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

Sai Aparna Aketi∗ saketi@purdue.edu
Department of Electrical and Computer Engineering
Purdue University

Sangamesh Kodge∗ skodge@purdue.edu
Department of Electrical and Computer Engineering
Purdue University

Kaushik Roy kaushik@purdue.edu
Department of Electrical and Computer Engineering
Purdue University

Reviewed on OpenReview: https://openreview.net/forum?id=vkiKzK5G3e

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 Mean (NGM), 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. We theoretically analyze the convergence characteristics of NGM 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 2−20% 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) [Konečný et al., 2016] is a popular setting in the distributed machine learning

∗These authors contributed equally to this work
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. 

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 D² 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 Nadridze 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%. However, the improvement in accuracy is only 1 – 2% in case of highly skewed data distribution as shown in Aketi et al. (2022). Tracking mechanisms such as Gradient Tracking (Di Lorenzo & Scutari, 2016; Pu & Nedić, 2021) and Momentum Tracking (Takezawa et al., 2021) have been proposed to tackle non-IID data in decentralized settings at the cost of 2× communication overhead. 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 Mean (NGM) algorithm with two variants to handle non-IID data in peer-to-peer decentralized learning setups. 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 $NGM_{mv}$ where the communication budget is $1\times$, and with the data-variant cluster means in $NGM_{dv}$ 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 $NGM_{dv}$ 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 show that $NGM_{dv}$ can be combined with error feedback-based compression methods to compress the additional round of cross-gradient communication. Note that $NGM_{mv}$ has no communication overhead. We provide a detailed convergence analysis of the proposed $NGM_{dv}$ algorithm and show that the analysis can be trivially extended to $NGM_{mv}$. We validate the performance of the proposed algorithm on various datasets, model architectures, and graph topologies. We compare the proposed algorithm with several baselines such as D-PSGD (Lian et al., 2017), QG-DSGDm (Lin et al., 2021), Momentum Tracking (Takezawa et al., 2022), and CGA (Esfandiari et al., 2021). Our experiments show that $NGM$ can achieve superior performance on non-IID data compared to the current state-of-the-art approaches. We also report the order of communication, memory, and compute overheads required for various decentralized algorithms as compared to D-PSGD.

**Contributions:** In summary, we make the following contributions.

- We propose the Neighborhood Gradient Mean ($NGM$) algorithm with two variations for decentralized learning on non-IID data. $NGM_{mv}$ utilizes self-gradients and model-variant cross-gradients to improve the performance on non-IID data. $NGM_{dv}$ 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 $NGM$ is $O\left(\frac{1}{\sqrt{NK}}\right)$, which is consistent with the state-of-the-art decentralized learning algorithms.

- $NGM_{mv}$ outperforms the QG-DSGDm baseline by $2\ldots20\%$ without any communication overhead.

- $NGM_{dv}$ either remains competitive or outperforms by $0\ldots6\%$ 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):

$$\min_{x \in \mathbb{R}^d} \mathcal{F}(x) = \frac{1}{N} \sum_{i=1}^{N} f_i(x),$$

and $f_i(x) = \mathbb{E}_{d_i \in D_i}[F_i(x; d_i)] \; \forall i$

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 $N_i$ 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 Mean

We propose the Neighborhood Gradient Mean (NGM) algorithm and a compressed version of NGM 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 $D^i$.

$$g^{ii} = \nabla_x f_i(x^i; d^i)$$

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 $D^j$.

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

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 ($g^{ii}$). 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 $g^{ii}$ 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 NGM algorithm

The pseudo-code of the Neighborhood Gradient Mean (NGM) algorithm is shown in Algorithm. 1. The overview of the algorithm is illustrated in Figure. 5 of Appendix A.3.

The main contribution of the proposed NGM algorithm is the local gradient manipulation step (line 12 in Algorithm. 1). We first introduce a hyper-parameter $\alpha$ called NGM mixing weight which is either set to 0 or 1. In the $k^{th}$ iteration of NGM, 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 $\alpha$ 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 $\alpha$ 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 $\{g^{ii} \forall j \in N_i\}$ that includes self-gradients and model-variant cross-gradients, and b) Data-variant cluster $\{g^{ij} \forall j \in N_i\}$ 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 $\alpha$ value as shown in Equation. 4 which assumes a uniform mixing matrix ($w_{ij} = 1/m; m = |N_i|$).
Algorithm 1 Neighborhood Gradient Mean (NGM)

Input: Each agent \( i \in [1, N] \) initializes model weights \( x_i^{(0)} \), step size \( \eta \), momentum coefficient \( \beta \), averaging rate \( \gamma \), mixing matrix \( W = [w_{ij}]_{i,j \in [1, N]} \), NGM 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 \text{Train( )} procedure
1. \textbf{procedure} \text{Train( )}
2. \hspace{1em} for \( k = 0, 1, \ldots, K - 1 \) do
3. \hspace{2em} \( d_k^i \sim D^i \)
4. \hspace{2em} \( g_k^i = \nabla_x f_i(d_k^i; x_k^i) \)
5. \hspace{2em} \text{SendReceive}(x_k^i)
6. \hspace{2em} for each neighbor \( j \in \{\mathcal{N}_i - i\} \) do
7. \hspace{3em} \( g_k^{ji} = \nabla_x f_i(d_k^j; x_k^j) \)
8. \hspace{3em} if \( \alpha \neq 0 \): \text{SendReceive}(g_k^{ji})
9. \hspace{2em} end
10. \hspace{2em} \( \bar{g}_k = \sum_{j \in \mathcal{N}_i} [(1 - \alpha)w_{ji}g_k^{ji} + \alpha w_{ij}g_k^{ij}] \)
11. \hspace{2em} \( v_k^i = \beta v_{(k-1)}^i - \eta \bar{g}_k^i \)
12. \hspace{2em} \( \bar{x}_k^i = x_k^i + v_k^i \)
13. \hspace{2em} \( x_{k+1}^i = \bar{x}_k^i + \gamma \sum_{j \in \mathcal{N}_i}(w_{ij} - I_{ij})x_k^j \)
14. \hspace{2em} end
15. \hspace{2em} return
16. \hspace{1em} \textbf{end}

When \( \alpha \) is set to zero, the proposed algorithm modifies the local gradient with model-variant cluster mean and this is referred to as \textit{NGM}_{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 \textit{NGM}_{dv}. Setting \( \alpha \) 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).

\[
\bar{g}_k = (1 - \alpha) \left[ \frac{1}{m} \sum_{j \in \mathcal{N}_i} g_k^{ji} \right] + \alpha \left[ \frac{1}{m} \sum_{j \in \mathcal{N}_i} g_k^{ij} \right]
\]

(a) Model-variant cluster mean \hspace{2em} (b) Data-variant cluster mean

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 \( \epsilon \) for \textit{NGM}$_{mv}$ and the bias term \( \omega \) for \textit{NGM}$_{dv}$ as shown in Equation.

\[
\begin{align*}
\bar{g}_k^i &= g_k^i + (1 - \alpha) \left[ \frac{1}{m} \sum_{j \in \mathcal{N}_i} (g_k^{ji} - g_k^{ij}) \right] + \alpha \left[ \frac{1}{m} \sum_{j \in \mathcal{N}_i} (g_k^{ij} - g_k^{ii}) \right] \\
\epsilon_k^i &= \frac{1}{m} \sum_{j \in \mathcal{N}_i} \left( \nabla_x F(d_k^j; x_k^j) - \nabla_x F(d_k^i; x_k^i) \right) \\
\omega_k^i &= \frac{1}{m} \sum_{j \in \mathcal{N}_i} \left( \nabla_x F(d_k^j; x_k^j) - \nabla_x F(d_k^i; x_k^i) \right)
\end{align*}
\]
The bias term $\epsilon$ 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 $\omega$ 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.

4 Convergence Analysis of NGM

In this section, we provide a convergence analysis for NGM. We assume that the following statements hold:

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

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

$$||\nabla f_i(x) - \nabla F(x)||^2 \leq \zeta^2 \quad \forall i \in [1, N]$$

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

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

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

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

where $\lambda_i(W)$ is the $i$th largest eigenvalue of $W$ and $\rho$ 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 NGM gradient update $\tilde{g}^i$. For all $i$, we have:

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

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 $\zeta^2$ determines the heterogeneity in the data distribution i.e., the non-IIDness.

The intra-gradient variation bounded by $\sigma^2$ determines the stochastic noise across the mini-batches within an agent. Intuitively, the distance between the self-gradients $g^i$ and the NGM$_{dv}$ gradient update $\tilde{g}^i$ increases with an increase in the degree of heterogeneity in data distribution which is expected. Note that the convergence rate for NGM$_{mv}$ remains the same as CGA where $E[||\tilde{g}^i - g^i||^2]$ is bounded by a positive constant which is of the order of $\frac{1}{K}$.

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

**Theorem 1.** (Convergence of NGM$_{dv}$ algorithm) Given assumptions 1-3 and lemma 1, let step size $\eta$ satisfy the following conditions:

1. $\eta \leq \sqrt{4\left(1 - \sqrt{\rho}\right)^2 + 6\left(1 - \sqrt{\rho}\right)(1 - \beta)^2} - 2\left(1 - \sqrt{\rho}\right)$

2. $1 - \frac{L\eta}{2}\left(\sqrt{1 + \frac{4}{L\eta}} - 1\right) < \beta < 1$

For all $K \geq 1$, we have

$$\frac{1}{K} \sum_{k=0}^{K-1} E[||\nabla F(\tilde{x}_k)||^2] \leq \frac{1}{C_1 K} (F(\tilde{x}^0) - F^*) + C_2(\sigma^2 + \zeta^2) + C_3\left(\frac{\sigma^2 + \zeta^2}{(1 - \sqrt{\rho})^2}\right)$$

where $\beta = \text{momentum coefficient}$, $\tilde{x}_k = \frac{1}{N} \sum_{i=1}^{N} x_i^k$
The result of the theorem, \[ \mathcal{O} \left( \frac{\sqrt{N}}{K} \right) \] 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 and momentum coefficient is presented in the Appendix. A.2.3. Further, we present a corollary to show the convergence rate of \( NGM_{dv} \) in terms of the number of iterations.

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

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

The proof for Corollary. 1 can be found in Appendix. A.2.4. Here, the constant \( C \) depended on the spectral gap \( \rho \), stochastic noise bound \( \sigma^2 \) and the inter gradient variance bound \( \zeta^2 \). The bound on \( \zeta^2 \) is the result of the assumption that the graphs with higher connectivity (lower spectral gap) can tolerate more non-IIDness. Corollary. 1 indicates that the proposed algorithm achieves linear speedup (with a convergence rate of \( \mathcal{O} \left( \frac{1}{\sqrt{N}} \right) \)) 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.

<table>
<thead>
<tr>
<th>Method</th>
<th>Rate</th>
<th>Assumption</th>
<th>Simplified Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>D-PSGD</td>
<td>( \mathcal{O} \left( \frac{1 + \sigma^2 + \sigma^2}{\sqrt{NK}} + \frac{N^2 \sigma^2}{K(1 - \sqrt{\rho})^2} + \frac{N^2 \zeta^2}{K(1 - \sqrt{\rho})^2} \right) )</td>
<td>-</td>
<td>( \mathcal{O} \left( \frac{1}{\sqrt{NK}} + \frac{1}{K} \right) )</td>
</tr>
<tr>
<td>CGA</td>
<td>( \mathcal{O} \left( \frac{1 + \sigma^2 + \sigma^2}{\sqrt{NK}} + \frac{N(\zeta^2 + \sigma^2 + \sigma^2)}{K(1 - \sqrt{\rho})^2} + \frac{\sqrt{N}}{K} \right) )</td>
<td>( \zeta^2 = \mathcal{O} \left( \frac{1}{K} \right) )</td>
<td>( \mathcal{O} \left( \frac{1}{\sqrt{NK}} + \frac{1}{K} + \frac{1}{K^2} \right) )</td>
</tr>
<tr>
<td>( NGM_{dv} ) (ours)</td>
<td>( \mathcal{O} \left( \frac{1 + \sigma^2 + \sigma^2}{\sqrt{NK}} + \frac{N(\zeta^2 + \sigma^2 + \sigma^2)}{K(1 - \sqrt{\rho})^2} + \frac{\sqrt{N}}{K} \right) )</td>
<td>( \zeta^2 = \mathcal{O} \left( \frac{1}{1 - \sqrt{\rho}/N} \right) )</td>
<td>( \mathcal{O} \left( \frac{1}{\sqrt{NK}} + \frac{1}{K} \right) )</td>
</tr>
</tbody>
</table>

### 5 Experiments

In this section, we analyze the performance of the proposed \( NGM_{mv} \) and \( NGM_{dv} \) techniques and compare them with the corresponding baseline i.e., D-PSGD algorithm \( \text{Lian et al. 2017} \), QG-DSGDm \( \text{Lin et al. 2021} \), Momentum Tracking \( \text{Takezawa et al. 2022} \) and state-of-the-art CGA method \( \text{Esfandiari et al. 2021} \). Here, we use low-resolution \((32 \times 32)\) images of the Imagenette dataset and state-of-the-art CGA method \( \text{Esfandiari et al. 2021} \) for experiments on a diverse set of datasets, model architectures, tasks, topologies, and numbers of agents. We present the analysis on – (a) Datasets (Appendix A.3): vision datasets (CIFAR-10, CIFAR-100, Fashion MNIST and Imagenette \( \text{Husain 2018} \)) and language datasets (AGNews \( \text{Zhang et al. 2015} \)). (b) Model architectures (Appendix A.6): 5-layer CNN, VGG-11, ResNet-20, LeNet-5, MobileNet-V2, ResNet-18, BERTmini and DistilBERTbase (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 \((32 \times 32)\) images of the Imagenette dataset for the experiments in Table. 3. The results for high resolution \((224 \times 224)\) Imagenette are presented in Table. 4.

