j) - \nabla F_i(x,d^i)])\right\|^2\right] \\
&+ \mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i} w_{ij}(\mathbb{E}[\nabla F_j(x,d^j) - \nabla F_i(x,d^i)])\right\|^2\right] \\
&\overset{(b)}{=} \mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i} w_{ij}(\nabla F_j(x,d^j) - \nabla f_j(x)) - (\nabla F_i(x,d^i) - \nabla f_i(x))\right\|^2\right] \\
&+ \mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i} w_{ij}(\nabla f_j(x) - \nabla f_i(x))\right\|^2\right] \\
&\overset{(c)}{=} \frac{1}{N^2}\sum_{i=1}^{N}\mathbb{E}\left[\left\|\sum_{j\in\mathcal{N}_i} w_{ij}(\nabla F_j(x,d^j) - \nabla f_j(x)) - (\nabla F_i(x,d^i) - \nabla f_i(x))\right\|^2\right] \\
&+ \mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i} w_{ij}(\nabla f_j(x) - \nabla f_i(x))\right\|^2\right] \\
&\overset{(d)}{\leq} \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i} w_{ij}\mathbb{E}\left[\left\|\nabla F_j(x,d^j) - \nabla f_j(x) - (\nabla F_i(x,d^i) - \nabla f_i(x))\right\|^2\right] \\
&+ \frac{1}{N}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i} w_{ij}\mathbb{E}\left[\left\|\nabla f_j(x) - \nabla\mathcal{F}(x) + \nabla\mathcal{F}(x) - \nabla f_i(x)\right\|^2\right] \\
&\overset{(e)}{\leq} \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i} w_{ij}4\sigma^2 + \frac{1}{N}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i} w_{ij}\mathbb{E}\left[\left\|\nabla f_j(x) - \nabla\mathcal{F}(x) + \nabla\mathcal{F}(x) - \nabla f_i(x)\right\|^2\right] \\
&= \frac{4N\sigma^2}{N^2} + \frac{1}{N}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i} w_{ij}\mathbb{E}\left[\left\|\nabla f_j(x) - \nabla\mathcal{F}(x) + \nabla\mathcal{F}(x) - \nabla f_i(x)\right\|^2\right] \quad (\because \sum_{j\in\mathcal{N}_i} w_{ij} = 1) \\
&\overset{(f)}{\leq} \frac{4\sigma^2}{N} + \frac{1}{N}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i} w_{ij}\left(2\mathbb{E}\left[\left\|\nabla f_j(x) - \nabla\mathcal{F}(x)\right\|^2\right] + 2\mathbb{E}\left[\left\|\nabla f_i(x) - \nabla\mathcal{F}(x)\right\|^2\right]\right) \\
&\overset{(g)}{\leq} \frac{4\sigma^2}{N} + \frac{1}{N}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i} w_{ij}4\zeta^2
\end{aligned}$$

$$= \frac{4\sigma^2}{N} + 4\zeta^2$$

(a) identity $\mathbb{E}[||Z||^2] = \mathbb{E}[||Z - E[Z]||^2] + \mathbb{E}[||\mathbb{E}[Z]||^2]$ holds for any random vector Z; (b) the fact $\mathbb{E}[\nabla F_i(x, d^i)] = \nabla f_i(x)$; (c) $\nabla F_i(x, d^i) - \nabla f_i(x)$ are independent random vectors with 0 mean and $\mathbb{E}[|| \sum_{i=1}^N Z_i||] = \sum_{i=1}^N \mathbb{E}[||Z_i||^2]$ holds when Z_i are independent with mean zero; (d) jensen's inequality; (e) follows from assumption 2 $\mathbb{E}_{d\sim D_i}||\nabla F_i(x; d) - \nabla f_i(x)||^2 \leq \sigma^2 \ \forall i \in [1, N]$; (f) the basic inequality $||a_1 + a_2||^2 \leq 2||a_1||^2 + 2||a_2||^2$ for any vectors a_1, a_2; and (g) follows from assumption 2 that $||\nabla f_i(x) - \nabla \mathcal{F}(x)||^2 \leq \zeta^2 \ \ \forall i \in [1, N]$

$\therefore$ we have following bound given by lemma. 1: $\mathbb{E}[||\widetilde{g}^i - g^i||^2] \leq 4(\frac{\sigma^2}{N} + \zeta^2)$

A.2 Convergence Analysis

In this section, we present the proof for our main theorem. 1 indicating the convergence of the proposed NGC_{dv} algorithm. Before we proceed to the proof of the theorem, we present a lemma showing that NGC achieves consensus among the different agents.

A.2.1 Bound for Consensus Error

Lemma 2. *Given assumptions 1-3 and lemma 1, the distance between the average sequence iterate $\bar{x}^k$ and the sequence iterates x_i^k's generated NGC (i.e., the consensus error of the proposed algorithm) is given by*

$$\sum_{k=0}^{K} \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}[||\bar{x}_k - x_k^i||^2] \leq \quad \frac{14\eta^2(\sigma^2 + \zeta^2)}{(1-\beta)^2(1-\sqrt{\rho})^2} K + \frac{6\eta^2}{(1-\beta)^2(1-\sqrt{\rho})} \sum_{k=0}^{K-1} \mathbb{E}[\| \frac{1}{N} \sum_{i=1}^{N} \nabla f_i(\mathbf{x}_k^i)\|^2] \tag{12}$$

where $K \geq 1$ and $\beta \in [0, 1)$ is the momentum coefficient.

To prove Lemma 2 and Theorem. 1, we follow the similar approach as Esfandiari et al. (2021). Hence, we also define the following auxiliary sequence along with Lemma 3

$$\bar{\mathbf{z}}_k := \frac{1}{1-\beta}\bar{\mathbf{x}}_k - \frac{\beta}{1-\beta}\bar{\mathbf{x}}_{k-1} \tag{13}$$

Where $k > 0$ and $\mathbf{x}_k$ is obtained by multiplying the update law by $\frac{1}{N}\mathbf{1}\mathbf{1}^\top$, ($\mathbf{1}$ is the column vector with entries being 1).

$$\begin{aligned} \bar{\mathbf{v}}_k &= \beta\bar{\mathbf{v}}_{k-1} - \eta\frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{g}}_{k-1}^i \\ \bar{\mathbf{x}}_k &= \bar{\mathbf{x}}_{k-1} + \bar{\mathbf{v}}_k \end{aligned} \tag{14}$$

If $k = 0$ then $\bar{\mathbf{z}}^k = \bar{\mathbf{x}}^k$. For the rest of the analysis, the initial value will be directly set to 0.

To prove Lemma. 2, we use the following facts.

$$\bar{\mathbf{z}}_{k+1} - \bar{\mathbf{z}}_k = -\frac{\eta}{1-\beta}\frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{g}}_k^i. \tag{15}$$

$$\sum_{k=0}^{K-1} \|\bar{\mathbf{z}}_k - \bar{\mathbf{x}}_k\|^2 \leq \frac{\eta^2\beta^2}{(1-\beta)^4} \sum_{k=0}^{K-1} \left\| \frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{g}}_k^i \right\|^2. \tag{16}$$

Note that the proof for Eq. 15 and Eq. 16 can be found in Esfandiari et al. (2021) as lemma-3 and lemma-4 respectively.

Before proceeding to prove Lemma 2, we introduce some key notations and facts that serve to characterize the Lemma similar to Esfandiari et al. (2021).

We define the following notations:

$$\begin{aligned}
\tilde{\mathbf{G}}_k &\triangleq [\tilde{\mathbf{g}}_k^1, \tilde{\mathbf{g}}_k^2, ..., \tilde{\mathbf{g}}_k^N] \\
\mathbf{V}_k &\triangleq [\mathbf{v}_k^1, \mathbf{v}_k^2, ..., \mathbf{v}_k^N] \\
\mathbf{X}_k &\triangleq [\mathbf{x}_k^1, \mathbf{x}_k^2, ..., \mathbf{x}_k^N] \\
\mathbf{G}_k &\triangleq [\mathbf{g}_k^1, \mathbf{g}_k^2, ..., \mathbf{g}_k^N] \\
\mathbf{H}_k &\triangleq [\nabla f_1(\mathbf{x}_k^1), \nabla f_2(\mathbf{x}_k^2), ..., \nabla f_N(\mathbf{x}_k^N)]
\end{aligned} \tag{17}$$

We can observe that the above matrices are all with dimension $d \times N$ such that any matrix $\mathbf{A}$ satisfies $\|\mathbf{A}\|_{\mathfrak{F}}^2 = \sum_{i=1}^N \|\mathbf{a}_i\|^2$, where $\mathbf{a}_i$ is the i-th column of the matrix $\mathbf{A}$. Thus, we can obtain that:

$$\|\mathbf{X}_k(\mathbf{I} - \mathbf{Q})\|_{\mathfrak{F}}^2 = \sum_{i=1}^N \|\mathbf{x}_k^i - \bar{\mathbf{x}}_k\|^2. \tag{18}$$

Define $\mathbf{Q} = \frac{1}{N}\mathbf{1}\mathbf{1}^\top$. For each doubly stochastic matrix $\mathbf{W}$, the following properties can be obtained

$$\begin{aligned}
&(i) \quad \mathbf{QW} = \mathbf{WQ}; \\
&(ii) \quad (\mathbf{I} - \mathbf{Q})\mathbf{W} = \mathbf{W}(\mathbf{I} - \mathbf{Q}); \\
&(iii) \quad \text{For any integer } k \geq 1,\ \|(\mathbf{I} - \mathbf{Q})\mathbf{W}\|_{\mathfrak{S}} \leq (\sqrt{\rho})^k, \text{ where } \|\cdot\|_{\mathfrak{S}} \text{ is the spectrum norm of a matrix.}
\end{aligned} \tag{19}$$

Let $\mathbf{A}_i, i \in \{1, 2, ..., N\}$ be N arbitrary real square matrices. It follows that

$$\|\sum_{i=1}^N \mathbf{A}_i\|_{\mathfrak{F}}^2 \leq \sum_{i=1}^N \sum_{j=1}^N \|\mathbf{A}_i\|_{\mathfrak{F}} \|\mathbf{A}_j\|_{\mathfrak{F}}. \tag{20}$$