*Our PyTorch code is available at [https://github.com/aparna-aketi/neighborhood_gradient_clustering](https://github.com/aparna-aketi/neighborhood_gradient_clustering)*
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 $NGM_{mv}$ are compared with D-PSGD and QG-DSGDm, $NGM_{dv}$ with Momentum Tracking and CGA. The communication cost for each experiment in this section is presented in Appendix A.9.

5.2 Results

Firstly, we evaluate variants of $NGM$ 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 $NGM_{mv}$ consistently outperforms QG-DSGDm for all models and graph sizes over ring topology with a significant performance gain varying from 2 – 20%. For the torus topology, $NGM_{mv}$ outperforms D-PSGD by 14 – 18% and QG-DSGDm fails to converge. Our experiments show the superiority of $NGM_{dv}$ over CGA. The performance gains are more pronounced when considering larger graphs (with 20 agents) and compact models such as ResNet-20. We observe that CGA and $NGM$ perform better than Momentum Tracking in most cases. We further demonstrate the generalizability of the proposed method by evaluating it on various image datasets (refer Table 3) such as Fashion MNIST, and Imagenette and on challenging datasets such as CIFAR-100. Table 3 show that $NGM_{mv}$ outperforms QG-DSGDm by 1 – 18% across various datasets whereas $NGM_{dv}$ remain competitive with an average improvement of $\sim 1\%$.

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.

<table>
<thead>
<tr>
<th>Method</th>
<th>Agents</th>
<th>5layer CNN Ring</th>
<th>5layer CNN Torus</th>
<th>VGG-11 Ring</th>
<th>ResNet-20 Ring</th>
</tr>
</thead>
<tbody>
<tr>
<td>D-PSGD</td>
<td>5</td>
<td>76.00 ± 1.44</td>
<td>-</td>
<td>67.04 ± 5.36</td>
<td>82.13 ± 0.84</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>47.68 ± 3.20</td>
<td>55.34 ± 6.32</td>
<td>44.14 ± 3.30</td>
<td>31.66 ± 6.01</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>44.85 ± 1.94</td>
<td>50.12 ± 1.91</td>
<td>38.92 ± 2.99</td>
<td>31.94 ± 2.91</td>
</tr>
<tr>
<td>QG-DSGDm</td>
<td>5</td>
<td>78.77 ± 0.69</td>
<td>-</td>
<td>81.71 ± 0.45</td>
<td>82.59 ± 2.66</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>47.83 ± 3.39</td>
<td>14.48 ± 0.50</td>
<td>57.77 ± 0.33</td>
<td>46.09 ± 7.90</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>46.57 ± 4.80</td>
<td>17.25 ± 5.81</td>
<td>32.32 ± 7.61</td>
<td>44.78 ± 3.39</td>
</tr>
<tr>
<td>$NGM_{mv}$ (ours)</td>
<td>5</td>
<td>82.20 ± 0.34</td>
<td>-</td>
<td>83.65 ± 0.38</td>
<td>85.88 ± 0.58</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>67.43 ± 1.15</td>
<td>73.84 ± 0.33</td>
<td>59.92 ± 2.12</td>
<td>66.02 ± 2.86</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>58.80 ± 1.30</td>
<td>64.55 ± 1.16</td>
<td>52.70 ± 1.63</td>
<td>50.74 ± 2.36</td>
</tr>
<tr>
<td>Momentum Tracking</td>
<td>5</td>
<td>80.34 ± 0.67</td>
<td>-</td>
<td>79.62 ± 7.42</td>
<td>85.65 ± 0.30</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>58.54 ± 7.51</td>
<td>67.46 ± 1.10</td>
<td>73.51 ± 1.42</td>
<td>66.37 ± 7.72</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>36.58 ± 11.5</td>
<td>56.71 ± 1.96</td>
<td>36.12 ± 13.8</td>
<td>44.35 ± 11.7</td>
</tr>
<tr>
<td>CGA</td>
<td>5</td>
<td>82.20 ± 0.43</td>
<td>-</td>
<td>84.41 ± 0.22</td>
<td>87.52 ± 0.50</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>72.96 ± 0.40</td>
<td>76.04 ± 0.62</td>
<td>79.66 ± 0.46</td>
<td>79.98 ± 1.23</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>69.88 ± 0.84</td>
<td>73.21 ± 0.27</td>
<td>73.30 ± 0.12</td>
<td>75.13 ± 1.56</td>
</tr>
<tr>
<td>$NGM_{dv}$ (ours)</td>
<td>5</td>
<td>83.36 ± 0.65</td>
<td>-</td>
<td>85.15 ± 0.58</td>
<td>88.52 ± 0.19</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>75.34 ± 0.30</td>
<td>78.53 ± 0.56</td>
<td>79.55 ± 0.30</td>
<td>84.02 ± 0.44</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>73.36 ± 0.88</td>
<td>75.11 ± 0.07</td>
<td>79.43 ± 0.62</td>
<td>81.26 ± 0.69</td>
</tr>
</tbody>
</table>

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$_{min}$ model with the AGNews dataset distributed over 4 and 8 agents and DistilBert$_{base}$ distributed over 4 agents. For $NGM_{mv}$, we see a maximum improvement of 2.1% over the baseline D-PSGD algorithm. Even for the text classification task, we observe $NGM_{dv}$ to be competitive with CGA. These observations are consistent with the results of the image classification.
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.

<table>
<thead>
<tr>
<th>Method</th>
<th>Agents</th>
<th>Fashion MNIST (LeNet-5)</th>
<th>CIFAR-100 (ResNet-20)</th>
<th>Imagenette (32×32) (MobileNet-V2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>D-PSGD</td>
<td>5</td>
<td>86.43 ± 0.14</td>
<td>44.66 ± 5.23</td>
<td>47.09 ± 9.20</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>75.49 ± 0.32</td>
<td>19.03 ± 13.27</td>
<td>32.81 ± 2.18</td>
</tr>
<tr>
<td>QG-DSGDm</td>
<td>5</td>
<td>89.28 ± 0.52</td>
<td>49.60 ± 3.30</td>
<td>43.70 ± 2.80</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>84.38 ± 0.58</td>
<td>41.28 ± 0.61</td>
<td>17.80 ± 2.66</td>
</tr>
<tr>
<td>NGM$_{mv}$ (ours)</td>
<td>5</td>
<td>90.03 ± 0.11</td>
<td>55.96 ± 0.95</td>
<td>60.15 ± 2.17</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>87.10 ± 0.47</td>
<td>49.30 ± 1.02</td>
<td>36.13 ± 1.97</td>
</tr>
<tr>
<td>Momentum Tracking</td>
<td>5</td>
<td>90.52 ± 0.56</td>
<td>49.52 ± 2.36</td>
<td>38.79 ± 2.21</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>87.82 ± 0.19</td>
<td>37.74 ± 4.74</td>
<td>23.07 ± 11.3</td>
</tr>
<tr>
<td>CGA</td>
<td>5</td>
<td>90.03 ± 0.39</td>
<td>56.43 ± 2.39</td>
<td>72.82 ± 1.25</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>87.61 ± 0.30</td>
<td>53.61 ± 1.07</td>
<td>61.97 ± 0.58</td>
</tr>
<tr>
<td>NGM$_{dv}$ (ours)</td>
<td>5</td>
<td>90.61 ± 0.18</td>
<td>56.50 ± 3.23</td>
<td>74.49 ± 0.93</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>87.24 ± 0.23</td>
<td>53.77 ± 0.15</td>
<td>64.06 ± 1.11</td>
</tr>
</tbody>
</table>

tasks. Finally, through this exhaustive set of experiments, we demonstrate that the NGM$_{dv}$ can serve as a simple plugin to boost the performance of decentralized learning on non-IID data without significant memory and compute overheads. Further, NGM$_{mv}$ demonstrates that locally available model-variant cross-gradient information at each agent can be efficiently utilized to improve decentralized learning with no communication overhead.

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.

<table>
<thead>
<tr>
<th>Method</th>
<th>AGNews-BERT$_{mini}$</th>
<th>AGNews-DistilBERT$_{base}$</th>
<th>Imagenette (224×224)-ResNet-18</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Agents = 4</td>
<td>Agents = 8</td>
<td>Agents = 5</td>
</tr>
<tr>
<td>D-PSGD</td>
<td>89.21 ± 0.41</td>
<td>85.48 ± 0.71</td>
<td>65.43 ± 4.60</td>
</tr>
<tr>
<td>QG-DSGDm</td>
<td>87.38 ± 5.29</td>
<td>76.61 ± 3.63</td>
<td>74.22 ± 2.30</td>
</tr>
<tr>
<td>NGM$_{mv}$ (ours)</td>
<td><strong>89.40 ± 0.13</strong></td>
<td><strong>87.58 ± 0.07</strong></td>
<td><strong>78.26 ± 0.67</strong></td>
</tr>
<tr>
<td>Momentum Tracking</td>
<td>82.45 ± 0.13</td>
<td>87.82 ± 0.047</td>
<td>81.11 ± 0.29</td>
</tr>
<tr>
<td>CGA</td>
<td>91.43 ± 0.11</td>
<td>89.15 ± 0.45</td>
<td>85.00 ± 0.67</td>
</tr>
<tr>
<td>NGM$_{dv}$ (ours)</td>
<td><strong>92.24 ± 0.29</strong></td>
<td><strong>89.02 ± 0.39</strong></td>
<td><strong>85.85 ± 0.60</strong></td>
</tr>
</tbody>
</table>

Figure 1: Average Validation loss of NGM$_{mv}$ during training CIFAR-10 on a 5-layer CNN with 5 agents.
5.3 Analysis

We empirically show the generalization characteristics in terms of validation loss convergence for the proposed algorithm on 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 generalization characteristics of the NGM algorithm with the respective baselines. For the same decentralized setup, we observe that NGM has 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 NGM is much lower than that of the D-PSGD baseline resulting in better performance of NGM. We then compare the model variance and data variance bias terms of NGM with CGA as shown in Figure 3(b) and 3(c) respectively. We observe that both the model variance and the data variance bias for NGM are significantly lower than CGA. This is because CGA gives more importance to self-_gradients as it updates in the descent direction which is close to self- gradients and is positively correlated to data-variant cross-gradients. In contrast, NGM accounts for the biases directly and gives equal importance to self- gradients and data-variant cross-gradients, thereby achieving superior performance. Further, Figure 4(a) presents the average consensus error i.e., \( \frac{1}{n} \| x_k - \bar{x}_k \|_2^2 \) over time for training CIFAR-10 dataset on 5-layer CNN with respect to various algorithms. This empirically shows that the proposed NGM algorithm reaches the consensus at the same rate as CGA.