We present the following auxiliary lemmas to prove the lemma. 2

Lemma 3. *Let all assumptions hold. Let g^i be the unbiased estimate of $\nabla f_i(\mathbf{x}^i)$ at the point $\mathbf{x}^i$ such that $\mathbb{E}[\mathbf{g}^i] = \nabla f_i(\mathbf{x}^i)$, for all $i \in [N] := \{1, 2, ..., N\}$. Thus the following relationship holds*

$$\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^N \tilde{\mathbf{g}}^i\right\|^2\right] \leq \frac{10\sigma^2}{N} + 8\zeta^2 + 2\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^N \nabla f_i(\mathbf{x}^i)\right\|^2\right] \tag{21}$$

Proof for Lemma 3:

$$\begin{aligned}
\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^N \tilde{\mathbf{g}}^i\right\|^2\right] &= \mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^N (\tilde{\mathbf{g}}^i - \mathbf{g}^i + \mathbf{g}^i)\right\|^2\right] \\
&= \mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^N (\tilde{\mathbf{g}}^i - \mathbf{g}^i) + \frac{1}{N}\sum_{i=1}^N \mathbf{g}^i\right\|^2\right] \\
&\overset{a}{\leq} 2\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^N (\tilde{\mathbf{g}}^i - \mathbf{g}^i)\right\|^2\right] + 2\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^N \mathbf{g}^i\right\|^2\right] \\
&\overset{b}{\leq} 8(\frac{\sigma^2}{N} + \zeta^2) + \frac{2\sigma^2}{N} + 2\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^N \nabla f_i(\mathbf{x}^i)\right\|^2\right] \\
&= \frac{10\sigma^2}{N} + 8\zeta^2 + 2\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^N \nabla f_i(\mathbf{x}^i)\right\|^2\right]
\end{aligned}$$

(a) refers to the fact that the inequality $\|\mathbf{a}+\mathbf{b}\|^2 \leq 2\|\mathbf{a}\|^2 + 2\|\mathbf{b}\|^2$. (b) The first term uses Lemma 1 and the second term uses the conclusion of Lemma 1 in Yu et al. (2019).

Lemma 4. *Let all assumptions hold. Let g^i be the unbiased estimate of $\nabla f_i(\mathbf{x}^i)$ at the point $\mathbf{x}^i$ such that $\mathbb{E}[\mathbf{g}^i] = \nabla f_i(\mathbf{x}^i)$, for all $i \in [N] := \{1, 2, ..., N\}$. Thus the following relationship holds*

$$\mathbb{E}\left[\left\|\tilde{\mathbf{G}}_\tau - \mathbf{G}_\tau\right\|_{\mathfrak{F}}^2\right] \leq 4N(\sigma^2 + \zeta^2) \tag{22}$$

Proof for Lemma 4 is similar to Lemma 1 and is as follows:

$$\begin{aligned}
\mathbb{E}\left[\left\|\tilde{\mathbf{G}}_\tau - \mathbf{G}_\tau\right\|_{\mathfrak{F}}^2\right] &= \sum_{i=1}^{N} \mathbb{E}\left[\left\| \sum_{j\in\mathcal{N}_i} w_{ij}(\nabla F_j(x, d^j) - \nabla F_i(x, d^i))\right\|^2\right] \\
&= \sum_{i=1}^{N} \mathbb{E}\left[\left\| \sum_{j\in\mathcal{N}_i} w_{ij}(\nabla F_j(x, d^j) - \nabla f_j(x)) - (\nabla F_i(x, d^i) - \nabla f_i(x))\right\|^2\right] \\
&+ \sum_{i=1}^{N} \mathbb{E}\left[\left\| \sum_{j\in\mathcal{N}_i} w_{ij}(\nabla f_j(x) - \nabla f_i(x))\right\|^2\right] \\
&= \mathbb{E}\left[\left\| \sum_{j\in\mathcal{N}_i} w_{ij}(\nabla F_j(x, d^j) - \nabla F_i(x, d^i) - \mathbb{E}[\nabla F_j(x, d^j) - \nabla F_i(x, d^i)])\right\|^2\right] \\
&\leq \sum_{i=1}^{N} \sum_{j\in\mathcal{N}_i} w_{ij}\mathbb{E}\left[\left\|\nabla F_j(x, d^j) - \nabla f_j(x) - (\nabla F_i(x, d^i) - \nabla f_i(x))\right\|^2\right] \\
&+ \sum_{i=1}^{N} \sum_{j\in\mathcal{N}_i} w_{ij}\mathbb{E}\left[\left\|\nabla f_j(x) - \nabla \mathcal{F}(x) + \nabla \mathcal{F}(x) - \nabla f_i(x)\right\|^2\right] \\
&\leq 4N(\sigma^2 + \zeta^2) \quad \text{(follows from assumption 2)}
\end{aligned}$$

The properties shown in Eq. 19 and 20 have been well established and in this context, we skip the proof here. We are now ready to prove Lemma 2.

Proof for Lemma 2: Since $\mathbf{X}_k = \mathbf{X}_{k-1}\mathbf{W} + \mathbf{V}_k$ we have:

$$\mathbf{X}_k(\mathbf{I} - \mathbf{Q}) = \mathbf{X}_{k-1}(\mathbf{I} - \mathbf{Q})\mathbf{W} + \mathbf{V}_k(\mathbf{I} - \mathbf{Q}) \tag{23}$$

Applying the above equation k times we have:

$$\mathbf{X}_k(\mathbf{I} - \mathbf{Q}) = \mathbf{X}_0(\mathbf{I} - \mathbf{Q})\mathbf{W}^k + \sum_{\tau=1}^{k} \mathbf{V}_\tau(\mathbf{I} - \mathbf{Q})\mathbf{W}^{k-\tau} \overset{\mathbf{X}_0=0}{=} \sum_{\tau=1}^{k} \mathbf{V}_\tau(\mathbf{I} - \mathbf{Q})\mathbf{W}^{k-\tau} \tag{24}$$

As $\bar{\mathbf{V}}_k = \beta\bar{\mathbf{V}}_{k-1} - \eta\frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{G}}_{k-1}^i \overset{\mathbf{V}_0=0}{=} -\eta\frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{G}}_{k-1}^i$, we can get:

$$\begin{aligned}
\mathbf{X}_k(\mathbf{I} - \mathbf{Q}) &= -\eta\sum_{\tau=1}^{k}\sum_{l=0}^{\tau-1}\tilde{\mathbf{G}}_l\beta^{\tau-1-l}(\mathbf{I} - \mathbf{Q})\mathbf{W}^{k-\tau} \\
&= -\eta\sum_{\tau=1}^{k}\sum_{l=0}^{\tau-1}\tilde{\mathbf{G}}_l\beta^{\tau-1-l}\mathbf{W}^{k-\tau}(\mathbf{I} - \mathbf{Q}) - \eta\sum_{n=1}^{k-1}\tilde{\mathbf{G}}_n[\sum_{l=n+1}^{k}\beta^{l-1-n}\mathbf{W}^{k-1-n}(\mathbf{I} - \mathbf{Q}) \\
&= -\eta\sum_{\tau=0}^{k-1}\frac{1-\beta^{k-\tau}}{1-\beta}\tilde{\mathbf{G}}_\tau(\mathbf{I} - \mathbf{Q})\mathbf{W}^{k-1-\tau}.
\end{aligned} \tag{25}$$

Therefore, for $k \geq 1$, we have:

$$\begin{aligned}\mathbb{E}\left[\left\|\mathbf{X}_k(\mathbf{I}-\mathbf{Q})\right\|_{\mathfrak{F}}^2\right] &= \eta^2\mathbb{E}\left[\left\|\sum_{\tau=0}^{k-1}\frac{1-\beta^{k-\tau}}{1-\beta}\tilde{\mathbf{G}}_\tau(\mathbf{I}-\mathbf{Q})\mathbf{W}^{k-1-\tau}\right\|_{\mathfrak{F}}^2\right] \\ &\overset{a}{\leq} \underbrace{2\eta^2\mathbb{E}\left[\left\|\sum_{\tau=0}^{k-1}\frac{1-\beta^{k-\tau}}{1-\beta}(\tilde{\mathbf{G}}_\tau-\mathbf{G}_\tau)(\mathbf{I}-\mathbf{Q})\mathbf{W}^{k-1-\tau}\right\|_{\mathfrak{F}}^2\right]}_{I} + \\ &\underbrace{2\eta^2\mathbb{E}\left[\left\|\sum_{\tau=0}^{k-1}\frac{1-\beta^{k-\tau}}{1-\beta}\mathbf{G}_\tau(\mathbf{I}-\mathbf{Q})\mathbf{W}^{k-1-\tau}\right\|_{\mathfrak{F}}^2\right]}_{II}\end{aligned} \tag{26}$$

(a) follows from the inequality $\|\mathbf{A}+\mathbf{B}\|_{\mathfrak{F}}^2 \leq 2\|\mathbf{A}\|_{\mathfrak{F}}^2 + 2\|\mathbf{B}\|_{\mathfrak{F}}^2$.

We develop upper bounds of the term **I**:

$$\begin{aligned}\mathbb{E}\left[\left\|\sum_{\tau=0}^{k-1}\frac{1-\beta^{k-\tau}}{1-\beta}(\tilde{\mathbf{G}}_\tau-\mathbf{G}_\tau)(\mathbf{I}-\mathbf{Q})\mathbf{W}^{k-1-\tau}\right\|_{\mathfrak{F}}^2\right] &\overset{a}{\leq} \sum_{\tau=0}^{k-1}\mathbb{E}\left[\left\|\frac{1-\beta^{k-\tau}}{1-\beta}(\tilde{\mathbf{G}}_\tau-\mathbf{G}_\tau)(\mathbf{I}-\mathbf{Q})\mathbf{W}^{k-1-\tau}\right\|_{\mathfrak{F}}^2\right] \\ &\overset{b}{\leq} \frac{1}{(1-\beta)^2}\sum_{\tau=0}^{k-1}\rho^{k-1-\tau}\mathbb{E}\left[\left\|\tilde{\mathbf{G}}_\tau-\mathbf{G}_\tau\right\|_{\mathfrak{F}}^2\right] \\ &\overset{c}{\leq} \frac{4}{(1-\beta)^2}\sum_{\tau=0}^{k-1}\rho^{k-1-\tau}N(\sigma^2+\zeta^2) \overset{d}{\leq} \frac{4N(\sigma^2+\zeta^2)}{(1-\beta)^2(1-\rho)}\end{aligned} \tag{27}$$

(a) follows from Jensen's inequality. (b) follows from the inequality $|\frac{1-\beta^{k-\tau}}{1-\beta}| \leq \frac{1}{1-\beta}$. (c) follows from Lemma 2 and Frobenius norm. (d) follows from Assumption 3.