Additionally, we show that NGM can be used in synergy with QGM to further improve the performance of decentralized learning on non-IID Data. Figure 4(b) shows the test accuracy on the CIFAR-10 dataset trained on ResNet-20 over ring topology. We observe that the QGM version of NGM performs better than NGM by 16 – 27%. Similarly, the QGM version of NGM performs 1.3 – 2.5% better than the local momentum version of NGM.
5.4 Communication Compression

In this section, we discuss the impact of communication compression on the proposed \textit{NGM} algorithm. At every iteration, the \textit{NGM}_{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 explore a compressed version of \textit{NGM} referred to as \textit{CompNGM} using Error Feedback SGD (EF-SGD) (Karimireddy et al., 2019) to compress the cross-gradients. \textit{CompNGM} compresses the error-compensated cross-gradients from 32 bits (floating point precision of arithmetic operations) to 1 bit by using scaled signed gradients. The pseudo-code for \textit{CompNGM} is shown in Algorithm 3 in the Appendix. Additional results and analysis on \textit{CompNGM} are presented in the Appendix. A.4.

Table 5: Test Accuracy of \textit{CompNGM}, Accuracy drop compared to \textit{NGM}_{dv} and communication reduction in Giga Bytes compared to \textit{NGM}_{dv} for various datasets trained on ring topology.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Model</th>
<th>5 agents</th>
<th>10 agents</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Acc. (%)</td>
<td>Acc. Drop (%)</td>
<td>Comm. Drop (GB)</td>
</tr>
<tr>
<td>Fashion MNIST</td>
<td>LeNet-5</td>
<td>90.48 ± 0.19</td>
<td>0.13</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>ResNet-20</td>
<td>87.56 ± 0.34</td>
<td>0.96</td>
</tr>
<tr>
<td>CIFAR-100</td>
<td>ResNet-20</td>
<td>57.51 ± 0.48</td>
<td>-1.01</td>
</tr>
<tr>
<td>Imagenette (32×32)</td>
<td>MobileNet-V2</td>
<td>72.91 ± 1.06</td>
<td>1.58</td>
</tr>
</tbody>
</table>

Table 5 shows that \textit{CompNGM} reduces the communication cost by 1.94× by trading off 0.1 – 1.57% test accuracy for 5 agents and 2 – 10% test accuracy for 10 agents. This shows that \textit{NGM} can be combined with communication compression techniques to achieve a trade-off between accuracy boost and communication overhead. Further, \textit{NGM} can be combined with stronger compression methods such as EF21 (Richtárik et al., 2021) to have a better trade-off between accuracy and communication cost.

5.5 Hardware Benefits

The proposed \textit{NGM}_{dv} algorithm is superior in terms of memory and compute efficiency (see Table 6 and Appendix. A.8) while having equal communication cost as compared to \textit{CGA}. \textit{NGM}_{dv} replaces the complicated quadratic projection step of \textit{CGA} with a simple weighted averaging step. Since \textit{NGM}_{dv} involves weighted averaging, an additional buffer to store the cross-gradients is not required. \textit{CGA} stores all the cross-gradients in a matrix form for quadratic programming projection of the local gradient. Therefore, \textit{NGM} has no memory overhead compared to the baseline D-PSGD algorithm, while \textit{CGA} requires additional memory equivalent to the number of neighbors times model size. Moreover, the quadratic programming projection step \cite{Goldfarb1983} in \textit{CGA} is much more expensive in compute and latency than the weighted averaging step of cross-gradients in \textit{NGM}. Our evaluation clearly shows that \textit{NGM} is superior to \textit{CGA} in terms of test accuracy, memory efficiency, compute efficiency, and latency.
Table 6: Comparison of communication, memory, and compute overheads per mini-batch compared to D-PSGD. \(m_s\): model size, \(n_i\): number of neighbors, \(b\): floating point precision, Q: compute for Quadratic Programming, B: compute for Backward Pass. Note that \(O(Q) \gg O(n_im_s)\) (Nesterov & Todd, 1997).

<table>
<thead>
<tr>
<th>Method</th>
<th>Comm.</th>
<th>Memory</th>
<th>Compute</th>
</tr>
</thead>
<tbody>
<tr>
<td>QG-DSGDm</td>
<td>0</td>
<td>(O(m_s))</td>
<td>(O(n_im_s))</td>
</tr>
<tr>
<td>(NGM_{mv})</td>
<td>0</td>
<td>0</td>
<td>(O(n_im_s + n_iB))</td>
</tr>
<tr>
<td>Momentum Tracking</td>
<td>(O(n_im_s))</td>
<td>(O(2m_s))</td>
<td>(O(n_im_s))</td>
</tr>
<tr>
<td>CGA</td>
<td>(O(n_im_s))</td>
<td>(O(n_im_s))</td>
<td>(O(Q + n_iB))</td>
</tr>
<tr>
<td>(NGM_{dv})</td>
<td>(O(n_im_s))</td>
<td>0</td>
<td>(O(n_im_s + n_iB))</td>
</tr>
<tr>
<td>CompCGA</td>
<td>(O(\frac{n_im_s}{n_i}))</td>
<td>(O(n_im_s))</td>
<td>(O(Q + n_iB))</td>
</tr>
<tr>
<td>CompNGM</td>
<td>(O(\frac{n_im_s}{n_i}))</td>
<td>(O(n_im_s))</td>
<td>(O(n_im_s + n_iB))</td>
</tr>
</tbody>
</table>

6 Discussion and Limitations

As mentioned in Section [3.1], the \(NGM\) mixing weight \(\alpha\) is either set to 0 or 1 resulting in \(NGM_{mv}\) or \(NGM_{dv}\) respectively. This implies that \(NGM_{mv}\) utilizes model-variant cross-gradients whereas, \(NGM_{dv}\) utilizes data-variant cross-gradients. However, the \(NGM_{dv}\) 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 \(NGM\) mixing weight \(\alpha\) in the range of [0, 1]. Figure 3(d) depicts the change in average validation accuracy with \(\alpha\) and we observe that \(\alpha = 1\) performs best. Further, Figure 3(b) shows that the model variant bias reduces for \(NGM_{dv}\), even though it does not explicitly utilize the model-variant cross-gradients. We hypothesize that, in \(NGM_{dv}\), 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. In summary, for \(1 \times\) communication constraints i.e., when agents do not have access to data-variant cross-gradients, \(NGM_{mv}\) effectively utilizes locally available model-variant cross-gradients to improve the accuracy. On the contrary, when \(2 \times\) communication is allowed (non-zero \(\alpha\)), the best performance is achieved by only utilizing the data-variant cross-gradients.


<table>
<thead>
<tr>
<th>Trained for 60 epochs</th>
<th>D-PSGD</th>
<th>QG-DSGDm</th>
<th>(NGM_{mv}) +LM</th>
<th>(NGM_{mv}) +QGM</th>
<th>(NGM_{mv})</th>
<th>Momentum Tracking</th>
<th>CGA</th>
<th>(NGM_{dv}) +LM</th>
<th>(NGM_{dv}) +QGM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy (%)</td>
<td>25.02</td>
<td>45.19</td>
<td>42.66</td>
<td>49.22</td>
<td>51.00</td>
<td>43.65</td>
<td>-</td>
<td>49.42</td>
<td>54.31</td>
</tr>
<tr>
<td>Epoch Time (min)</td>
<td>7.6</td>
<td>7.9</td>
<td>18.6</td>
<td>18.8</td>
<td>19.1</td>
<td>10.7</td>
<td>406</td>
<td>21.5</td>
<td>22.2</td>
</tr>
</tbody>
</table>
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 Mean (NGM) algorithm with two variants that improve decentralized learning on non-IID data. $NGM_{mv}$ improves the performance of decentralized learning utilizing model-variant cross-gradients without any communication overhead. $NGM_{dv}$ further increases these gains by operating on data-variant cross-gradients. Additionally, we present a compressed version of our algorithm ($CompNGM$) to reduce the communication overhead associated with data-variant cross-gradients of $NGM_{dv}$. 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.

Authors’ Contributions:

Sai Aparna Aketi and Sangamesh Kodge, both worked on developing the algorithm. The experiments for Computer Vision tasks were designed by Sai Aparna Aketi and the experiments for Natural Language Processing tasks were designed by Sangamesh Kodge. The theory for the convergence analysis and consensus error bounds was developed by Sai Aparna Aketi and was verified by Sangamesh Kodge. All the authors contributed equally in writing and proofreading the paper.

Acknowledgments

This work was supported in part by, the Center for Brain-inspired Computing (C-BRIC), the Center for the Co-Design of Cognitive Systems (COCOSYS), a DARPA-sponsored JUMP center, the Semiconductor Research Corporation (SRC), the National Science Foundation, and DARPA ShELL.

References


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.


A Appendix

A.1 Proof of Lemma. [1]

This section presents detailed proof for Lemma. [1] The NGM algorithm modifies the local gradients as follows

\[ \tilde{g}^i = (1 - \alpha) \sum_{j \in N_i} w_{ij} g^{ij} + \alpha \sum_{j \in N_i} w_{ij} g^{ij} = g^i + \sum_{j \in N_i} w_{ij} (g^{ij} - g^i) \quad \text{(NGM}_{\text{dv}} \text{ uses } \alpha = 1) \]

\[ \tilde{g}^i - g^i = \sum_{j \in N_i} w_{ij} (\nabla F_j(x, d^j) - \nabla F_i(x, d^i)) \]

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

\[ \mathbb{E} \left[ \left\| \frac{1}{N} \sum_{i=1}^{N} (\tilde{g}^i - g^i) \right\|^2 \right] = \mathbb{E} \left[ \left\| \frac{1}{N} \sum_{i=1}^{N} \sum_{j \in N_i} w_{ij} (\nabla F_j(x, d^j) - \nabla F_i(x, d^i)) \right\|^2 \right] \]

\[ = (a) \mathbb{E} \left[ \left\| \frac{1}{N} \sum_{i=1}^{N} \sum_{j \in N_i} w_{ij} (\nabla F_j(x, d^j) - \nabla F_i(x, d^i) - \mathbb{E} \nabla F_j(x, d^j) - \mathbb{E} \nabla F_i(x, d^i)) \right\|^2 \right] \]

\[ + \mathbb{E} \left[ \left\| \frac{1}{N} \sum_{i=1}^{N} \sum_{j \in N_i} w_{ij} (\mathbb{E} \nabla F_j(x, d^j) - \nabla F_i(x, d^i)) \right\|^2 \right] \]

\[ = (b) \mathbb{E} \left[ \left\| \frac{1}{N} \sum_{i=1}^{N} \sum_{j \in N_i} w_{ij} (\nabla f_j(x) - \nabla f_i(x)) \right\|^2 \right] \]

\[ + \mathbb{E} \left[ \left\| \frac{1}{N} \sum_{i=1}^{N} \sum_{j \in N_i} w_{ij} (\nabla f_j(x) - \nabla f_i(x)) \right\|^2 \right] \]

\[ \leq (c) \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j \in N_i} w_{ij} \mathbb{E} \left[ \left\| \nabla F_j(x, d^j) - \nabla f_j(x) - (\nabla F_i(x, d^i) - \nabla f_i(x)) \right\|^2 \right] \]

\[ + \frac{1}{N} \sum_{i=1}^{N} \sum_{j \in N_i} w_{ij} \mathbb{E} \left[ \left\| \nabla f_j(x) - \nabla F(x) + \nabla F(x) - \nabla f_i(x) \right\|^2 \right] \]

\[ \leq (d) \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j \in N_i} w_{ij} 4\sigma^2 + \frac{1}{N} \sum_{i=1}^{N} \sum_{j \in N_i} w_{ij} \mathbb{E} \left[ \left\| \nabla f_j(x) - \nabla F(x) + \nabla F(x) - \nabla f_i(x) \right\|^2 \right] \]

\[ = \frac{4N\sigma^2}{N^2} + \frac{1}{N} \sum_{i=1}^{N} \sum_{j \in N_i} w_{ij} \mathbb{E} \left[ \left\| \nabla f_j(x) - \nabla F(x) + \nabla F(x) - \nabla f_i(x) \right\|^2 \right] \]

\[ \leq (f) \frac{4\sigma^2}{N} + \frac{1}{N} \sum_{i=1}^{N} \sum_{j \in N_i} w_{ij} \left( 2\mathbb{E} \left[ \left\| \nabla f_j(x) - \nabla F(x) \right\|^2 \right] + 2\mathbb{E} \left[ \left\| \nabla F(x) - \nabla F(x) \right\|^2 \right] \right) \]

\[ \leq (g) \frac{4\sigma^2}{N} + \frac{1}{N} \sum_{i=1}^{N} \sum_{j \in N_i} w_{ij} 4\zeta^2 \]
\[ \frac{4\sigma^2}{N} + 4\zeta^2 \]

(a) identity \( \mathbb{E}[||Z||^2] = \mathbb{E}[||Z - E[Z]||^2 + \mathbb{E}[||E[Z]||^2] \) 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 \( E_{d_i} \mathbb{E}[\sum_{i=1}^N Z_i] \); (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 \( ||\nabla f_i(x) - \nabla F(x)||^2 \leq \zeta^2 \) for any vectors \( a_1, a_2 \).

\[ E[||\bar{g} - g_i||^2] \leq 4(\frac{\sigma^2}{N} + \zeta^2) \]

A.2 Convergence Analysis

In this section, we present the proof for our main theorem, indicating the convergence of the proposed \( NGM_{da} \) algorithm. Before we proceed to the proof of the theorem, we present a lemma showing that \( NGM \) achieves consensus among the different agents.

A.2.1 Bound for Consensus Error

**Lemma 2.** Given assumptions 1-3 and lemma 3, the distance between the average sequence iterate \( \bar{x}_k \) and the sequence iterates \( x_{i_1}^k \)’s generated \( NGM \) (i.e., the consensus error of the proposed algorithm) is given by

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

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

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

\[ \bar{z}_k := \frac{1}{1-\beta} \bar{x}_k - \frac{\beta}{1-\beta} \bar{x}_{k-1} \]

Where \( k > 0 \) and \( x_k \) 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.

\[ \bar{v}_k = \beta \bar{v}_{k-1} - \eta \frac{1}{N} \sum_{i=1}^{N} \bar{g}_{i_{k-1}} \]

\[ \bar{x}_k = \bar{x}_{k-1} + \bar{v}_k \]

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

To prove Lemma 2 we use the following facts.

\[ \bar{z}_{k+1} - \bar{z}_k = -\frac{\eta}{1-\beta} \frac{1}{N} \sum_{i=1}^{N} \bar{g}_i. \]

\[ \sum_{k=0}^{K-1} ||\bar{z}_k - \bar{x}_k||^2 \leq \frac{\eta^2 \beta^2}{(1-\beta)^4} \sum_{k=0}^{K-1} \frac{1}{N} \sum_{i=1}^{N} \bar{g}_i^2. \]

Note that the proof for Eq. 14 and Eq. 15 can be found in \( \text{(Esfandiari et al., 2021)} \) 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:

\[ \mathbf{G}_k \triangleq [\mathbf{g}_1, \mathbf{g}_2^2, \ldots, \mathbf{g}_k^N] \]
\[ \mathbf{V}_k \triangleq [\mathbf{v}_k, \mathbf{v}_k^2, \ldots, \mathbf{v}_k^N] \]
\[ \mathbf{X}_k \triangleq [\mathbf{x}_1, \mathbf{x}_2^2, \ldots, \mathbf{x}_k^N] \]
\[ \mathbf{G}_k \triangleq [\mathbf{g}_1, \mathbf{g}_2^2, \ldots, \mathbf{g}_k^N] \]
\[ \mathbf{H}_k \triangleq [\nabla f_1(\mathbf{x}_1^k), \nabla f_2(\mathbf{x}_2^k), \ldots, \nabla f_N(\mathbf{x}_N^k)] \]

We can observe that the above matrices are all with dimension \( d \times N \) such that any matrix \( \mathbf{A} \) satisfies
\[ \| \mathbf{A} \|^2 \geq \sum_{i=1}^N \| \mathbf{a}_i \|^2, \]
where \( \mathbf{a}_i \) is the \( i \)-th column of the matrix \( \mathbf{A} \). Thus, we can obtain that:
\[ \| \mathbf{X}_k(\mathbf{I} - \mathbf{Q}) \|^2 = \sum_{i=1}^N \| \mathbf{x}_i^k - \bar{\mathbf{x}}_k \|^2. \]  

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

(i) \( \mathbf{QW} = \mathbf{WQ} \);

(ii) \( (\mathbf{I} - \mathbf{Q})\mathbf{W} = \mathbf{W}(\mathbf{I} - \mathbf{Q}) \);

(iii) For any integer \( k \geq 1 \), \( \| (\mathbf{I} - \mathbf{Q})\mathbf{W} \|_\sigma \leq (\sqrt{\sigma})^k \), where \( \| \cdot \|_\sigma \) is the spectrum norm of a matrix.

Let \( \mathbf{A}_i, i \in \{1, 2, \ldots, N\} \) be \( N \) arbitrary real square matrices. It follows that
\[ \| \sum_{i=1}^N \mathbf{A}_i \|^2 \leq \sum_{i=1}^N \sum_{j=1}^N \mathbf{A}_i \| \mathbf{A}_j \|_\sigma. \]

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

**Lemma 3.** Let all assumptions hold. Let \( \mathbf{g}^i \) 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 \( i \in [N] := \{1, 2, \ldots, N\} \). Thus the following relationship holds
\[ \mathbb{E} \left[ \left\| \frac{1}{N} \sum_{i=1}^N \mathbf{g}^i \right\|^2 \right] \leq \frac{10\sigma^2}{N} + 8\zeta^2 + 2\mathbb{E} \left[ \left\| \frac{1}{N} \sum_{i=1}^N \nabla f_i(\mathbf{x}^i) \right\|^2 \right] \]

**Proof for Lemma 3:**
\[ \mathbb{E} \left[ \left\| \frac{1}{N} \sum_{i=1}^N \mathbf{g}^i \right\|^2 \right] = \mathbb{E} \left[ \left\| \frac{1}{N} \sum_{i=1}^N (\mathbf{g}^i - \mathbf{g}^i + \mathbf{g}^i) \right\|^2 \right] \]
\[ \leq a 2\mathbb{E} \left[ \left( \frac{1}{N} \sum_{i=1}^N (\mathbf{g}^i - \mathbf{g}^i) \right)^2 \right] + 2\mathbb{E} \left[ \left( \frac{1}{N} \sum_{i=1}^N \mathbf{g}^i \right)^2 \right] \]
\[ \leq b 8(\frac{\sigma^2}{N} + \zeta^2) + \frac{2\sigma^2}{N} + 2\mathbb{E} \left[ \left( \frac{1}{N} \sum_{i=1}^N \nabla f_i(\mathbf{x}^i) \right)^2 \right] \]
\[ \leq \frac{10\sigma^2}{N} + 8\zeta^2 + 2\mathbb{E} \left[ \left( \frac{1}{N} \sum_{i=1}^N \nabla f_i(\mathbf{x}^i) \right)^2 \right] \]

18
(a) refers to the fact that the inequality \(|a + b|^2 \leq 2|a|^2 + 2|b|^2\). (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 \(\nabla f_i(x^i)\) at the point \(x^i\) such that \(\mathbb{E}[g^i] = \nabla f_i(x^i)\), for all \(i \in [N] := \{1, 2, ..., N\}\). Thus the following relationship holds

\[
\mathbb{E} \left[ \left\| G - G^* \right\|_F^2 \right] \leq 4N(\sigma^2 + \zeta^2) \tag{21}
\]

**Proof for Lemma 4** is similar to Lemma 3 and is as follows:

\[
\mathbb{E} \left[ \left\| G - G^* \right\|_F^2 \right] = \sum_{i=1}^{N} \mathbb{E} \left[ \left\| \sum_{j \in N_i} w_{ij} (\nabla F_j(x, d^j) - \nabla F_i(x, d^j)) \right\|_2^2 \right]
\]

\[
= \sum_{i=1}^{N} \mathbb{E} \left[ \left\| \sum_{j \in N_i} w_{ij} (\nabla F_j(x, d^j) - \nabla f_j(x)) - (\nabla F_i(x, d^i) - \nabla f_i(x)) \right\|_2^2 \right]
\]

\[
+ \sum_{i=1}^{N} \mathbb{E} \left[ \left\| \sum_{j \in N_i} w_{ij} (\nabla f_j(x) - \nabla f_i(x)) \right\|_2^2 \right]
\]

\[
\leq \sum_{i=1}^{N} \sum_{j \in N_i} w_{ij} \mathbb{E} \left[ \left\| \nabla F_j(x, d^j) - \nabla f_j(x) - (\nabla F_i(x, d^i) - \nabla f_i(x)) \right\|_2^2 \right]
\]

\[
+ \sum_{i=1}^{N} \sum_{j \in N_i} w_{ij} \mathbb{E} \left[ \left\| \nabla f_j(x) - \nabla f_i(x) \right\|_2^2 \right]
\]

\[
\leq 4N(\sigma^2 + \zeta^2) \quad \text{(follows from assumption 2)}
\]

The properties shown in Eq. 18 and 19 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 \(X_k = X_{k-1}W + V_k\) we have:

\[
X_k(I - Q) = X_{k-1}(I - Q)W + V_k(I - Q) \tag{22}
\]

Applying the above equation \(k\) times we have:

\[
X_k(I - Q) = X_0(I - Q)W^k + \sum_{\tau=1}^{k} V_\tau(I - Q)W^{k-\tau} X_n = \sum_{\tau=1}^{k} V_\tau(I - Q)W^{k-\tau} \tag{23}
\]

As \(V_k = \beta V_{k-1} - \eta \frac{1}{N} \sum_{i=1}^{N} G^i_{k-1} \), we can get:

\[
X_k(I - Q) = -\eta \sum_{\tau=1}^{k} \sum_{l=0}^{\tau-1} \tilde{G}_\tau \beta^\tau - 1 - l (I - Q)W^{k-\tau}
\]

\[
= -\eta \sum_{\tau=1}^{k} \sum_{l=0}^{\tau-1} \tilde{G}_\tau \beta^\tau - 1 - l W^{k-\tau} (I - Q) - \eta \sum_{n=1}^{k-1} \tilde{G}_n \sum_{l=n+1}^{k} \beta^{l-1-n} W^{k-1-n} (I - Q) \tag{24}
\]

\[
= -\eta \sum_{\tau=0}^{k-1} \frac{1 - \beta \tau}{1 - \beta} \tilde{G}_\tau (I - Q)W^{k-1-\tau}.
\]
Therefore, for $k \geq 1$, we have:

$$
E \left[ \left\| X_k (I - Q) \right\|_3^2 \right] = \eta^2 E \left[ \left\| \sum_{\tau=0}^{k-1} \frac{1 - \beta^{k-\tau}}{1 - \beta} G_\tau (I - Q) W^{k-1-\tau} \right\|_3^2 \right]
\leq \frac{a}{\beta} E \left[ \left\| \sum_{\tau=0}^{k-1} \frac{1 - \beta^{k-\tau}}{1 - \beta} (G_\tau - G_\tau) (I - Q) W^{k-1-\tau} \right\|_3^2 \right] + 2 \eta^2 E \left[ \left\| \sum_{\tau=0}^{k-1} \frac{1 - \beta^{k-\tau}}{1 - \beta} G_\tau (I - Q) W^{k-1-\tau} \right\|_3^2 \right] 
\tag{25}
$$

(a) follows from the inequality $\|A + B\|_3^2 \leq 2A_3 + 2B_3$, (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 $II$.

$$
E \left[ \left\| \sum_{\tau=0}^{k-1} \frac{1 - \beta^{k-\tau}}{1 - \beta} G_\tau (I - Q) W^{k-1-\tau} \right\|_3^2 \right] \leq \sum_{\tau=0}^{k-1} \sum_{\tau' = 0}^{k-1} E \left[ \left\| \frac{1 - \beta^{k-\tau}}{1 - \beta} G_\tau (I - Q) W^{k-1-\tau} \right\|_3^2 \right] 
\leq \frac{1}{(1 - \beta)^2} \sum_{\tau=0}^{k-1} \sum_{\tau' = 0}^{k-1} \rho^{(k-1 - \tau - \tau')} E \left[ \left\| G_\tau \|_3 \right\| G_{\tau'} \|_3 \right] 
\leq \frac{b}{(1 - \beta)^2} \sum_{\tau=0}^{k-1} \sum_{\tau' = 0}^{k-1} \rho^{(k-1 - \tau - \tau')} \left( \frac{1}{2} E[\|G_\tau\|_3^2] + \frac{1}{2} E[\|G_{\tau'}\|_3^2] \right) 
\leq \frac{1}{(1 - \beta)^2} \sum_{\tau=0}^{k-1} \sum_{\tau' = 0}^{k-1} \rho^{(k-1 - \tau - \tau')} E[\|G_\tau\|_3^2] 
\leq \frac{c}{(1 - \beta)^2 (1 - \sqrt{\rho})} \sum_{\tau=0}^{k-1} \rho^{(k-1 - \tau)} E[\|G_\tau\|_3^2] 
\tag{27}
$$

(a) follows from Eq. 19, (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 $\sum_{\tau=0}^{k-1} \rho^{k-1 - \tau} \leq \frac{\rho^{1 - \frac{1}{2}}}{1 - \sqrt{\rho}}$. 