We then proceed to find the upper bound for term **II**.

$$\begin{aligned}\mathbb{E}\left[\left\|\sum_{\tau=0}^{k-1}\frac{1-\beta^{k-\tau}}{1-\beta}\mathbf{G}_\tau(\mathbf{I}-\mathbf{Q})\mathbf{W}^{k-1-\tau}\right\|_{\mathfrak{F}}^2\right] &\overset{a}{\leq} \sum_{\tau=0}^{k-1}\sum_{\tau'=0}^{k-1}\mathbb{E}\left[\left\|\frac{1-\beta^{k-\tau}}{1-\beta}\mathbf{G}_\tau(\mathbf{I}-\mathbf{Q})\mathbf{W}^{k-1-\tau}\right\|_{\mathfrak{F}}\right. \\ &\qquad\left.\left\|\frac{1-\beta^{k-\tau}}{1-\beta}\mathbf{G}_{\tau'}(\mathbf{I}-\mathbf{Q})\mathbf{W}^{k-1-\tau'}\right\|_{\mathfrak{F}}\right] \\ &\leq \frac{1}{(1-\beta)^2}\sum_{\tau=0}^{k-1}\sum_{\tau'=0}^{k-1}\rho^{(k-1-\frac{\tau+\tau'}{2})}\mathbb{E}\left[\|\mathbf{G}_\tau\|_{\mathfrak{F}}\|\mathbf{G}_{\tau'}\|_{\mathfrak{F}}\right] \\ &\overset{b}{\leq} \frac{1}{(1-\beta)^2}\sum_{\tau=0}^{k-1}\sum_{\tau'=0}^{k-1}\rho^{(k-1-\frac{\tau+\tau'}{2})}\left(\frac{1}{2}\mathbb{E}[\|\mathbf{G}_\tau\|_{\mathfrak{F}}^2]+\frac{1}{2}\mathbb{E}[\|\mathbf{G}_{\tau'}\|_{\mathfrak{F}}^2]\right) \\ &= \frac{1}{(1-\beta)^2}\sum_{\tau=0}^{k-1}\sum_{\tau'=0}^{k-1}\rho^{(k-1-\frac{\tau+\tau'}{2})}\mathbb{E}[\|\mathbf{G}_\tau\|_{\mathfrak{F}}^2] \\ &\overset{c}{\leq} \frac{1}{(1-\beta)^2(1-\sqrt{\rho})}\sum_{\tau=0}^{k-1}\rho^{(\frac{k-1-\tau}{2})}\mathbb{E}[\|\mathbf{G}_\tau\|_{\mathfrak{F}}^2]\end{aligned} \tag{28}$$

(a) follows from Eq. 20. (b) follows from the inequality $xy \leq \frac{1}{2}(x^2+y^2)$ for any two real numbers x, y. (c) is derived using $\sum_{\tau_1=0}^{k-1}\rho^{k-1-\frac{\tau_1+\tau}{2}} \leq \frac{\rho^{\frac{k-1-\tau}{2}}}{1-\sqrt{\rho}}$.

We then proceed with finding the bounds for $\mathbb{E}[\|\mathbf{G}_\tau\|_{\mathfrak{F}}^2]$:

$$\begin{aligned}\mathbb{E}[\|\mathbf{G}_\tau\|_{\mathfrak{F}}^2] &= \mathbb{E}[\|\mathbf{G}_\tau - \mathbf{H}_\tau + \mathbf{H}_\tau - \mathbf{H}_\tau\mathbf{Q} + \mathbf{H}_\tau\mathbf{Q}\|_{\mathfrak{F}}^2] \\ &\leq 3\mathbb{E}[\|\mathbf{G}_\tau - \mathbf{H}_\tau\|_{\mathfrak{F}}^2] + 3\mathbb{E}[\|\mathbf{H}_\tau(I-\mathbf{Q})\|_{\mathfrak{F}}^2] + 3\mathbb{E}[\|\mathbf{H}_\tau\mathbf{Q}\|_{\mathfrak{F}}^2] \\ &\overset{a}{\leq} 3N\sigma^2 + 3N\zeta^2 + 3\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_\tau^i)\|^2]\end{aligned} \tag{29}$$

(a) holds because $\mathbb{E}[\|\mathbf{H}_\tau\mathbf{Q}\|_{\mathfrak{F}}^2] \leq \mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_\tau^i)\|^2]$
Substituting (29) in (28):

$$\begin{aligned}\mathbb{E}\left[\left\|\sum_{\tau=0}^{k-1}\frac{1-\beta^{k-\tau}}{1-\beta}\mathbf{G}_\tau(\mathbf{I}-\mathbf{Q})\mathbf{W}^{k-1-\tau}\right\|_{\mathfrak{F}}^2\right] &\leq \frac{1}{(1-\beta)^2(1-\sqrt{\rho})}\sum_{\tau=0}^{k-1}\rho^{(\frac{k-1-\tau}{2})}\left[3N\sigma^2 + 3N\zeta^2 + 3N\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_\tau^i)\|^2]\right] \\ &\leq \frac{3N(\sigma^2+\zeta^2)}{(1-\beta)^2(1-\sqrt{\rho})^2} + \frac{3N}{(1-\beta)^2(1-\sqrt{\rho})}\sum_{\tau=0}^{k-1}\rho^{(\frac{k-1-\tau}{2})}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_\tau^i)\|^2]\end{aligned} \tag{30}$$

substituting (30) and (27) into the main inequality (26):

$$\begin{aligned}\mathbb{E}\left[\left\|\mathbf{X}_k(\mathbf{I}-\mathbf{Q})\right\|_{\mathfrak{F}}^2\right] &\leq \frac{8\eta^2N(\sigma^2+\zeta^2)}{(1-\beta)^2(1-\rho)} + \frac{2\eta^2}{(1-\beta)^2(1-\sqrt{\rho})}\Bigg(\frac{3N(\sigma^2)}{1-\sqrt{\rho}} + \frac{3N(\zeta^2)}{1-\sqrt{\rho}} + \\ &\qquad 3N\sum_{\tau=0}^{k-1}\rho^{(\frac{k-1-\tau}{2})}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_\tau^i)\|^2]\Bigg) \\ &= \frac{2\eta^2}{(1-\beta)^2}\left(\frac{4N(\sigma^2+\zeta^2)}{1-\rho} + \frac{3N\sigma^2}{(1-\sqrt{\rho})^2} + \frac{3N\zeta^2}{(1-\sqrt{\rho})^2}\right) + \\ &\qquad \frac{6N\eta^2}{(1-\beta)^2(1-\sqrt{\rho})}\sum_{\tau=0}^{k-1}\rho^{(\frac{k-1-\tau}{2})}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_\tau^i)\|^2] \\ &\leq \frac{14N\eta^2}{(1-\beta)^2}\left(\frac{\sigma^2}{(1-\sqrt{\rho})^2} + \frac{\zeta^2}{(1-\sqrt{\rho})^2}\right) + \frac{6N\eta^2}{(1-\beta)^2(1-\sqrt{\rho})}\sum_{\tau=0}^{k-1}\rho^{(\frac{k-1-\tau}{2})}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_\tau^i)\|^2]\end{aligned} \tag{31}$$

Summing over $k \in \{1, \ldots, K-1\}$ and noting that $\mathbb{E}\left[\left\|\mathbf{X}_0(\mathbf{I}-\mathbf{Q})\right\|_{\mathfrak{F}}^2\right] = 0$:

$$\begin{aligned}\sum_{k=1}^{K-1}\mathbb{E}\left[\left\|\mathbf{X}_k(\mathbf{I}-\mathbf{Q})\right\|_{\mathfrak{F}}^2\right] &\leq CK + \frac{6N\eta^2}{(1-\beta)^2(1-\sqrt{\rho})}\sum_{k=1}^{K-1}\sum_{\tau=0}^{k-1}\rho^{(\frac{k-1-\tau}{2})}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_\tau^i)\|^2] \\ &\leq CK + \frac{6N\eta^2}{(1-\beta)^2(1-\sqrt{\rho})}\sum_{k=0}^{K-1}\frac{1-\rho^{(\frac{K-1-k}{2})}}{1-\sqrt{\rho}}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_k^i)\|^2] \\ &\leq CK + \frac{6N\eta^2}{(1-\beta)^2(1-\sqrt{\rho})}\sum_{k=0}^{K-1}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_k^i)\|^2]\end{aligned} \tag{32}$$

Where $C = \frac{14N\eta^2}{(1-\beta)^2}\left(\frac{\sigma^2}{(1-\sqrt{\rho})^2} + \frac{\zeta^2}{(1-\sqrt{\rho})^2}\right)$.