20
We then proceed with finding the bounds for $\mathbb{E}[\|G_{\tau}\|_{\mathfrak{g}}^{2}]$:

$$
\mathbb{E}[\|G_{\tau}\|_{\mathfrak{g}}^{2}] = \mathbb{E}[\|G_{\tau} - H_{\tau} + H_{\tau} - H_{\tau}Q + H_{\tau}Q\|_{\mathfrak{g}}^{2}]
\leq 3\mathbb{E}[\|G_{\tau} - H_{\tau}\|_{\mathfrak{g}}^{2}] + 3\mathbb{E}[\|H_{\tau}(I - Q)\|_{\mathfrak{g}}^{2}] + 3\mathbb{E}[\|H_{\tau}Q\|_{\mathfrak{g}}^{2}]
\leq 3N\sigma^{2} + 3N\zeta^{2} + 3\mathbb{E}[\|\frac{1}{N} \sum_{i=1}^{N} \nabla f_{i}(x_{\tau}^{i})\|^{2}]
$$

(a) holds because $\mathbb{E}[\|H_{\tau}Q\|_{\mathfrak{g}}^{2}] \leq \mathbb{E}[\|\frac{1}{N} \sum_{i=1}^{N} \nabla f_{i}(x_{\tau}^{i})\|^{2}]$

Substituting (28) in (27):

$$
\mathbb{E}\left[\left\| \sum_{\tau=0}^{k-1} \frac{1 - \beta^{k-\tau}}{1 - \beta} G_{\tau}(I - Q) W^{k-1-\tau}\right\|_{\mathfrak{g}}^{2}\right] \leq \frac{1}{(1 - \beta)^{2}(1 - \sqrt{\rho})} \sum_{\tau=0}^{k-1} \rho^{(k-1-\tau)} \left[ 3N\sigma^{2} + 3N\zeta^{2} + 3N\mathbb{E}[\|\frac{1}{N} \sum_{i=1}^{N} \nabla f_{i}(x_{\tau}^{i})\|^{2}] \right]
\leq \frac{3N\sigma^{2} + 3N\zeta^{2}}{(1 - \beta)^{2}(1 - \sqrt{\rho})} + \frac{3N\mathbb{E}[\|\frac{1}{N} \sum_{i=1}^{N} \nabla f_{i}(x_{\tau}^{i})\|^{2}]}{(1 - \beta)^{2}(1 - \sqrt{\rho})} \sum_{\tau=0}^{k-1} \rho^{(k-1-\tau)}
$$

substituting (29) and (26) into the main inequality (25):

$$
\mathbb{E}\left[\left\| X_{k}(I - Q)\right\|_{\mathfrak{g}}^{2}\right] \leq \frac{8\eta^{2}N\sigma^{2} + \zeta^{2}}{(1 - \beta)^{2}(1 - \rho)} + \frac{2\eta^{2}}{(1 - \beta)^{2}(1 - \sqrt{\rho})} \left( \frac{3N\sigma^{2}}{1 - \sqrt{\rho}} + \frac{3N\zeta^{2}}{1 - \sqrt{\rho}} + 3N\mathbb{E}[\|\frac{1}{N} \sum_{i=1}^{N} \nabla f_{i}(x_{\tau}^{i})\|^{2}] \right)
= \frac{2\eta^{2}}{(1 - \beta)^{2}} \left( \frac{4N\sigma^{2} + \zeta^{2}}{1 - \rho} + \frac{3N\sigma^{2}}{(1 - \sqrt{\rho})^{2}} + \frac{3N\zeta^{2}}{(1 - \sqrt{\rho})^{2}} \right) + \frac{6N\eta^{2}}{(1 - \beta)^{2}(1 - \sqrt{\rho})} \sum_{\tau=0}^{k-1} \rho^{(k-1-\tau)} \mathbb{E}[\|\frac{1}{N} \sum_{i=1}^{N} \nabla f_{i}(x_{\tau}^{i})\|^{2}]
\leq \frac{14N\eta^{2}}{(1 - \beta)^{2}} \left( \frac{\sigma^{2}}{(1 - \sqrt{\rho})^{2}} + \frac{\zeta^{2}}{(1 - \sqrt{\rho})^{2}} \right) + \frac{6N\eta^{2}}{(1 - \beta)^{2}(1 - \sqrt{\rho})} \sum_{\tau=0}^{k-1} \rho^{(k-1-\tau)} \mathbb{E}[\|\frac{1}{N} \sum_{i=1}^{N} \nabla f_{i}(x_{\tau}^{i})\|^{2}]
$$

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

$$
\sum_{k=1}^{K-1} \mathbb{E}\left[\left\| X_{k}(I - Q)\right\|_{\mathfrak{g}}^{2}\right] \leq CK + \frac{6N\eta^{2}}{(1 - \beta)^{2}(1 - \sqrt{\rho})} \sum_{k=1}^{K-1} \rho^{(k-1-\tau)} \mathbb{E}[\|\frac{1}{N} \sum_{i=1}^{N} \nabla f_{i}(x_{\tau}^{i})\|^{2}]
\leq CK + \frac{6N\eta^{2}}{(1 - \beta)^{2}(1 - \sqrt{\rho})} \sum_{k=0}^{K-1} \rho^{\frac{1}{2}(k-\frac{1}{2}-k)} \mathbb{E}[\|\frac{1}{N} \sum_{i=1}^{N} \nabla f_{i}(x_{\tau}^{i})\|^{2}]
$$

(31)
Where $C = \frac{14N\eta^2}{(1-\beta)^2} \left( \frac{\sigma^2}{(1-\sqrt{\beta})^2} + \frac{\zeta^2}{(1-\sqrt{\beta})^2} \right)$.

Dividing both sides by $N$:

$$
\sum_{k=0}^{K-1} \frac{1}{N} \mathbb{E} \left[ \left\| X_k (I - Q) \right\|_2^2 \right] \leq \frac{14\eta^2}{K} \left( \frac{\sigma^2}{(1-\sqrt{\beta})^2} + \frac{\zeta^2}{(1-\sqrt{\beta})^2} \right) + \frac{6\eta^2}{K} \sum_{k=0}^{K-1} \mathbb{E} \left[ \left\| \sum_{i=1}^{N} \nabla f_i(x_k^i) \right\|_2^2 \right]
$$

Hence, we obtain the following as the lemma.

$$
\sum_{k=0}^{K-1} \frac{1}{N} \sum_{i=1}^{N} \mathbb{E} \left[ \left\| x_k - x_k^i \right\|_2^2 \right] \leq \frac{14\eta^2}{K} \left( \sigma^2 + \zeta^2 \right) \sum_{k=0}^{K-1} \mathbb{E} \left[ \left\| \sum_{i=1}^{N} \nabla f_i(x_k^i) \right\|_2^2 \right] \quad \text{(33)}
$$

**A.2.2 Proof for Theorem.**

**Proof:** Using the L-smoothness properties for $F$ we have:

$$
\mathbb{E}[F(\tilde{z}_{k+1})] \leq \mathbb{E}[F(\tilde{z}_k)] + \mathbb{E}[\langle \nabla F(\tilde{z}_k), \tilde{z}_{k+1} - \tilde{z}_k \rangle] + \frac{L}{2} \mathbb{E}[\left\| \tilde{z}_{k+1} - \tilde{z}_k \right\|_2^2] \quad \text{(34)}
$$

Using Eq. (44) we have:

$$
\mathbb{E}[\langle \nabla F(\tilde{z}_k), \tilde{z}_{k+1} - \tilde{z}_k \rangle] = \frac{-\eta}{1-\beta} \mathbb{E}[\langle \nabla F(\tilde{z}_k), \frac{1}{N} \sum_{i=1}^{N} g_k^i \rangle] = \underbrace{\frac{-\eta}{1-\beta} \mathbb{E}[\langle \nabla F(\tilde{z}_k), \frac{1}{N} \sum_{i=1}^{N} g_k^i \rangle]}_{T_1} - \underbrace{\frac{\eta}{1-\beta} \mathbb{E}[\langle \nabla F(\tilde{z}_k), \frac{1}{N} \sum_{i=1}^{N} g_k^i \rangle]}_{T_2} \quad \text{(35)}
$$

We proceed by bounding $T_1$:

$$
\frac{-\eta}{1-\beta} \mathbb{E}[\langle \nabla F(\tilde{z}_k) - \nabla F(\tilde{x}_k), \frac{1}{N} \sum_{i=1}^{N} g_k^i \rangle] \leq \frac{(1-\beta)}{2L} \mathbb{E}[\left\| \nabla F(\tilde{z}_k) - \nabla F(\tilde{x}_k) \right\|_2^2] + \frac{\beta L \eta^2}{2(1-\beta)^3} \mathbb{E}[\left\| \frac{1}{N} \sum_{i=1}^{N} \tilde{g}_k^i \right\|_2^2] \quad \text{(36)}
$$

(i) holds as $(a, b) \leq \frac{1}{2} \left\| a \right\|_2^2 + \frac{1}{2} \left\| b \right\|_2^2$ where $a = \frac{1-\beta}{\sqrt{\beta} L} (\nabla F(\tilde{z}_k) - \nabla F(\tilde{x}_k))$ and $b = -\frac{\eta \sqrt{\beta} L}{1-\beta^2} \frac{1}{N} \sum_{i=1}^{N} \tilde{g}_k^i$. and (ii) uses the fact that $F$ is L-smooth.
We split the term $T_2$ as follows:

$$
\langle \nabla F (x_k), \frac{1}{N} \sum_{i=1}^{N} g_k^i \rangle = \langle \nabla F (x_k), \frac{1}{N} \sum_{i=1}^{N} (g_k^i - g_k^i + g_k^i) \rangle =
\langle \nabla F (x_k), \frac{1}{N} \sum_{i=1}^{N} (g_k^i - g_k^i) \rangle + \langle \nabla F (x_k), \frac{1}{N} \sum_{i=1}^{N} g_k^i \rangle
$$

(37)

Now, We first analyze the term $T_3$:

$$
\frac{-\eta}{(1 - \beta)} \mathbb{E}[\langle \nabla F (x_k), \frac{1}{N} \sum_{i=1}^{N} (g_k^i - g_k^i) \rangle] \leq \frac{(1 - \beta)}{2\beta L} \mathbb{E}[\|\nabla F (x_k)\|^2] + \frac{\eta^2 \beta L}{2(1 - \beta)^3} \mathbb{E}[\|\frac{1}{N} \sum_{i=1}^{N} (g_k^i - g_k^i)\|^2]
$$

(38)

This holds as $\langle a, b \rangle \leq \frac{1}{2} \|a\|^2 + \frac{1}{2} \|b\|^2$ where $a = \frac{-\sqrt{1 - \beta}}{\sqrt{\beta L}} \nabla F (x_k)$ and $b = \frac{\eta \sqrt{\beta L}}{(1 - \beta)^2} \frac{1}{N} \sum_{i=1}^{N} (g_k^i - g_k^i)$.

Analyzing the term $T_4$:

$$
\mathbb{E} \left[ \langle \nabla F (x_k), \frac{1}{N} \sum_{i=1}^{N} g_k^i \rangle \right] = \mathbb{E} \left[ \langle \nabla F (x_k), \frac{1}{N} \sum_{i=1}^{N} \nabla f_i (x_k^i) \rangle \right]
$$

(39)

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

$$
\langle \nabla F (x_k), \frac{1}{N} \sum_{i=1}^{N} \nabla f_i (x_k^i) \rangle = \frac{1}{2} \left( \|\nabla F (x_k)\|^2 + \|\frac{1}{N} \sum_{i=1}^{N} \nabla f_i (x_k^i)\|^2 - \|\nabla F (x_k)\| - \frac{1}{N} \sum_{i=1}^{N} \nabla f_i (x_k^i)\|^2 \right)
$$

(40)

(a) holds because $\|\nabla F (x_k)\| - \frac{1}{N} \sum_{i=1}^{N} \nabla f_i (x_k^i)\|^2 = \|\frac{1}{N} \sum_{i=1}^{N} \nabla f_i (x_k^i)\|^2 - \frac{1}{N} \sum_{i=1}^{N} \nabla f_i (x_k^i)\|^2 < \frac{1}{N} \sum_{i=1}^{N} \|\nabla f_i (x_k^i) - \nabla f_i (x_k^i)\|^2 \leq \frac{1}{N} \sum_{i=1}^{N} \|\nabla f_i (x_k^i)\|^2$.

Substituting (40) into (39), we have:

$$
\mathbb{E} \left[ \langle \nabla F (x_k), \frac{1}{N} \sum_{i=1}^{N} g_k^i \rangle \right] \geq \frac{1}{2} \left( \mathbb{E}[\|\nabla F (x_k)\|^2] + \mathbb{E}[\|\frac{1}{N} \sum_{i=1}^{N} \nabla f_i (x_k^i)\|^2] - L^2 \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}[\|\nabla f_i (x_k^i)\|^2] \right)
$$

(41)

Substituting (38), (41) into (37), we have

$$
-\frac{\eta}{1 - \beta} \mathbb{E}[\langle \nabla F (x_k), \frac{1}{N} \sum_{i=1}^{N} g_k^i \rangle] \leq \left( \frac{1 - \beta}{2\beta L} - \frac{\eta}{2(1 - \beta)} \right) \mathbb{E}[\|\nabla F (x_k)\|^2] + \frac{\eta^2 \beta L}{2(1 - \beta)^3} \mathbb{E}[\|\frac{1}{N} \sum_{i=1}^{N} (g_k^i - g_k^i)\|^2] - \frac{\eta L^2}{2(1 - \beta)} \mathbb{E}[\|\frac{1}{N} \sum_{i=1}^{N} \nabla f_i (x_k^i)\|^2] + \frac{\eta L^2}{2(1 - \beta)} \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}[\|\nabla f_i (x_k^i)\|^2]
$$

(42)
Now, finally substituting (42), (36) into (33), we have

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

\[
\mathbb{E}[\|\nabla F(\bar{x}_k)\|^2] - \frac{\eta}{2(1 - \beta)} \mathbb{E}[\|\frac{1}{N} \sum_{i=1}^N \nabla f_i(x^*_k)\|]^2 + \frac{\eta^2\beta L}{2(1 - \beta)^3} \mathbb{E}[\|\frac{1}{N} \sum_{i=1}^N (\mathbf{g}^*_i - \mathbf{g}_k)\|]^2] + \frac{\eta L^2}{2(1 - \beta)} \frac{1}{N} \sum_{i=1}^N \mathbb{E}[\|\bar{x}_k - x^*_k\|]^2
\]

Equation (44) states that:

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

Substituting (43), (44) in (34):

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

\[
\mathbb{E}[\|\nabla F(\bar{x}_k)\|^2] - \frac{\eta}{2(1 - \beta)} \mathbb{E}[\|\frac{1}{N} \sum_{i=1}^N \nabla f_i(x^*_k)\|]^2 + \frac{\eta^2\beta L}{2(1 - \beta)^3} \mathbb{E}[\|\frac{1}{N} \sum_{i=1}^N (\mathbf{g}^*_i - \mathbf{g}_k)\|]^2] + \frac{\eta L^2}{2(1 - \beta)} \frac{1}{N} \sum_{i=1}^N \mathbb{E}[\|\bar{x}_k - x^*_k\|]^2
\]

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

\[
\mathbb{E}[\|\nabla F(\bar{x}_k)\|^2] \leq \frac{1}{C_1} \left( \mathbb{E}[F(\bar{z}_k)] - \mathbb{E}[F(\bar{z}_{k+1})] \right) + \tilde{C}_2 \mathbb{E}[\|\frac{1}{N} \sum_{i=1}^N (\mathbf{g}^*_i)^2\|] + \tilde{C}_3 \mathbb{E}[\|\bar{z}_k - \bar{x}_k\|^2]
\]

\[
- \tilde{C}_6 \mathbb{E}[\|\frac{1}{N} \sum_{i=1}^N \nabla f_i(x^*_k)\|]^2 + \tilde{C}_4 \mathbb{E}[\|\frac{1}{N} \sum_{i=1}^N (\mathbf{g}^*_i - \mathbf{g}_k)\|]^2] + \tilde{C}_5 \frac{1}{N} \sum_{i=1}^N \mathbb{E}[\|\bar{x}_k - x^*_k\|]^2
\]

Where \(\tilde{C}_2 = \left(\frac{\beta L_\eta^2}{2(1 - \beta)^3} + \frac{\eta^2 L}{(1 - \beta)^2}\right) / C_1, \tilde{C}_3 = \frac{(1 - \beta)L}{2\beta} / C_1, \tilde{C}_4 = \frac{\eta^2\beta L}{2(1 - \beta)^3} / C_1, \tilde{C}_5 = \frac{\eta L^2}{2(1 - \beta)} / C_1, \tilde{C}_6 = \frac{\eta}{2(1 - \beta)} / C_1.

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

\[
\sum_{k=0}^{K-1} \mathbb{E}\left[\|\nabla F(\bar{x}_k)\|^2\right] \leq \frac{1}{C_1} \left( \mathbb{E}[F(\bar{z}_0)] - \mathbb{E}[F(\bar{z}_K)] \right) - \tilde{C}_6 \sum_{k=0}^{K-1} \mathbb{E}\left[\left\|\frac{1}{N} \sum_{i=1}^N \nabla f_i(x^*_k)\right\|^2\right] + \tilde{C}_2 \sum_{k=0}^{K-1} \mathbb{E}\left[\left\|\frac{1}{N} \sum_{i=1}^N \mathbf{g}^*_i\right\|^2\right]
\]

\[
+ \tilde{C}_3 \sum_{k=0}^{K-1} \mathbb{E}\left[\|\bar{z}_k - \bar{x}_k\|^2\right] + \tilde{C}_4 \sum_{k=0}^{K-1} \mathbb{E}\left[\left\|\frac{1}{N} \sum_{i=1}^N (\mathbf{g}^*_i - \mathbf{g}_k)\right\|^2\right] + \tilde{C}_5 \sum_{k=0}^{K-1} \frac{1}{N} \sum_{i=1}^N \mathbb{E}\left[\|\bar{x}_k - x^*_k\|^2\right]
\]

Substituting Lemma 1, Lemma 2, Lemma 3 and Equation 45 into the above equation, we have:
Now, we simplify the coefficients:

\[
\sum_{k=0}^{K-1} \mathbb{E} \left[ \| \nabla F(\bar{x}_k) \|^2 \right] \leq \frac{1}{C_1} \left( \mathbb{E} [F(\bar{x}_0)] - \mathbb{E} [F(\bar{x}_k)] \right) - \left( \tilde{C}_6 - 2\tilde{C}_2 - 2\tilde{C}_3 \frac{\eta^2 \beta^2}{(1-\beta)^4} - \tilde{C}_5 \frac{6\eta^2}{(1-\beta)^2(1-\sqrt{\rho})} \right) \\
\sum_{k=0}^{K-1} \mathbb{E} \left[ \left\| \frac{1}{N} \sum_{i=1}^{N} \nabla f_i(x_k) \right\|^2 \right] \leq \left( \tilde{C}_2 + \tilde{C}_3 \frac{\eta^2 \beta^2}{(1-\beta)^4} \right) \left( \frac{10\sigma^2}{N} + 8\varsigma^2 \right) K + 4\tilde{C}_4 \left( \frac{\sigma^2}{N} + \varsigma^2 \right) K \\
+ \tilde{C}_5 \frac{14\eta^2(\sigma^2 + \varsigma^2)}{(1-\beta)^2(1-\sqrt{\rho})^2} K
\]

Dividing both sides by \(K\) and considering the fact that \(\bar{z}_0 = \bar{x}_0\) and \(\tilde{C}_6 - 2\tilde{C}_2 - 2\tilde{C}_3 \frac{\eta^2 \beta^2}{(1-\beta)^4} - \tilde{C}_5 \frac{6\eta^2}{(1-\beta)^2(1-\sqrt{\rho})} \geq 0\):

\[
\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E} \left[ \| \nabla F(\bar{x}_k) \|^2 \right] \leq \frac{1}{C_1 K} \left( F(\bar{x}_0) - F^* \right) + \left( \tilde{C}_2 + \tilde{C}_3 \frac{\eta^2 \beta^2}{(1-\beta)^4} \right) \left( \frac{10\sigma^2}{N} + 8\varsigma^2 \right) \\
+ 4\tilde{C}_4 \left( \frac{\sigma^2}{N} + \varsigma^2 \right) + \tilde{C}_5 \frac{14\eta^2(\sigma^2 + \varsigma^2)}{(1-\beta)^2(1-\sqrt{\rho})^2} \leq \frac{1}{C_1 K} (F(\bar{x}_0) - F^*) + \left( 10\tilde{C}_2 + 10\tilde{C}_3 \frac{\eta^2 \beta^2}{(1-\beta)^4} + 4\tilde{C}_4 \right) \left( \frac{\sigma^2}{N} + \varsigma^2 \right) + \tilde{C}_5 \frac{14\eta^2(\sigma^2 + \varsigma^2)}{(1-\beta)^2(1-\sqrt{\rho})^2} \leq \frac{1}{C_1 K} (F(\bar{x}_0) - F^*) + \left( 10\tilde{C}_2 + 10\tilde{C}_3 \frac{\eta^2 \beta^2}{(1-\beta)^4} + 4\tilde{C}_4 \right) \left( \frac{\sigma^2}{N} + \varsigma^2 \right) + \tilde{C}_5 \frac{14\eta^2(\sigma^2 + \varsigma^2)}{(1-\beta)^2(1-\sqrt{\rho})^2}
\]

Now, we simplify the coefficients:

\[
10\tilde{C}_2 + 10\tilde{C}_3 \frac{\eta^2 \beta^2}{(1-\beta)^4} + 4\tilde{C}_4 \leq \frac{10\eta^2 L(\eta^2 + \beta)}{(1-\beta)^4} / C_1 = \frac{20\eta^2 L^2 \beta(\eta^2 + \beta)}{(1-\beta)^2(\eta^2 + \beta)} \\
\tilde{C}_5 \frac{14\eta^2}{(1-\beta)^2} = \frac{14\eta^2 L^3 \beta}{(1-\beta)^2(\eta^2 + \beta)}
\]

Now we redefine the coefficients:

\[
C_2 = \frac{20\eta^2 L^2 \beta(\eta^2 + \beta)}{(1-\beta)^2(\eta^2 + \beta)} \\
C_3 = \frac{14\eta^2 L^3 \beta}{(1-\beta)^2(\eta^2 + \beta)}
\]

Therefore we arrive at the bound given by the theorem. \(\square\)

\[
1 \sum_{k=0}^{K-1} \mathbb{E} \left[ \| \nabla F(\bar{x}_k) \|^2 \right] \leq \frac{1}{C_1 K} (F(\bar{x}_0) - F^*) + C_2 \left( \frac{\sigma^2}{N} + \varsigma^2 \right) + C_3 \left( \frac{\sigma^2 + \varsigma^2}{(1-\sqrt{\rho})^2} \right)
\]

(45)

### A.2.3 Discussion on the Step Size

In the proof of Theorem 1, we assumed the following \(\tilde{C}_6 - 2\tilde{C}_2 - 2\tilde{C}_3 \frac{\eta^2 \beta^2}{(1-\beta)^4} - \tilde{C}_5 \frac{6\eta^2}{(1-\beta)^2(1-\sqrt{\rho})} \geq 0\).

The above equation is true under the following conditions:

\[
(i) \ C_1 > 0 \\
(ii) \ 1 - \frac{4L}{(1-\beta)^2} \eta - \frac{6L^2}{(1-\beta)^2(1-\sqrt{\rho})} \eta^2 \geq 0
\]
Solving the first inequality gives us $\beta > 1 - \frac{L\eta}{2}(\sqrt{1 + \frac{4}{L\eta}} - 1)$.

We can simplify this further by using the Taylor series approximation i.e., $\frac{1}{L\eta} < \beta < \frac{4}{L}$.