Dividing both sides by N:

$$\sum_{k=1}^{K-1} \frac{1}{N}\mathbb{E}\left[\left\|\mathbf{X}_k(\mathbf{I}-\mathbf{Q})\right\|_{\mathfrak{F}}^2\right] \leq \frac{14\eta^2}{(1-\beta)^2}\left(\frac{\sigma^2}{(1-\sqrt{\rho})^2} + \frac{\zeta^2}{(1-\sqrt{\rho})^2}\right)K + \frac{6\eta^2}{(1-\beta)^2(1-\sqrt{\rho})}\sum_{k=0}^{K-1}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_k^i)\|^2] \tag{33}$$

Hence, we obtain the following as the lemma. 2:

$$\sum_{k=0}^{K-1} \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}\left[\left\|\bar{\mathbf{x}}_k - \mathbf{x}_k^i\right\|^2\right] \leq \frac{14\eta^2(\sigma^2+\zeta^2)}{(1-\beta)^2(1-\sqrt{\rho})^2}K + \frac{6\eta^2}{(1-\beta)^2(1-\sqrt{\rho})}\sum_{k=0}^{K-1}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_k^i)\|^2] \tag{34}$$

A.2.2 Proof for Theorem. 1

Proof: Using the L-smoothness properties for $\mathcal{F}$ we have:

$$\mathbb{E}[\mathcal{F}(\bar{\mathbf{z}}_{k+1})] \leq \mathbb{E}[\mathcal{F}(\bar{\mathbf{z}}_k)] + \mathbb{E}[\langle\nabla\mathcal{F}(\bar{\mathbf{z}}_k), \bar{\mathbf{z}}_{k+1} - \bar{\mathbf{z}}_k\rangle] + \frac{L}{2}\mathbb{E}[\|\bar{\mathbf{z}}_{k+1} - \bar{\mathbf{z}}_k\|^2] \tag{35}$$

Using Eq. 15 we have:

$$\mathbb{E}[\langle\nabla\mathcal{F}(\bar{\mathbf{z}}_k), \bar{\mathbf{z}}_{k+1} - \bar{\mathbf{z}}_k\rangle] = \frac{-\eta}{1-\beta}\mathbb{E}[\langle\nabla\mathcal{F}(\bar{\mathbf{z}}_k), \frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{g}}_k^i\rangle] = \underbrace{\frac{-\eta}{1-\beta}\mathbb{E}[\langle\nabla\mathcal{F}(\bar{\mathbf{z}}_k) - \nabla\mathcal{F}(\bar{\mathbf{x}}_k), \frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{g}}_k^i\rangle]}_{T_1} - \underbrace{\frac{\eta}{1-\beta}\mathbb{E}[\langle\nabla\mathcal{F}(\bar{\mathbf{x}}_k), \frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{g}}_k^i\rangle]}_{T_2} \tag{36}$$

We proceed by bounding T_1:

$$\begin{aligned}
&\frac{-\eta}{1-\beta}\mathbb{E}[\langle\nabla\mathcal{F}(\bar{\mathbf{z}}_k) - \nabla\mathcal{F}(\bar{\mathbf{x}}_k), \frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{g}}_k^i\rangle] \\
&\overset{(i)}{\leq} \frac{(1-\beta)}{2\beta L}\mathbb{E}[\|\nabla\mathcal{F}(\bar{\mathbf{z}}_k) - \nabla\mathcal{F}(\bar{\mathbf{x}}_k)\|^2] + \frac{\beta L\eta^2}{2(1-\beta)^3}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{g}}_k^i\|^2] \\
&\overset{(ii)}{\leq} \frac{(1-\beta)L}{2\beta}\mathbb{E}[\|\bar{\mathbf{z}}_k - \bar{\mathbf{x}}_k\|^2] + \frac{\beta L\eta^2}{2(1-\beta)^3}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{g}}_k^i\|^2]
\end{aligned} \tag{37}$$

(i) holds as $\langle\mathbf{a},\mathbf{b}\rangle \leq \frac{1}{2}\|\mathbf{a}\|^2 + \frac{1}{2}\|\mathbf{b}\|^2$ where $\mathbf{a} = \frac{\sqrt{1-\beta}}{\sqrt{\beta L}}(\nabla\mathcal{F}(\bar{\mathbf{z}}_k) - \nabla\mathcal{F}(\bar{\mathbf{x}}_k))$ and $\mathbf{b} = -\frac{\eta\sqrt{\beta L}}{(1-\beta)^{\frac{3}{2}}}\frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{g}}_k^i$. and (ii) uses the fact that $\mathcal{F}$ is L-smooth.

We split the term T_2 as follows:

$$\begin{aligned}&\langle \nabla \mathcal{F}\left(\bar{\mathbf{x}}_k\right), \frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{g}}_k^i\rangle = \langle \nabla \mathcal{F}\left(\bar{\mathbf{x}}_k\right), \frac{1}{N}\sum_{i=1}^{N}\left(\tilde{\mathbf{g}}_k^i - \mathbf{g}_k^i + \mathbf{g}_k^i\right)\rangle = \\ &\underbrace{\langle \nabla \mathcal{F}\left(\bar{\mathbf{x}}_k\right), \frac{1}{N}\sum_{i=1}^{N}\left(\tilde{\mathbf{g}}_k^i - \mathbf{g}_k^i\right)\rangle}_{T_3} + \underbrace{\langle \nabla \mathcal{F}\left(\bar{\mathbf{x}}_k\right), \frac{1}{N}\sum_{i=1}^{N}\mathbf{g}_k^i\rangle}_{T_4}\end{aligned} \tag{38}$$

Now, We first analyze the term T_3:

$$\begin{aligned}&\frac{-\eta}{(1-\beta)}\mathbb{E}[\langle \nabla \mathcal{F}\left(\bar{\mathbf{x}}_k\right), \frac{1}{N}\sum_{i=1}^{N}\left(\tilde{\mathbf{g}}_k^i - \mathbf{g}_k^i\right)\rangle] \\ &\leq \frac{(1-\beta)}{2\beta L}\mathbb{E}[\|\nabla\mathcal{F}(\bar{\mathbf{x}}_k)\|^2] + \frac{\eta^2\beta L}{2(1-\beta)^3}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}(\tilde{\mathbf{g}}_k^i - \mathbf{g}_k^i)\|^2]\end{aligned} \tag{39}$$

This holds as $\langle \mathbf{a}, \mathbf{b}\rangle \leq \frac{1}{2}\|\mathbf{a}\|^2 + \frac{1}{2}\|\mathbf{b}\|^2$ where $\mathbf{a} = \frac{-\sqrt{1-\beta}}{\sqrt{\beta L}}\nabla\mathcal{F}(\bar{\mathbf{x}}_k)$ and $\mathbf{b} = \frac{\eta\sqrt{\beta L}}{(1-\beta)^{\frac{3}{2}}}\frac{1}{N}\sum_{i=1}^{N}(\tilde{\mathbf{g}}_k^i - \mathbf{g}_k^i)$.

Analyzing the term T_4:

$$\mathbb{E}\Big[\langle \nabla \mathcal{F}\left(\bar{\mathbf{x}}_k\right), \frac{1}{N}\sum_{i=1}^{N}\mathbf{g}_k^i\rangle\Big] = \mathbb{E}\Big[\langle \nabla\mathcal{F}(\bar{\mathbf{x}}_k), \frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_k^i)\rangle\Big] \tag{40}$$

Using the equity $\langle \mathbf{a}, \mathbf{b}\rangle = \frac{1}{2}[\|\mathbf{a}\|^2 + \|\mathbf{b}\|^2 - \|\mathbf{a}-\mathbf{b}\|^2]$, we have :

$$\begin{aligned}\langle \nabla \mathcal{F}\left(\bar{\mathbf{x}}_k\right), \frac{1}{N}\sum_{i=1}^{N}\nabla f_i\left(\mathbf{x}_k^i\right)\rangle &= \frac{1}{2}\left(\|\nabla \mathcal{F}\left(\bar{\mathbf{x}}_k\right)\|^2 + \|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_k^i)\|^2 - \|\nabla\mathcal{F}(\bar{\mathbf{x}}_k) - \frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_k^i)\|^2\right) \\ &\overset{a}{\geq} \frac{1}{2}\left(\|\nabla\mathcal{F}(\bar{\mathbf{x}}_k)\|^2 + \|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_k^i)\|^2 - L^2\frac{1}{N}\sum_{i=1}^{N}\|\bar{\mathbf{x}}_k - \mathbf{x}_k^i\|^2\right)\end{aligned} \tag{41}$$

(a) holds because $\|\nabla\mathcal{F}(\bar{\mathbf{x}}_k) - \frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_k^i)\|^2 = \|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\bar{\mathbf{x}}_k) - \frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_k^i)\|^2 \leq \frac{1}{N}\sum_{i=1}^{N}\|\nabla f_i(\bar{\mathbf{x}}_k) - \nabla f_i(\mathbf{x}_k^i)\|^2 \leq \frac{1}{N}\sum_{i=1}^{N}L^2\|\bar{\mathbf{x}}_k - \mathbf{x}_k^i\|^2$.

Substituting (41) into (40), we have

$$\mathbb{E}\Big[\langle \nabla \mathcal{F}\left(\bar{\mathbf{x}}_k\right), \frac{1}{N}\sum_{i=1}^{N}\mathbf{g}_k^i\rangle\Big] \geq \frac{1}{2}\left(\mathbb{E}[\|\nabla\mathcal{F}(\bar{\mathbf{x}}_k)\|^2] + \mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_k^i)\|^2] - L^2\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[\|\bar{\mathbf{x}}_k - \mathbf{x}_k^i\|^2]\right) \tag{42}$$

Substituting (39), (42) into (38), we have

$$\begin{aligned}-\frac{\eta}{1-\beta}\mathbb{E}[\langle \nabla\mathcal{F}(\bar{\mathbf{x}}_k), \frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{g}}_k^i\rangle] \leq &\Big(\frac{(1-\beta)}{2\beta L} - \frac{\eta}{2(1-\beta)}\Big)\mathbb{E}[\|\nabla\mathcal{F}(\bar{\mathbf{x}}_k)\|^2] + \frac{\eta^2\beta L}{2(1-\beta)^3}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}(\tilde{\mathbf{g}}_k^i - \mathbf{g}_k^i)\|^2] \\ &- \frac{\eta}{2(1-\beta)}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_k^i)\|^2] + \frac{\eta L^2}{2(1-\beta)}\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[\|\bar{\mathbf{x}}_k - \mathbf{x}_k^i\|^2]\end{aligned} \tag{43}$$

Now, finally substituting (43), (37) into (36), we have

$$\mathbb{E}[\langle \nabla \mathcal{F}(\bar{\mathbf{z}}_k), \bar{\mathbf{z}}_{k+1} - \bar{\mathbf{z}}_k \rangle] \leq \frac{(1-\beta)L}{2\beta}\mathbb{E}[\|\bar{\mathbf{z}}_k - \bar{\mathbf{x}}_k\|^2] + \frac{\beta L \eta^2}{2(1-\beta)^3}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}(\tilde{\mathbf{g}}_k^i)\|^2] + \left(\frac{(1-\beta)}{2\beta L} - \frac{\eta}{2(1-\beta)}\right)$$
$$\mathbb{E}[\|\nabla \mathcal{F}(\bar{\mathbf{x}}_k)\|^2] - \frac{\eta}{2(1-\beta)}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_k^i)\|^2] + \frac{\eta^2 \beta L}{2(1-\beta)^3}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}(\tilde{\mathbf{g}}_k^i - \mathbf{g}_k^i)\|^2] + \frac{\eta L^2}{2(1-\beta)}\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[\|\bar{\mathbf{x}}_k - \mathbf{x}_k^i\|^2] \tag{44}$$

Equation. 15 states that:

$$\mathbb{E}[\|\bar{\mathbf{z}}_{k+1} - \bar{\mathbf{z}}_k\|^2] = \frac{\eta^2}{(1-\beta)^2}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{g}}_k^i\|^2]. \tag{45}$$

Substituting (44), (45) in (35):

$$\mathbb{E}[\mathcal{F}(\bar{\mathbf{z}}_{k+1})] \leq \mathbb{E}[\mathcal{F}(\bar{\mathbf{z}}_k)] + \frac{(1-\beta)L}{2\beta}\mathbb{E}[\|\bar{\mathbf{z}}_k - \bar{\mathbf{x}}_k\|^2] + \frac{\beta L \eta^2}{2(1-\beta)^3}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}(\tilde{\mathbf{g}}_k^i)\|^2] + \left(\frac{(1-\beta)}{2\beta L} - \frac{\eta}{2(1-\beta)}\right)$$
$$\mathbb{E}[\|\nabla \mathcal{F}(\bar{\mathbf{x}}_k)\|^2] - \frac{\eta}{2(1-\beta)}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_k^i)\|^2] + \frac{\eta^2 \beta L}{2(1-\beta)^3}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}(\tilde{\mathbf{g}}_k^i - \mathbf{g}_k^i)\|^2] +$$
$$\frac{\eta L^2}{2(1-\beta)}\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[\|\bar{\mathbf{x}}_k - \mathbf{x}_k^i\|^2] + \frac{\eta^2}{(1-\beta)^2}\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{g}}_k^i\|^2].$$

We find the bound for $\mathbb{E}[\|\nabla \mathcal{F}(\bar{\mathbf{x}}_k)\|^2]$ by rearranging the terms and dividing by $C_1 = \frac{\eta}{2(1-\beta)} - \frac{(1-\beta)}{2\beta L}$:

$$\mathbb{E}[\|\nabla \mathcal{F}(\bar{\mathbf{x}}_k)\|^2] \leq \frac{1}{C_1}\left(\mathbb{E}[\mathcal{F}(\bar{\mathbf{z}}_k)] - \mathbb{E}[\mathcal{F}(\bar{\mathbf{z}}_{k+1})]\right) + C_2\,\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}(\tilde{\mathbf{g}}_k^i)\|^2] + C_3\,\mathbb{E}[\|\bar{\mathbf{z}}_k - \bar{\mathbf{x}}_k\|^2]$$
$$- C_6\,\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_k^i)\|^2] + C_4\,\mathbb{E}[\|\frac{1}{N}\sum_{i=1}^{N}(\tilde{\mathbf{g}}_k^i - \mathbf{g}_k^i)\|^2] + \frac{C_5}{N}\sum_{i=1}^{N}\mathbb{E}[\|\bar{\mathbf{x}}_k - \mathbf{x}_k^i\|^2]$$

Where $C_2 = \left(\frac{\beta L \eta^2}{2(1-\beta)^3} + \frac{\eta^2 L}{(1-\beta)^2}\right)/C_1$, $C_3 = \frac{(1-\beta)L}{2\beta}/C_1$, $C_4 = \frac{\eta^2 \beta L}{2(1-\beta)^3}/C_1$, $C_5 = \frac{\eta L^2}{2(1-\beta)}/C_1$, $C_6 = \frac{\eta}{2(1-\beta)}/C_1$.

Summing over $k \in \{0, 1, \ldots, K-1\}$:

$$\sum_{k=0}^{K-1}\mathbb{E}\left[\|\nabla \mathcal{F}(\bar{\mathbf{x}}_k)\|^2\right] \leq \frac{1}{C_1}\left(\mathbb{E}\left[\mathcal{F}(\bar{\mathbf{z}}_0)\right] - \mathbb{E}\left[\mathcal{F}(\bar{\mathbf{z}}_k)\right]\right) - C_6\sum_{k=0}^{K-1}\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i\left(\mathbf{x}_k^i\right)\right\|^2\right] + C_2\sum_{k=0}^{K-1}\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\tilde{\mathbf{g}}_k^i\right\|^2\right]$$
$$+ C_3\sum_{k=0}^{K-1}\mathbb{E}\left[\|\bar{\mathbf{z}}_k - \bar{\mathbf{x}}_k\|^2\right] + C_4\sum_{k=0}^{K-1}\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\left(\tilde{\mathbf{g}}_k^i - \mathbf{g}_k^i\right)\right\|^2\right] + C_5\sum_{k=0}^{K-1}\frac{1}{N}\sum_{l=1}^{N}\mathbb{E}\left[\|\bar{\mathbf{x}}_k - \mathbf{x}_k^i\|^2\right]$$

Substituting Lemma 1, Lemma 2, Lemma 3 and Equation. 16 into the above equation, we have:

$$\sum_{k=0}^{K-1}\mathbb{E}\left[\|\nabla\mathcal{F}\left(\bar{\mathbf{x}}_k\right)\|^2\right]\leq\frac{1}{C_1}\left(\mathbb{E}\left[\mathcal{F}\left(\bar{\mathbf{z}}_0\right)\right]-\mathbb{E}\left[\mathcal{F}\left(\bar{\mathbf{z}}_k\right)\right]\right)-\left(C_6-2C_2-2C_3\frac{\eta^2\beta^2}{(1-\beta)^4}-C_5\frac{6\eta^2}{(1-\beta)^2(1-\sqrt{\rho})}\right)$$
$$\sum_{k=0}^{K-1}\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i\left(\mathbf{x}_k^i\right)\right\|^2\right]+\left(C_2+C_3\frac{\eta^2\beta^2}{(1-\beta)^4}\right)\left(\frac{10\sigma^2}{N}+8\zeta^2\right)K+4C_4\left(\frac{\sigma^2}{N}+\zeta^2\right)K$$
$$+C_5\frac{14\eta^2(\sigma^2+\zeta^2)}{(1-\beta)^2(1-\sqrt{\rho})^2}K$$

Dividing both sides by K and considering the fact that $\bar{\mathbf{z}}_0=\bar{\mathbf{x}}_0$ and $\left(C_6-2C_2-2C_3\frac{\eta^2\beta^2}{(1-\beta)^4}-C_5\frac{6\eta^2}{(1-\beta)^2(1-\sqrt{\rho})}\right)\geq 0$:

$$\begin{aligned}\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\left[\|\nabla\mathcal{F}\left(\bar{\mathbf{x}}_k\right)\|^2\right]&\leq\frac{1}{C_1K}\left(\mathcal{F}\left(\bar{\mathbf{x}}_0\right)-\mathcal{F}^\star\right)+\left(C_2+C_3\frac{\eta^2\beta^2}{(1-\beta)^4}\right)\left(\frac{10\sigma^2}{N}+8\zeta^2\right)\\&+4C_4\left(\frac{\sigma^2}{N}+\zeta^2\right)+C_5\frac{14\eta^2(\sigma^2+\zeta^2)}{(1-\beta)^2(1-\sqrt{\rho})^2}\\&\leq\frac{1}{C_1K}\left(\mathcal{F}\left(\bar{\mathbf{x}}_0\right)-\mathcal{F}^\star\right)+\left(10C_2+10C_3\frac{\eta^2\beta^2}{(1-\beta)^4}+4C_4\right)\left(\frac{\sigma^2}{N}+\zeta^2\right)\\&+C_5\frac{14\eta^2(\sigma^2+\zeta^2)}{(1-\beta)^2(1-\sqrt{\rho})^2}\end{aligned}$$

Therefore we arrive at the bound given by the theorem. 1:

$$\begin{aligned}\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[||\nabla\mathcal{F}(\bar{x}_k)||^2]&\leq\frac{1}{C_1K}(\mathcal{F}(\bar{x}^0)-\mathcal{F}^*)+\left(10C_2+10C_3\frac{\eta^2\beta^2}{(1-\beta)^4}+4C_4\right)(\frac{\sigma^2}{N}+\zeta^2)+\\&C_5\frac{14\eta^2(\sigma^2+\zeta^2)}{(1-\beta)^2(1-\sqrt{\rho})^2}\end{aligned}\tag{46}$$

A.2.3 Discussion on the Step Size

In the proof of Theorem 1, we assumed the following $C_6-2C_2-2C_3\frac{\eta^2\beta^2}{(1-\beta)^4}-C_5\frac{6\eta^2}{(1-\beta)^2(1-\sqrt{\rho})}\geq 0$. The above equation is true under the following conditions:

$$\begin{aligned}&(i)\ \ C_1>0\\&(ii)\ \ 1-\frac{4L}{(1-\beta)^2}\eta-\frac{6L^2}{(1-\beta)^2(1-\sqrt{\rho})}\eta^2\geq 0\end{aligned}$$

Solving the first inequality gives us $\frac{(1-\beta)^2}{\beta L}<\eta$.