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

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

Therefore, the step size $\eta$ is defined as

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

### A.2.4 Proof for Corollary 1

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

$$C_1 = \mathcal{O}(\frac{\sqrt{N}}{\sqrt{K}}), \quad \tilde{C}_2 = \mathcal{O}(\frac{\sqrt{N}}{\sqrt{K}}), \quad \tilde{C}_3 = \mathcal{O}(\frac{\sqrt{K}}{\sqrt{N}}), \quad \tilde{C}_4 = \mathcal{O}(\frac{\sqrt{N}}{\sqrt{K}}), \quad \tilde{C}_5 = \mathcal{O}(1),$$

This implies that

$$C_2 = \mathcal{O}(\frac{\sqrt{N}}{\sqrt{K}}), \quad C_3 = \mathcal{O}(\frac{N}{K}).$$

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

$$\frac{\mathcal{F}(x_0) - \mathcal{F}^*}{C_1 K} = \mathcal{O}\left(\frac{1}{\sqrt{NK}}\right).$$

For the remaining terms we have,

$$C_2 \frac{\sigma^2}{N} = \mathcal{O}\left(\frac{1}{\sqrt{NK}}\right), \quad C_2 \zeta^2 = \mathcal{O}\left(\frac{\sqrt{N}}{K}\right)$$

$$C_3 \frac{\sigma^2}{(1 - \sqrt{\rho})^2} = \mathcal{O}\left(\frac{N}{K}\right), \quad C_3 \frac{\zeta^2}{(1 - \sqrt{\rho})^2} = \mathcal{O}\left(\frac{N}{K^{1.5}}\right)$$

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

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

which suggests when $K$ is sufficiently large, NGM 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. 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 depicts the overview of the proposed NGM algorithm. Note that we do not have step 4 of the Figure in NGM algorithm.
Algorithm 2 Decentralized Peer-to-Peer Training (D-FPSGD with momentum)

**Input:** Each agent $i \in [1, N]$ initializes model weights $x_{i0}$, step size $\eta$, averaging rate $\gamma$, 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 $\text{Train}(\cdot)$ procedure

1. procedure $\text{Train}(\cdot)$
2. for $k = 0, 1, \ldots, K - 1$ do
3. $d_{ik} \sim D_i$ // sample data from training dataset.
4. $g_{ik} = \nabla_x f_i(d_{ik}, x_k^k)$ // compute the local gradients
5. $v_{ik} = \beta v_{ik(k-1)}^k - \eta g_{ik}^k$ // momentum step
6. $\bar{x}_k^k = x_k^k + v_k^k$ // update the model
7. $\text{SendReceive}(\bar{x}_k^k)$ // share model parameters with neighbors $N(i)$.
8. $x_{i(k+1)}^k = \bar{x}_k^k + \gamma \sum_{j \in N_i} (w_{ij} - I_{ij}) \bar{x}_j^k$ // gossip averaging step
9. end
10. return
Table 8, 9, 10 compare the proposed CompNGM to 1 bit by using scaled signed gradients. The error between the compressed and non-compressed gradient compensates 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.

Tables. 8 9 10 use the multistep-lr scheduler. We observe that both for iso-communication cost and iso-epochs, $\text{NGM}_{dv}$ performs better than CompNGM in terms of validation accuracy.

Algorithm 3 Compressed Neighborhood Gradient Mean (CompNGM)

**Input:** Each agent $i \in [1,N]$ initializes model weights $x_i^{(0)}$, step size $\eta$, averaging rate $\gamma$, dimension of the gradient $\delta$, mixing matrix $W = [w_{ij}]_{i,j \in [1,N]}$; $\text{NGM}$ mixing weight $\alpha$, and $I_{ij}$ are elements of $N \times N$ identity matrix.

Each agent simultaneously implements the Train( ) procedure

1. **procedure** Train( )
2. 
3. for $k=0,1,\ldots,K-1$ do
4. 
5. 
6. 
7. 
8. 
9. 
10. 
11. 
12. 
13. 
14. 
15. 
16. 
17. 
18. 
19. 
20. 
21. 
22. 
23. 

// sample data from training dataset
// compute the local self-gradients
// error compensation for self-gradients
// compress the compensated self-gradients
// update the error variable
// share model parameters with neighbors $N(i)$
// compute neighbors’ cross-gradients
// error compensation for cross-gradients
// compress the compensated cross-gradients
// update the error variable
// share compressed cross-gradients between $i$ and $j$
// modify local gradients
// momentum step
// update the model
// gossip averaging step

return
Table 8: 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.

<table>
<thead>
<tr>
<th>Method</th>
<th>Agents</th>
<th>5layer CNN</th>
<th>5layer CNN</th>
<th>VGG-11</th>
<th>ResNet-20</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Ring</td>
<td>Torus</td>
<td>Ring</td>
<td>Ring</td>
</tr>
<tr>
<td>CompCGA</td>
<td>5</td>
<td>82.00 ± 0.25</td>
<td>-</td>
<td>83.65 ± 0.41</td>
<td>86.73 ± 0.34</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>71.41 ± 0.94</td>
<td>75.95 ± 0.41</td>
<td>73.96 ± 0.31</td>
<td>73.63 ± 0.55</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>68.15 ± 0.79</td>
<td>71.71 ± 0.54</td>
<td>73.72 ± 2.74</td>
<td>66.34 ± 0.98</td>
</tr>
<tr>
<td>CompNGM (ours)</td>
<td>5</td>
<td>82.91 ± 0.21</td>
<td>-</td>
<td>84.03 ± 0.32</td>
<td>87.56 ± 0.34</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>74.36 ± 0.42</td>
<td>77.82 ± 0.20</td>
<td>77.02 ± 0.14</td>
<td>78.50 ± 0.98</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>71.46 ± 0.85</td>
<td>73.62 ± 0.74</td>
<td>73.76 ± 0.20</td>
<td>72.62 ± 0.71</td>
</tr>
</tbody>
</table>

Table 9: 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.

<table>
<thead>
<tr>
<th>Method</th>
<th>Agents</th>
<th>Fashion MNIST (LeNet-5)</th>
<th>CIFAR-100 (ResNet-20)</th>
<th>Imagenette(32x32) (MobileNet-V2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CompCGA</td>
<td>5</td>
<td>90.45 ± 0.34</td>
<td>55.74 ± 0.33</td>
<td>72.76 ± 0.44</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>81.62 ± 0.37</td>
<td>38.84 ± 0.54</td>
<td>59.92 ± 0.72</td>
</tr>
<tr>
<td>CompNGM (ours)</td>
<td>5</td>
<td>90.48 ± 0.19</td>
<td>57.51 ± 0.48</td>
<td>72.91 ± 1.06</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>83.38 ± 0.39</td>
<td>43.07 ± 0.32</td>
<td>61.91 ± 2.10</td>
</tr>
</tbody>
</table>

Table 10: 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.

<table>
<thead>
<tr>
<th>Method</th>
<th>AGNews-BERT&lt;sub&gt;mini&lt;/sub&gt; Agents = 4</th>
<th>AGNews-DistilBERT&lt;sub&gt;base&lt;/sub&gt; Agents = 8</th>
<th>Imagenette (224×224)-ResNet-18 Agents = 5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AGNews-BERT&lt;sub&gt;mini&lt;/sub&gt; Agents = 4</td>
<td>AGNews-DistilBERT&lt;sub&gt;base&lt;/sub&gt; Agents = 8</td>
<td>Imagenette (224×224)-ResNet-18 Agents = 5</td>
</tr>
<tr>
<td></td>
<td>AGNews-BERT&lt;sub&gt;mini&lt;/sub&gt; Agents = 4</td>
<td>AGNews-DistilBERT&lt;sub&gt;base&lt;/sub&gt; Agents = 8</td>
<td>Imagenette (224×224)-ResNet-18 Agents = 5</td>
</tr>
<tr>
<td>CompCGA</td>
<td>91.05 ± 0.29</td>
<td>88.91 ± 0.25</td>
<td>84.65 ± 0.57</td>
</tr>
<tr>
<td></td>
<td>93.54 ± 0.03</td>
<td>93.50 ± 0.16</td>
<td>62.93 ± 1.33</td>
</tr>
<tr>
<td>CompNGM (ours)</td>
<td>91.24 ± 0.43</td>
<td>89.01 ± 0.13</td>
<td>85.44 ± 0.10</td>
</tr>
</tbody>
</table>

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.

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 1000 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
Figure 6: Consensus error, validation loss, and validation accuracy against communication bits (top panel) and epochs (bottom panel) for training CIFAR-10 (non-IID) on ResNet-20 architecture over a ring topology with 10 agents.

the Imagenette dataset – a) a resized low resolution of $32 \times 32$ and, b) a full resolution of $224 \times 224$. 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., 2020a) 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., 2020a) 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 \cite{lecun1998gradient}, we use the standard architecture with 61,706 trainable parameters.

MobileNet-V2: We use the the standard MobileNet-V2 \cite{sandler2018mobilenetv2} architecture used for CIFAR dataset with 2.3M parameters except that BatchNorm+ReLU layers are replaced by EvoNorm-S0.

ResNet-18: For ResNet-18 \cite{he2016deep}, we use the standard architecture with 11M trainable parameters except that BatchNorm+ReLU layers are replaced by EvoNorm-S0.

BERT mini: For BERT mini \cite{devlin2018bert} 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 base: For DistilBERT base \cite{sanh2019distilbert} 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.

![Validation Loss](image)

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

Figure 7: 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 \(NGM_{dv}\) algorithm over IID and Non-IID data sampled from the CIFAR-10 dataset in Figure. 7(a) and 7(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. 7(c) shows the comparison of the convergence characteristics of the \(NGM_{dv}\) technique with the current state-of-the-art CGA algorithm. We observe that \(NGM_{dv}\) 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 \(NGM_{dv}\) and CGA techniques as shown in Figure. 8(a) and 8(b) respectively. We observe that both model variance and data variance bias for \(NGM_{dv}\) are significantly lower than CGA.

A.8 Resource Comparison

The communication cost, memory overhead, and compute overhead for various decentralized algorithms are shown in Table. The D-PSGD, QG-DSGDm, and \(NGM_{mx}\) algorithms require each agent to communicate model parameters of size \(m_s\) with all the \(n_i\) neighbors for the gossip averaging step and hence has a communication cost of \(O(n_i m_s)\). In the case of \(NGM_{dv}\) 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 \(O(2n_i m_s)\) and therefore an overhead of \(O(n_i m_s)\) compared to D-PSGD. Momentum Tracking also needs additional communication to communicate the tracking variable. \(CompNGM\) compresses the additional round of communication involved with \(NGM_{dv}\) from \(b\) bits to 1 bit. This reduces the communication overhead from \(O(n_i m_s)\) to \(O(n_i m_s/b)\).

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 \(O(n_i m_s)\) compared to D-PSGD. \(NGM_{dv}\) does not require any additional memory as it averages the received data-variant cross-gradients into the self-gradient.
Figure 8: 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.

Table 11: Memory overheads for various methods trained on different model architectures with CIFAR-10 dataset over undirected ring topology with 2 neighbors per agent.

<table>
<thead>
<tr>
<th>Architecture</th>
<th>CGA (MB)</th>
<th>NGM_{dv} (MB)</th>
<th>CompCGA (MB)</th>
<th>CompNGM (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 layer CNN</td>
<td>0.58</td>
<td>0</td>
<td>0.58</td>
<td>0.60</td>
</tr>
<tr>
<td>VGG-11</td>
<td>4.42</td>
<td>0</td>
<td>4.42</td>
<td>4.56</td>
</tr>
<tr>
<td>ResNet-20</td>
<td>2.28</td>
<td>0</td>
<td>2.28</td>
<td>2.15</td>
</tr>
</tbody>
</table>

The compressed version of NGM_{dv} requires an additional memory of $O(n_i m_s)$ to store the error variables $e_{ji}$ (refer Algorithm 3). CompCGA also needs to store error variables along with the projection matrix of compressed gradients. Therefore, CompCGA has a memory overhead of $O(n_i m_s + \frac{n_i m_s}{b})$. 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 11.

The computation of the cross-gradients (in both CGA and NGM algorithms) requires $n_i$ forward and backward passes through the deep learning model at each agent. This is reflected as $O(n_i B)$ in the compute overhead in Table 6. 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 47 - see Equation 5a in Esfandiari et al., 2021) optimization problem in an iterative manner:

$$\min_u \frac{1}{2} u^T G G^T u + g^T G^T u$$

subject to $u \geq 0$

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 $\geq 2$) order. In contrast, NGM involves a simple averaging step that requires $O(n_i m_s)$ 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 12, 13, 15, 14). The D-PSGD and NGM_{mv} have the lowest.
communication cost (1×). We emphasize that $NGM_{mv}$ outperforms D-PSGD in decentralized learning over label-wise non-IID data for the same communication cost. $NGM_{dv}$ and CGA have 2× communication overhead compared to D-PSGD where as $CompNGM$ and CompCGA have 1.03× communication overhead compared to D-PSGD. The compressed versions of $NGM_{dv}$ 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 $CompNGM$ reduces the communication cost by 32× compared to $NGM_{dv}$.