Now, solving the second inequality, combining the fact that $\eta>0$, we have then the specific form of η^*

$$\eta^*=\frac{\sqrt{4(1-\sqrt{\rho})^2+6(1-\sqrt{\rho})(1-\beta)^2}-2(1-\sqrt{\rho})}{6L}.$$

Therefore, the step size η is defined as

$$\frac{(1-\beta)^2}{\beta L}<\eta\leq\frac{\sqrt{4(1-\sqrt{\rho})^2+6(1-\sqrt{\rho})(1-\beta)^2}-2(1-\sqrt{\rho})}{6L}$$

A.2.4 Proof for Corollary 1

We assume that the step size η is $\mathcal{O}(\frac{\sqrt{N}}{\sqrt{K}})$ and ζ^2 is $\mathcal{O}(\frac{1}{\sqrt{K}})$. Given this assumption, we have the following

$$C_1 = \mathcal{O}(\frac{\sqrt{N}}{\sqrt{K}}), C_2 = \mathcal{O}(\frac{\sqrt{N}}{\sqrt{K}}), C_3 = \mathcal{O}(\frac{\sqrt{K}}{\sqrt{N}}), C_4 = \mathcal{O}(\frac{\sqrt{N}}{\sqrt{K}}), C_5 = \mathcal{O}(1),$$

Now we proceed to find the order of each term in Equation. 46. To do that we first point out that

$$\frac{\mathcal{F}(\bar{\mathbf{x}}_0) - \mathcal{F}^*}{C_1 K} = \mathcal{O}\Big(\frac{1}{\sqrt{NK}}\Big).$$

For the remaining terms we have,

$$\Big(10C_2 + 10C_3\frac{\eta^2\beta^2}{(1-\beta)^4} + 4C_4\Big)\frac{\sigma^2}{N} = \mathcal{O}\Big(\frac{1}{\sqrt{NK}}\Big), \quad \Big(10C_2 + 10C_3\frac{\eta^2\beta^2}{(1-\beta)^4} + 4C_4\Big)\zeta^2 = \mathcal{O}\Big(\frac{\sqrt{N}}{K}\Big)$$

$$C_5\frac{14\eta^2\sigma^2}{(1-\beta)^2(1-\sqrt{\rho})^2} = \mathcal{O}\Big(\frac{N}{K}\Big), \quad C_5\frac{14\eta^2\zeta^2}{(1-\beta)^2(1-\sqrt{\rho})^2} = \mathcal{O}\Big(\frac{N}{K^{1.5}}\Big)$$

Therefore, by omitting the constant N in this context, there exists a constant $C > 0$ such that the overall convergence rate is as follows:

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\left[\|\nabla\mathcal{F}(\bar{\mathbf{x}}_k)\|^2\right] \leq C\left(\frac{1}{\sqrt{NK}} + \frac{1}{K} + \frac{1}{K^{1.5}}\right), \tag{47}$$

which suggests when K is sufficiently large, *NGC* enables the convergence rate of $\mathcal{O}(\frac{1}{\sqrt{NK}})$.

A.3 Decentralized Learning Setup

The traditional decentralized learning algorithm (d-psgd) is described as Algorithm. 2. For the decentralized setup, we use an undirected ring and undirected torus graph topologies with a uniform mixing matrix. The undirected ring topology for any graph size has 3 peers per agent including itself and each edge has a weight of $\frac{1}{3}$. The undirected torus topology with 10 agents has 4 peers per agent including itself and each edge has a weight of $\frac{1}{4}$. The undirected torus topology 20 agents have 5 peers per agent including itself and each edge has a weight of $\frac{1}{5}$. Finally, Figure 5 depicts the overview of the proposed *NGC* algorithm. Note that we do not have step 4 of the Figure 5 in NGC_{mv} algorithm.

Algorithm 2 Decentralized Peer-to-Peer Training (*D-PSGD* with momentum)

Input: Each agent $i \in [1, N]$ initializes model weights $x^i_{(0)}$, step size η, averaging rate γ, mixing matrix $W = [w_{ij}]_{i,j\in[1,N]}$, and I_{ij} are elements of $N \times N$ identity matrix.

Each agent simultaneously implements the TRAIN( ) procedure
1. **procedure** TRAIN( )
2. **for** k=0, 1, ..., $K-1$ **do**
3. $d^i_k \sim D^i$ // sample data from training dataset.
4. $g^i_k = \nabla_x f_i(d^i_k; x^k_i)$ // compute the local gradients
5. $v^i_k = \beta v^i_{(k-1)} - \eta g^i_k$ // momentum step
6. $\widetilde{x}^i_k = x^i_k + v^i_k$ // update the model
7. SENDRECEIVE($\widetilde{x}^i_k$) // share model parameters with neighbors $N(i)$.
8. $x^i_{(k+1)} = \widetilde{x}^i_k + \gamma\sum_{j\in\mathcal{N}_i}(w_{ij} - I_{ij}) * \widetilde{x}^j_k$ // gossip averaging step
9. **end**
10. **return**

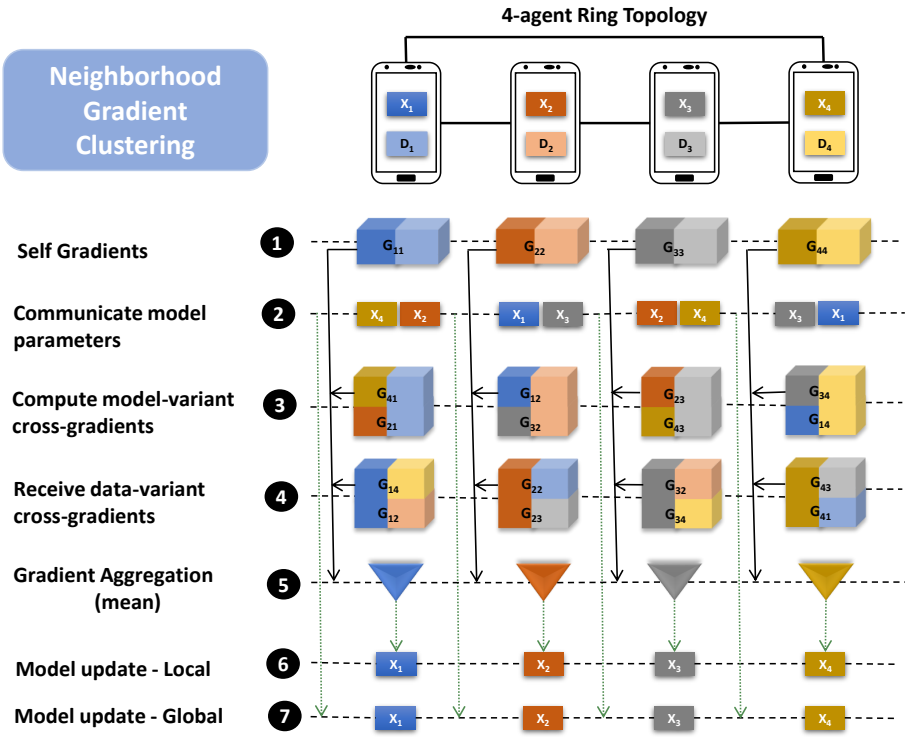


Figure 5: *NGC* algorithm overview. In the proposed algorithm, (1) each agent computes self-gradients of model parameters on its own data set; (2) each agent sends its model parameters to its neighbors; (3) each agent computes the model-variant cross-gradients of its neighbors' models on its own data set; (4) each agent receives the data-variant cross-gradients from its neighbors; (5) update the local gradient using the mean of self-gradients and cross-gradients ; (6) update the model parameter using local SGD step with updated gradients; (7) and update the model parameter using global gossip averaging step.

A.4 CompNGC Algorithm

The section presents the pseudocode for CompNGC (compressed version of NGC_{dv}) in Algorithm 3. The NGC_{dv} algorithm at every iteration involves two rounds of communication with the neighbors: 1) communicate the model parameters, and 2) communicate the cross-gradients. This communication overhead can be a bottleneck in a resource-constrained environment. Hence we propose a compressed version of NGC_{dv} using Error Feedback SGD (EF-SGD) Karimireddy et al. (2019) to compress gradients. We compress the error-compensated self-gradients and cross-gradients from 32 bits (floating point precision of arithmetic operations) to 1 bit by using scaled signed gradients. The error between the compressed and non-compressed gradient of the current iteration is added as feedback to the gradients in the next iteration before compression.

A.5 Datasets

In this section, we give a brief description of the datasets used in our experiments. We use a diverse set of datasets each originating from a different distribution of images to show the generalizability of the proposed techniques.

CIFAR-10: CIFAR-10 Krizhevsky et al. (2014) is an image classification dataset with 10 classes. The image samples are colored (3 input channels) and have a resolution of 32×32. There are $50,000$ training samples with 5000 samples per class and $10,000$ test samples with 1000 samples per class.

Algorithm 3 Compressed Neighborhood Gradient Clustering (*CompNGC*)

Input: Each agent $i \in [1, N]$ initializes model weights $x^i_{(0)}$, step size η, averaging rate γ, dimension of the gradient d, mixing matrix $W = [w_{ij}]_{i,j\in[1,N]}$, *NGC* mixing weight α, and I_{ij} are elements of $N \times N$ identity matrix.

Each agent simultaneously implements the TRAIN( ) procedure

procedure TRAIN( )
  for k=$0, 1, \ldots, K-1$ **do**
    $d^i_k \sim D^i$ // sample data from training dataset
    $g^{ii}_k = \nabla_x f_i(d^i_k; x^i_k)$ // compute the local self-gradients
    $p^{ii}_k = g^{ii}_k + e^{ii}_k$ // error compensation for self-gradients
    $\delta^{ii}_k = (||p^{ii}_k||_1/d)sgn(p^{ii})$ // compress the compensated self-gradients
    $e^{ii}_{k+1} = p^{ii}_k - \delta^{ii}_k$ // update the error variable
    SENDRECEIVE(x^i_k) // share model parameters with neighbors $N(i)$
    for each neighbor $j \in \{N(i) - i\}$ **do**
      $g^{ji}_k = \nabla_x f_i(d^i_k; x^j_k)$ // compute neighbors' cross-gradients
      $p^{ji}_k = g^{ji}_k + e^{ji}_k$ // error compensation for cross-gradients
      $\delta^{ji}_k = (||p^{ji}_k||_1/d)sgn(p^{ji})$ // compress the compensated cross-gradients
      $e^{ji}_{k+1} = p^{ji}_k - \delta^{ji}_k$ // update the error variable
      if $\alpha \neq 0$ **do**
        SENDRECEIVE(δ^{ji}_k) // share compressed cross-gradients between i and j
      end
    end
    $\widetilde{g}^i_k = (1-\alpha) * \sum_{j\in\mathcal{N}_i} w_{ji} * \delta^{ji}_k + \alpha * \sum_{j\in\mathcal{N}_i} w_{ij} * \delta^{ij}_k$ // modify local gradients
    $v^i_k = \beta v^i_{(k-1)} - \eta \widetilde{g}^i_k$ // momentum step
    $\widetilde{x}^i_k = x^i_k + v^i_k$ // update the model
    $x^i_{(k+1)} = \widetilde{x}^i_k + \gamma \sum_{j\in\mathcal{N}_i} (w_{ij} - I_{ij}) * x^j_k$ // gossip averaging step
  end
return

CIFAR-100: CIFAR-100 Krizhevsky et al. (2014) is an image classification dataset with 100 classes. The image samples are colored (3 input channels) and have a resolution of 32×32. There are $50,000$ training samples with 500 samples per class and $10,000$ test samples with 100 samples per class. CIFAR-100 classification is a harder task compared to CIFAR-10 as it has 100 classes with very less samples per class to learn from.

Fashion MNIST: Fashion MNIST Xiao et al. (2017) is an image classification dataset with 10 classes. The image samples are in greyscale (1 input channel) and have a resolution of 28×28. There are $60,000$ training samples with 6000 samples per class and $10,000$ test samples with 1000 samples per class.

Imagenette: Imagenette Husain (2018) is a 10-class subset of the ImageNet dataset. The image samples are colored (3 input channels) and have a resolution of 224×224. There are 9469 training samples with roughly 950 samples per class and 3925 test samples. We conduct our experiments on two different resolutions of the Imagenette dataset – a) a resized low resolution of 32×32 and, b) a full resolution of 224×224. The Imagenette experimental results reported in Table. 3 use the low-resolution images whereas experimental results in Table. 4 use the full resolution images.

AGNews: We use AGNews Zhang et al. (2015) dataset for Natural Language Processing (NLP) task. This is a text classification dataset where the given text news is classified into 4 classes, namely "World", "Sport", "Business" and "Sci/Tech". The dataset has a total of 120000 and 7600 samples for training and testing respectively, which are equally distributed across each class.

A.6 Network Architecture

We replace ReLU+BatchNorm layers of all the model architectures with EvoNorm-S0 Liu et al. (2020) as it was shown to be better suited for decentralized learning on non-IID data Lin et al. (2021).

5 layer CNN: The 5-layer CNN consists of 4 convolutional with EvoNorm-S0 Liu et al. (2020) as an activation-normalization layer, 3 max-pooling layers, and one linear layer. In particular, it has 2 convolutional layers with 32 filters, a max pooling layer, then 2 more convolutional layers with 64 filters each followed by another max pooling layer and a dense layer with 512 units. It has a total of $76K$ trainable parameters.

VGG-11: We modify the standard VGG-11 Simonyan & Zisserman (2014) architecture by reducing the number of filters in each convolutional layer by $4\times$ and using only one dense layer with 128 units. Each convolutional layer is followed by EvoNorm-S0 as the activation-normalization layer and it has $0.58M$ trainable parameters.

ResNet-20: For ResNet-20 He et al. (2016), we use the standard architecture with $0.27M$ trainable parameters except that BatchNorm+ReLU layers are replaced by EvoNorm-S0.

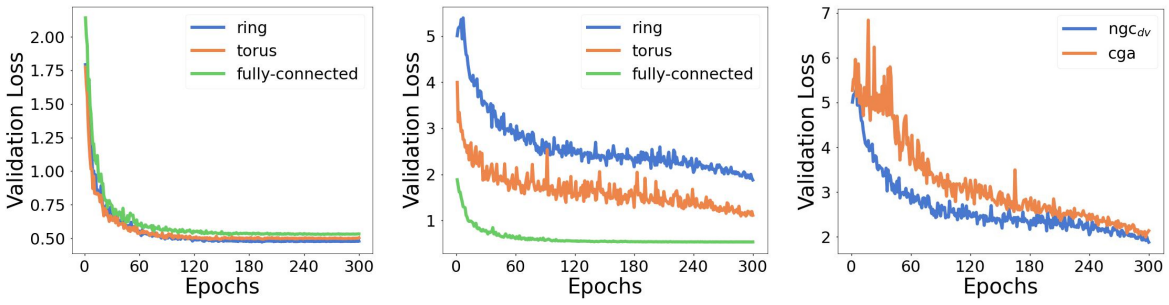
LeNet-5: For LeNet-5 LeCun et al. (1998), we use the standard architecture with $61,706$ trainable parameters.

MobileNet-V2: We use the the standard MobileNet-V2 Sandler et al. (2018) architecture used for CIFAR dataset with $2.3M$ parameters except that BatchNorm+ReLU layers are replaced by EvoNorm-S0.

ResNet-18: For ResNet-18 He et al. (2016), we use the standard architecture with $11M$ trainable parameters except that BatchNorm+ReLU layers are replaced by EvoNorm-S0.

BERT$_{\text{mini}}$: For BERT$_{\text{mini}}$ Devlin et al. (2018) we use the standard model from the paper. We restrict the sequence length of the model to 128. The model used in the work hence has $11.07M$ parameters.

DistilBERT$_{\text{base}}$: For DistilBERT$_{\text{base}}$ Sanh et al. (2019) we use the standard model from the paper. We restrict the sequence length of the model to 128. The model used in the work hence has $66.67M$ parameters.

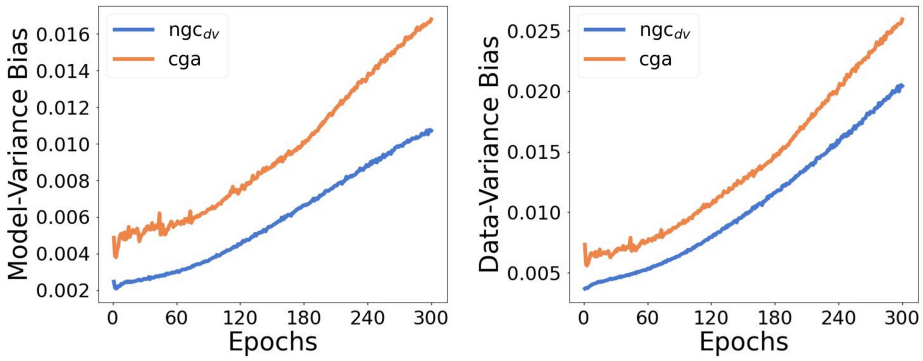


(a) NGC_{dv} on IID data (b) NGC_{dv} on non-IID data (c) Non-IID for ring topology

Figure 6: Validation loss for CIFAR-10 dataset trained on a 10 agents ring graph with 5-layer CNN.

A.7 Analysis for 10 Agents

We show the convergence characteristics of the proposed NGC_{dv} algorithm over IID and Non-IID data sampled from the CIFAR-10 dataset in Figure. 6(a), and 6(b) respectively. For Non-IID distribution, we observe that there is a slight difference in convergence rate (as expected) with a slower rate for sparser topology (undirected ring graph) compared to its denser counterpart (fully connected graph). Figure. 6(c) shows the comparison of the convergence characteristics of the NGC_{dv} technique with the current state-of-the-art CGA algorithm. We observe that NGC_{dv} has lower validation loss than CGA for the same decentralized setup indicating its superior performance over CGA. We also plot the model variance and data variance bias terms for both NGC_{dv} and CGA techniques as shown in Figure. 7(a), and 7(b) respectively. We observe that both model variance and data variance bias for NGC_{dv} are significantly lower than CGA.



(a) Average L1 norm of ϵ.

(b) Average L1 norm of ω.

Figure 7: Average L1 norm of model variance bias and data variance bias for 10 agents trained on CIFAR-10 dataset with 5 layer CNN architecture over an undirected ring topology.

A.8 Resource Comparison

The communication cost, memory overhead, and compute overhead for various decentralized algorithms are shown in Table. 5. The D-PSGD algorithm requires each agent to communicate model parameters of size m_s with all the $\mathcal{N}_i$ neighbors for the gossip averaging step and hence has a communication cost of $\mathcal{O}(\mathcal{N}_i m_s)$. In the case of NGC_{dv} and CGA, there is an additional communication round for sharing data-variant cross gradients apart from sharing model parameters for the gossip averaging step. So, both these techniques incur a communication cost of $\mathcal{O}(2\mathcal{N}_i m_s)$ and therefore an overhead of $\mathcal{O}(\mathcal{N}_i m_s)$ compared to D-PSGD. *CompNGC* compresses the additional round of communication involved with NGC_{dv} from b bits to 1 bit. This reduces the communication overhead from $\mathcal{O}(\mathcal{N}_i m_s)$ to $\mathcal{O}(\frac{\mathcal{N}_i m_s}{b})$.

Table 7: Memory overheads for various methods trained on different model architectures with CIFAR-10 dataset over undirected ring topology with 2 neighbors per agent.

Architecture	CGA (MB)	NGC_{dv} (MB)	CompCGA (MB)	*CompNGC* (MB)
5 layer CNN	0.58	0	0.58	0.60
VGG-11	4.42	0	4.42	4.56
ResNet-20	2.28	0	2.28	2.15

CGA algorithm stores all the received data-variant cross-gradients in the form of a matrix for the quadratic projection step. Hence, CGA has a memory overhead of $\mathcal{O}(\mathcal{N}_i m_s)$ compared to D-PSGD. NGC_{dv} does not require any additional memory as it averages the received data-variant cross-gradients into the self-gradient buffer. The compressed version of NGC_{dv} requires an additional memory of $\mathcal{O}(\mathcal{N}_i m_s)$ to store the error variables e_{ji} (refer Algorith. 3). CompCGA also needs to store error variables along with the projection matrix of compressed gradients. Therefore, CompCGA has a memory overhead of $\mathcal{O}(\mathcal{N}_i m_s + \frac{\mathcal{N}_i m_s}{b})$. Note that memory overhead depends on the type of graph topology and model architecture but not on the size of the graph. The memory overhead for different model architectures trained on undirected ring topology is shown in Table. 7

The computation of the cross-gradients (in both CGA and NGC algorithms) requires $\mathcal{N}_i$ forward and backward passes through the deep learning model at each agent. This is reflected as $\mathcal{O}(3\mathcal{N}_i FP)$ in the compute overhead in Table. 5. We assume that the compute effort required for the backward pass is twice that of the forward pass. CGA algorithm involves quadratic programming projection step Goldfarb & Idnani (1983) to update the local gradients. Quadratic programming solver (quadprog) uses Goldfarb/Idnani dual algorithm. CGA uses quadratic programming to solve the following (Equation 48 -see Equation 5a in Esfandiari et al.

(2021)) optimization problem in an iterative manner:

$$\begin{aligned} \min_u \quad & \frac{1}{2} u^T G G^T u + g^T G^T u \\ \text{s.t.} \quad & u \geq 0 \end{aligned} \tag{48}$$

where G is the matrix containing cross-gradients, g is the self-gradient, and the optimal gradient direction g^* in terms of the optimal solution of the above equation u^* is $g^* = G^T u^* + g$. The above optimization takes multiple iterations, resulting in compute and time complexity of polynomial(degree≥ 2) order. In contrast, NGC involves a simple averaging step that requires $O(\mathcal{N}_i m_s)$ addition operations.