### Table 12: Communication costs per agent in GBs for experiments in Table 2 and 8

<table>
<thead>
<tr>
<th>Method</th>
<th>Agents</th>
<th>5layer CNN</th>
<th>5layer CNN</th>
<th>VGG-11</th>
<th>ResNet-20</th>
</tr>
</thead>
<tbody>
<tr>
<td>D-PSGD, QG-DSGDm and</td>
<td>5</td>
<td>17.75</td>
<td>-</td>
<td>270.64</td>
<td>127.19</td>
</tr>
<tr>
<td>$NGM_{mv}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Momentum Tracking,</td>
<td>5</td>
<td>35.48</td>
<td>-</td>
<td>541.05</td>
<td>254.27</td>
</tr>
<tr>
<td>CGA and $NGM_{dv}$ (ours)</td>
<td>10</td>
<td>17.81</td>
<td>26.72</td>
<td>271.50</td>
<td>127.59</td>
</tr>
<tr>
<td>CompCGA and $CompNGM$</td>
<td>5</td>
<td>18.31</td>
<td>17.95</td>
<td>136.72</td>
<td>64.25</td>
</tr>
<tr>
<td>and</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(ours)</td>
<td>10</td>
<td>9.20</td>
<td>13.79</td>
<td>140.10</td>
<td>65.84</td>
</tr>
<tr>
<td>(ours)</td>
<td>20</td>
<td>4.64</td>
<td>9.28</td>
<td>70.61</td>
<td>33.18</td>
</tr>
</tbody>
</table>

### Table 13: Communication costs per agent in GBs for experiments in Table 3 and 9

<table>
<thead>
<tr>
<th>Method</th>
<th>Agents</th>
<th>Fashion MNIST (LeNet-5)</th>
<th>CIFAR-100 (ResNet-20)</th>
<th>Imagenette (MobileNet-V2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>D-PSGD, QG-DSGDm and</td>
<td>5</td>
<td>17.25</td>
<td>103.74</td>
<td>103.12</td>
</tr>
<tr>
<td>$NGM_{mv}$</td>
<td>10</td>
<td>8.61</td>
<td>51.89</td>
<td>51.60</td>
</tr>
<tr>
<td>Momentum Tracking,</td>
<td>5</td>
<td>34.50</td>
<td>207.47</td>
<td>206.23</td>
</tr>
<tr>
<td>CGA and $NGM_{dv}$ (ours)</td>
<td>10</td>
<td>17.23</td>
<td>103.79</td>
<td>103.19</td>
</tr>
<tr>
<td>CompCGA and $CompNGM$</td>
<td>5</td>
<td>17.79</td>
<td>106.98</td>
<td>106.34</td>
</tr>
<tr>
<td>(ours)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(ours)</td>
<td>10</td>
<td>8.88</td>
<td>53.52</td>
<td>53.21</td>
</tr>
</tbody>
</table>

### Table 14: Communication costs per agent in GBs for experiments in Table 4 and 10 (right)

<table>
<thead>
<tr>
<th>Method</th>
<th>Ring topology</th>
<th>chain topology</th>
</tr>
</thead>
<tbody>
<tr>
<td>D-PSGD, QG-DSGDm and</td>
<td>501.98</td>
<td>401.59</td>
</tr>
<tr>
<td>$NGM_{mv}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Momentum Tracking,</td>
<td>1003.96</td>
<td>803.17</td>
</tr>
<tr>
<td>CGA and $NGM_{dv}$ (ours)</td>
<td>517.67</td>
<td>414.14</td>
</tr>
<tr>
<td>CompCGA and $CompNGM$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 15: Communication costs per agent in GBs for experiments in Table 4 and 10 (left)

<table>
<thead>
<tr>
<th>Method</th>
<th>$BERT_{mini}$</th>
<th>Distil$BERT_{base}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agents = 4</td>
<td>234.30</td>
<td>118.20</td>
</tr>
<tr>
<td>Agents = 8</td>
<td>486.59</td>
<td>236.40</td>
</tr>
</tbody>
</table>

### A.10 Hyper-parameters

This section presents the hyper-parameters for the experimental results presented in Sec. 5. All the experiments were run for three randomly chosen seeds. We decay the step size by 10x after 50% and 75% of the training for all the experiments except for figures. To generate the figures, the simulations are run for 300
epochs with the StepLR scheduler where the learning rate is decayed by 0.981 after every epoch. We have used a momentum of 0.9 for all QG-DSDGm, Momentum Tracking, CGA, and NGM_{dv} experiments. However, for D-PSGD and NGM_{mv}, the momentum is set to either 0.9 or 0.0 based on the validation accuracy. The averaging rate is always set to 1.0 for the Momentum Tracking algorithm.

**Hyper-parameters for CIFAR-10**: All the experiments that involve 5-layer (Table. 2) have stopping criteria set to 100 epochs. We decay the step size by $10^x$ in multiple steps at $50^{th}$ and $75^{th}$ epoch. All the experiments for the CIFAR-10 dataset trained on VGG-11 and ResNet-20 (Table. 2) have stopping criteria set to 200 epochs. We decay the step size by $10^x$ in multiple steps at $100^{th}$ and $150^{th}$ epoch. Table 16 presents the $\beta$, $\eta$, and $\gamma$ corresponding to the validation accuracy, 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 $10^x$ in multiple steps at $50^{th}$ and $75^{th}$ epoch. Table 17 presents the $\beta$, $\eta$, and $\gamma$ corresponding to the momentum, step size, and gossip averaging rate. For all the experiments related to Fashion MNIST and Imagenette (low resolution of $(32 \times 32)$), 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. For all the experiments, we use a mini-batch size of 32 per agent.

**Hyper-parameters used for Table. 4 (right)**: All the experiments in Table. 4 (right) have stopping criteria set to 100 epochs. We decay the step size by $10^x$ at $50^{th}$, $75^{th}$ epoch. Table 18 (right) presents the $\beta$, $\eta$, and $\gamma$ corresponding to the 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.

**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 $10^x$ at $2^{nd}$ epoch. Table 18 (left) presents the $\beta$, $\eta$, and $\gamma$ corresponding to the 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.

**Hyper-parameters used for Table. 7**: All the experiments in Table. 7 have stopping criteria set to 60 epochs. The initial learning rate is set to 0.01 and is decayed by $10^x$ at $20^{th}$, $40^{th}$, $54^{th}$ epoch. For all the experiments, we use a mini-batch size of 64 per agent. The values used for the learning rate, momentum coefficient, and averaging rate are presented in Table. 19.
Table 17: Hyper-parameters used for Table 3 and 9

<table>
<thead>
<tr>
<th>Method</th>
<th>Agents</th>
<th>Fashion MNIST</th>
<th>CIFAR-100</th>
<th>Imagenette</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(n)</td>
<td>(β, η, γ)</td>
<td>(β, η, γ)</td>
<td>(β, η, γ)</td>
</tr>
<tr>
<td>D-PSGD</td>
<td>5</td>
<td>(0.0, 0.01, 1.0)</td>
<td>(0.0, 0.1, 1.0)</td>
<td>(0.0, 0.1, 1.0)</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>(0.0, 0.01, 1.0)</td>
<td>(0.0, 0.1, 1.0)</td>
<td>(0.0, 0.1, 1.0)</td>
</tr>
<tr>
<td>QG-DSGDm</td>
<td>5</td>
<td>(0.9, 0.01, 1.0)</td>
<td>(0.9, 0.1, 1.0)</td>
<td>(0.9, 0.01, 1.0)</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>(0.9, 0.01, 1.0)</td>
<td>(0.9, 0.1, 1.0)</td>
<td>(0.9, 0.01, 1.0)</td>
</tr>
<tr>
<td>NGM_{m}</td>
<td>5</td>
<td>(0.9, 0.01, 1.0)</td>
<td>(0.0, 0.1, 1.0)</td>
<td>(0.0, 0.1, 1.0)</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>(0.9, 0.01, 1.0)</td>
<td>(0.0, 0.1, 1.0)</td>
<td>(0.0, 0.1, 1.0)</td>
</tr>
<tr>
<td>Momentum Tracking</td>
<td>5</td>
<td>(0.9, 0.01, 1.0)</td>
<td>(0.9, 0.1, 1.0)</td>
<td>(0.9, 0.01, 1.0)</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>(0.9, 0.01, 1.0)</td>
<td>(0.9, 0.1, 1.0)</td>
<td>(0.9, 0.01, 1.0)</td>
</tr>
<tr>
<td>CGA and NGM_{d}</td>
<td>5</td>
<td>(0.9, 0.01, 1.0)</td>
<td>(0.0, 0.1, 1.0)</td>
<td>(0.0, 0.01, 0.5)</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>(0.9, 0.01, 1.0)</td>
<td>(0.9, 0.1, 1.0)</td>
<td>(0.0, 0.01, 0.5)</td>
</tr>
<tr>
<td>CompCGA and CompNGM</td>
<td>5</td>
<td>(0.9, 0.01, 0.1)</td>
<td>(0.9, 0.01, 0.1)</td>
<td>(0.0, 0.01, 0.1)</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>(0.9, 0.01, 0.1)</td>
<td>(0.9, 0.01, 0.1)</td>
<td>(0.0, 0.01, 0.5)</td>
</tr>
</tbody>
</table>

Table 18: Hyper-parameters used for Table 4 and 10

<table>
<thead>
<tr>
<th>Method</th>
<th>BERT mini</th>
<th>DistilBERT mini</th>
<th>Imagenette (ResNet-18)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Agents = 4</td>
<td>Agents = 8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(β, η, γ)</td>
<td>(β, η, γ)</td>
<td>(β, η, γ)</td>
</tr>
<tr>
<td>D-PSGD</td>
<td>(0.0, 0.01, 1.0)</td>
<td>(0.0, 0.01, 1.0)</td>
<td>(0.0, 0.01, 1.0)</td>
</tr>
<tr>
<td>QG-DSGDm</td>
<td>(0.9, 0.01, 1.0)</td>
<td>(0.9, 0.01, 1.0)</td>
<td>(0.9, 0.01, 1.0)</td>
</tr>
<tr>
<td>NGM_{m} (ours)</td>
<td>(0.0, 0.01, 1.0)</td>
<td>(0.0, 0.01, 1.0)</td>
<td>(0.0, 0.01, 1.0)</td>
</tr>
<tr>
<td>Momentum Tracking</td>
<td>(0.9, 0.005, 1.0)</td>
<td>(0.9, 0.01, 1.0)</td>
<td>(0.9, 0.01, 1.0)</td>
</tr>
<tr>
<td>CGA</td>
<td>(0.9, 0.01, 0.5)</td>
<td>(0.9, 0.01, 0.5)</td>
<td>(0.9, 0.01, 0.5)</td>
</tr>
<tr>
<td>NGM_{d}</td>
<td>(0.9, 0.01, 0.5)</td>
<td>(0.9, 0.01, 0.5)</td>
<td>(0.9, 0.01, 0.5)</td>
</tr>
<tr>
<td>CompCGA</td>
<td>(0.9, 0.01, 0.5)</td>
<td>(0.9, 0.01, 0.5)</td>
<td>(0.9, 0.01, 0.5)</td>
</tr>
<tr>
<td>CompNGM</td>
<td>(0.9, 0.01, 0.5)</td>
<td>(0.9, 0.01, 0.5)</td>
<td>(0.9, 0.01, 0.5)</td>
</tr>
</tbody>
</table>

Table 19: hyperparameters for Table 7

<table>
<thead>
<tr>
<th>Trained for 60 epochs</th>
<th>D-PSGD</th>
<th>QG-DSGDm</th>
<th>NGM_{m}</th>
<th>NGM_{d}</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NLM</td>
<td>NLM</td>
<td>NLM</td>
<td>NLM</td>
</tr>
<tr>
<td>Learning rate (η)</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>Momentum coefficient (β)</td>
<td>0.0</td>
<td>0.9</td>
<td>0.9</td>
<td>0.9</td>
</tr>
<tr>
<td>Averaging rate (γ)</td>
<td>0.5</td>
<td>1.0</td>
<td>0.5</td>
<td>1.0</td>
</tr>
</tbody>
</table>