A.9 Communication Cost

In this section we present the communication cost per agent in terms of Gigabytes of data transferred during the entire training process (refer Tables. 8, 9, 11, 10). The D-PSGD and NGC_{mv} have the lowest communication cost (1×). We emphasize that NGC_{mv} outperforms D-PSGD in decentralized learning over label-wise non-IID data for the same communication cost. NGC_{dv} and CGA have 2× communication overhead compared to D-PSGD where as *CompNGC* and CompCGA have 1.03× communication overhead compared to D-PSGD. The compressed versions of NGC_{dv} and CGA compress the second round of cross-gradient communication to 1 bit. We assume the full-precision cross-gradients to be of 32-bit precision and hence the *CompNGC* reduces the communication cost by 32× compared to NGC_{dv}.

Table 8: Communication costs per agent in GBs for experiments in Table 2

Method	Agents	5layer CNN Ring	5layer CNN Torus	VGG-11 Ring	ResNet-20 Ring
D-PSGD	5	17.75	-	270.64	127.19
and	10	8.92	13.38	135.86	63.84
NGC_{mv}	20	4.50	6.65	68.48	32.18
CGA	5	35.48	-	541.05	254.27
and	10	17.81	26.72	271.50	127.59
NGC_{dv}	20	8.98	17.95	136.72	64.25
CompCGA	5	18.31	-	279.09	131.16
and	10	9.20	13.79	140.10	65.84
CompNGC	20	4.64	9.28	70.61	33.18

Table 9: Communication costs per agent in GBs for experiments in Table 3

Method	Agents	Fashion MNIST (LeNet-5)	CIFAR-100 (ResNet-20)	Imagenette (MobileNet-V2)
D-PSGD and	5	17.25	103.74	103.12
NGC_{mv}	10	8.61	51.89	51.60
CGA and	5	34.50	207.47	206.23
NGC_{dv} (ours)	10	17.23	103.79	103.19
CompCGA and	5	17.79	106.98	106.34
CompNGC (ours)	10	8.88	53.52	53.21

A.10 Hyper-parameters

All the experiments were run for three randomly chosen seeds. We decay the step size by 10x after 50% and 75% of the training, unless mentioned otherwise.

Hyper-parameters for CIFAR-10 : All the experiments that involve 5layer CNN model (Table. 2) have stopping criteria set to 100 epochs. We decay the step size by 10× in multiple steps at 50th and 75th epoch.

Table 10: Communication costs per agent in GBs for experiments in Table 4 (right)

Method	Ring topology	chain topology
D-PSGD and NGC_{mv}	501.98	401.59
CGA and NGC_{dv}	1003.96	803.17
CompCGA and *CompNGC*	517.67	414.14

Table 11: Communication costs per agent in GBs for experiments in Table 4 (left)

Method	$\text{BERT}_{\text{mini}}$		$\text{DistilBERT}_{\text{base}}$
	Agents = 4	Agents = 8	Agents = 4
D-PSGD and NGC_{mv}	234.30	118.20	1410.39
CGA and NGC_{dv}	486.59	236.40	2820.77
CompCGA and *CompNGC*	241.6	121.89	1454.46

All the experiments for the CIFAR-10 dataset trained on VGG-11 and ResNet-20 architectures (Table. 2) have stopping criteria set to 200 epochs. We decay the step size by $10\times$ in multiple steps at 100th and 150th epoch. For all the experiments, we use a mini-batch size of 32 per agent. We have used a momentum of 0.9 for all CGA and NGC_{dv} experiments whereas D-PSGD and NGC_{mv} have no momentum. Table 12 presents the β, η, and γ corresponding to the momentum, step size, and gossip averaging rate.

Hyper-parameters used for Table. 3: All the experiments in Table. 3 have stopping criteria set to 100 epochs. We decay the step size by $10\times$ in multiple steps at 50th and 75th epoch. Table 13 presents the β, η, and γ corresponding to the ngc mixing weight, momentum, step size, and gossip averaging rate. For all the experiments related to Fashion MNIST and Imagenette (low resolution of (32×32)), we use a mini-batch size of 32 per agent. For all the experiments related to CIFAR-100, we use a mini-batch size of 20 per agent.

Hyper-parameters used for Table. 4 (right): All the experiments in Table. 4 have stopping criteria set to 100 epochs. We decay the step size by $10\times$ at $50^{th}, 75^{th}$ epoch. Table 14 presents the β, η, and γ corresponding to the *NGC* mixing weight, momentum, step size, and gossip averaging rate. For all the experiments, we use a mini-batch size of 32 per agent.

Hyper-parameters used for Table. 4 (left): All the experiments in Table. 4 (left) have stopping criteria set to 3 epochs. We decay the step size by $10\times$ at 2nd epoch. Table 15 presents the β, η, and γ corresponding to the ngc mixing weight, momentum, step size, and gossip averaging rate. For all the experiments, we use a mini-batch size of 32 per agent on the AGNews dataset.

Table 12: Hyper-parameters used for CIFAR-10 with non-IID data using various model architecture presented in Table 2

Method	Agents (n)	5 Layer CNN Ring (β, η, γ)	5 Layer CNN Torus (β, η, γ)	VGG-11 Ring (β, η, γ)	ResNet-20 Torus (β, η, γ)
D-PSGD and NGC_{mv}	5	$(0.0, 0.1, 1.0)$	–	$(0.0, 0.01, 1.0)$	$(0.0, 0.1, 1.0)$
	10	$(0.0, 0.1, 1.0)$	$(0.0, 0.1, 1.0)$	$(0.0, 0.01, 1.0)$	$(0.0, 0.1, 1.0)$
	20	$(0.0, 0.1, 1.0)$	$(0.0, 0.1, 1.0)$	$(0.0, 0.01, 1.0)$	$(0.0, 0.1, 1.0)$
CGA and NGC_{dv}	5	$(0.9, 0.01, 0.1)$	–	$(0.9, 0.1, 0.5)$	$(0.9, 0.1, 1.0)$
	10	$(0.9, 0.01, 0.5)$	$(0.9, 0.01, 0.1)$	$(0.9, 0.1, 0.5)$	$(0.9, 0.1, 1.0)$
	20	$(0.9, 0.01, 0.5)$	$(0.9, 0.01, 0.1)$	$(0.9, 0.1, 0.5)$	$(0.9, 0.1, 1.0)$
CompCGA and *CompNGC*	5	$(0.9, 0.01, 0.1)$	–	$(0.9, 0.01, 0.1)$	$(0.9, 0.01, 0.1)$
	10	$(0.9, 0.01, 0.5)$	$(0.9, 0.01, 0.1)$	$(0.9, 0.01, 0.1)$	$(0.9, 0.01, 0.1)$
	20	$(0.9, 0.01, 0.5)$	$(0.9, 0.01, 0.1)$	$(0.9, 0.01, 0.1)$	$(0.9, 0.01, 0.1)$

Table 13: Hyper-parameters used for Table. 3

Method	Agents (n)	Fashion MNIST (β, η, γ)	CIFAR-100 (β, η, γ)	Imagenette (β, η, γ)
D-PSGD and NGC_{mv}	5	$(0.0, 0.01, 1.0)$	$(0.0, 0.1, 1.0)$	$(0.0, 0.1, 1.0)$
	10	$(0.0, 0.01, 1.0)$	$(0.0, 0.1, 1.0)$	$(0.0, 0.1, 1.0)$
CGA and NGC_{dv}	5	$(0.9, 0.01, 1.0)$	$(0.9, 0.1, 1.0)$	$(0.0, 0.01, 0.5)$
	10	$(0.9, 0.01, 1.0)$	$(0.9, 0.1, 0.5)$	$(0.0, 0.01, 0.5)$
CompCGA and *CompNGC*	5	$(0.9, 0.01, 0.1)$	$(0.9, 0.01, 0.1)$	$(0.0, 0.01, 0.1)$
	10	$(0.9, 0.01, 0.1)$	$(0.9, 0.01, 0.1)$	$(0.0, 0.01, 0.5)$

Table 14: Hyper-parameters used for Table. 4 (right)

Method	Ring topology (β, η, γ)	Chain topology (β, η, γ)
D-PSGD	$(0.0, 0.01, 1.0)$	$(0.0, 0.01, 1.0)$
NGC_{mv} (ours)	$(0.0, 0.01, 1.0)$	$(0.0, 0.01, 1.0)$
CGA	$(0.9, 0.01, 0.5)$	$(0.9, 0.01, 0.1)$
NGC_{dv}	$(0.9, 0.01, 0.5)$	$(0.9, 0.01, 0.1)$
CompCGA	$(0.9, 0.01, 0.1)$	$(0.9, 0.01, 0.1)$
CompNGC	$(0.9, 0.01, 0.1)$	$(0.9, 0.01, 0.1)$

Table 15: Hyper-parameters used for Table. 4 (left)

Method	$\text{BERT}_{\text{mini}}$ Agents = 4 (β, η, γ)	$\text{BERT}_{\text{mini}}$ Agents = 8 (β, η, γ)	$\text{DistilBERT}_{\text{base}}$ Agents = 4 (β, η, γ)
D-PSGD	$(0.0, 0.01, 1.0)$	$(0.0, 0.01, 1.0)$	$(0.0, 0.01, 1.0)$
NGC_{mv} (ours)	$(0.0, 0.01, 1.0)$	$(0.0, 0.01, 1.0)$	$(0.0, 0.01, 1.0)$
CGA	$(0.9, 0.01, 0.5)$	$(0.9, 0.01, 0.5)$	$(0.9, 0.01, 0.5)$
NGC_{dv}	$(0.9, 0.01, 0.5)$	$(0.9, 0.01, 0.5)$	$(0.9, 0.01, 0.5)$
CompCGA	$(0.9, 0.01, 0.5)$	$(0.9, 0.01, 0.5)$	$(0.9, 0.01, 0.5)$
CompNGC	$(0.9, 0.01, 0.5)$	$(0.9, 0.01, 0.5)$	$(0.9, 0.01, 0.5)$