DeComFL: Federated Learning with Dimension-Free Communication

Zhe Li
Rochester Institute of Technology
Rochester, NY, 14623
zl4063@rit.edu

Bicheng Ying
Google Inc.
Los Angeles, CA, 90034
ybc@google.com

Zidong Liu
ComboCurve Inc.
Houston, TX, 77005
z.liu@combocurve.com

Chaosheng Dong
Amazon.com Inc.
Seattle, WA, 98109
chaosd@amazon.com

Haibo Yang
Rochester Institute of Technology
Rochester, NY, 14623
hbycis@rit.edu

Abstract

Federated Learning (FL) offers a promising framework for collaborative and privacy-preserving machine learning across distributed data sources. However, the substantial communication costs associated with FL significantly challenge its efficiency. Specifically, in each communication round, the communication costs scale linearly with the model's dimension, which presents a formidable obstacle, especially in large model scenarios. Despite various communication-efficient strategies, the intrinsic dimension-dependent communication cost remains a major bottleneck for current FL implementations. This paper proposes a novel dimension-free communication algorithm – DeComFL, which leverages the zeroth-order optimization techniques and reduces the communication cost from $\mathcal{O}(d)$ to $\mathcal{O}(1)$ by transmitting only a constant number of scalar values between clients and the server in each round, regardless of the dimension d of the model parameters. Theoretically, in non-convex functions, we prove that our algorithm achieves state-of-the-art rates, which show a linear speedup of the number of clients and local steps under standard assumptions. Empirical evaluations, encompassing large language model fine-tuning, demonstrate significant reductions in communication overhead. Notably, DeComFL achieves this by transmitting only around 1MB of data in total between the server and a client to fine-tune a model with billions of parameters. Our source code is available at https://github.com/ZidongLiu/DeComFL.

1 Introduction

Federated Learning (FL) is a promising distributed machine learning framework that enables a large number of clients to collaboratively train a model under the orchestration of a central server (Kairouz et al., 2021; McMahan et al., 2017). By allowing clients to train models locally without sharing their raw data, FL offers a privacy-preserving distributed learning paradigm. Thanks to these advantages, FL has become a popular learning paradigm used in many applications, such as healthcare(Xu et al., 2021) and edge devices(Nguyen et al., 2021; Wang et al., 2021), among others.

Despite its benefits, FL often encounters challenges to its efficiency due to *expensive communication costs*. Specifically, in one communication round, the server needs to broadcast the global model to all participating clients, and each of these clients is expected to transmit the newest local model to the server for global aggregation (McMahan et al., 2017). In other words, the communication costs for

International Workshop on Federated Foundation Models in Conjunction with NeurIPS 2024 (FL@FM-NeurIPS'24).

one participating client *scale linearly with the model's dimension*, presenting a prohibitively expensive communication overhead for FL systems, especially in large model and/or low communication speed scenarios. More specifically, on the one hand, foundation models in language and vision, such as GPT-3 (Brown et al., 2020), and other models (Bommasani et al., 2021), scale with billions of parameters, leading to tremendous total communication burden. For example, fine-tuning GPT-J-6B on 10 billion tokens with a batch size of 262K tokens across 4 machines would involve transferring 915.5 TB of data throughout the entire training process (Wang et al., 2023b). On the other hand, the typical communication speed for FL is several Mbps in wireless environments and up to several hundred Mbps in wired connections.

As a result, when training these models in a distributed/federated approach, communication becomes the key bottleneck in scaling. To achieve communication-efficient FL, several techniques have been developed, including lazy aggregation or multiple local update steps (McMahan et al., 2017), various compression techniques (Bernstein et al., 2018; Vogels et al., 2019; Yang et al., 2021; Wang et al., 2022; Hönig et al., 2022; Yi et al., 2024; Reisizadeh et al., 2020; Huang et al., 2023; Li & Li, 2023; Haddadpour et al., 2021), and client sampling strategies (Ribero & Vikalo, 2020). While these methods can reduce certain communication costs, their communication costs still scale linearly with the model's dimension for each participating client in one communication round. This intrinsic *dimension-dependent communication cost* remains a major bottleneck for current FL systems, particularly in the era of large deep learning models.

In this paper, we propose a novel FL approach to achieve dimension-free communication per round, leveraging zeroth-order optimization techniques (Nesterov & Spokoiny, 2017; Ghadimi & Lan, 2013; Liu et al., 2020). We exploit a unique property of zeroth-order gradients: they can be decomposed into a gradient scalar (magnitude) and a perturbation vector (direction). The gradient scalar is computed by using the finite difference of function values, while the perturbation vector can be generated identically across clients from a shared random seed. This decomposition is impossible in first-order methods. Therefore, instead of transmitting entire model parameters, we can *communicate gradient scalars and random seeds to reconstruct full gradients*, resulting in constant communication costs per round. However, in the FL setting, where clients collaborate to learn a global model, *simply transmitting seeds and gradient scalars to reconstruct gradients is insufficient to guarantee convergence to the desired global model*, as detailed in a later section. Hence, we propose a novel algorithm DeComFL, deviating from traditional FL appearance while achieving the same objective.

Our main results and contributions are summarized as follows:

- We propose DeComFL, a novel dimension-free communication in federated learning framework via zeroth order optimization. In each round, both the downlink (model pulling) and uplink (model uploading) communications involve transmitting only a constant number of scalar values between the participating clients and the server. This dramatically reduces the communication cost from $\mathcal{O}(d)$ to $\mathcal{O}(1)$ in both uplink and downlink, where d is the dimension of the model parameters.
- Theoretically, in non-convex functions, we prove that DeComFL achieves $\mathcal{O}\big(\frac{\sqrt{d}}{\sqrt{mPKR}}\big)$ under the standard conditions, where m is the number of participating clients in one communication round, P is the number of simultaneous perturbations, K is the number of local update steps and R is the number of communication rounds. This rate highlights the linear speedup in terms of the local update step, the number of perturbation and the clients.
- Comprehensive experiments on fine-tuning tasks demonstrate that DeComFL achieves comparable performance to existing algorithms while significantly reducing communication costs by several orders of magnitude. For instance, when fine-tuning an OPT-1.3B (Zhang et al., 2022) model in a FL setting, traditional methods require transmitting approximately 10.8 GB per round between each client and the server. In contrast, DeComFL requires only 1MB of total communication throughout the entire fine-tuning process.

2 Related Work

Communication-Efficient Federated Learning: Initially, (McMahan et al., 2017) proposed the FedAvg algorithm, which uses multiple local update steps to reduce the frequency of model transfers between the server and the client, thereby lowering the total communication cost. Since then, various techniques have been developed to further optimize communication efficiency, with most approaches involving compression methods. For instance, sparsification (Han et al., 2020; Li et al., 2020; Ozfatura et al., 2021; Wang et al., 2023a; Tang et al., 2022), quantization (Hönig et al., 2022; Huang et al., 2023; Haddadpour et al., 2021; Shlezinger et al., 2020; Jhunjhunwala et al., 2021; Bouzinis et al., 2023; Liu et al., 2023; Zakerinia et al., 2024), and low-rank approximations (Vogels et al., 2019; Martin & Mahoney, 2021). However, the communication cost per round between a client and the server remains dependent on the model dimension. Taking Top-$\Bbbk$ as an example (Stich et al., 2018), only the top $\Bbbk$ largest coordinates in the gradient vector are selected for communication. In theory, the convergence rate of Top-$\Bbbk$ depends on both the model dimension d and the hyper-parameter $\Bbbk$. In practice, the choice of $\Bbbk$ is linearly scaled with the model dimension d, i.e., $\Bbbk = c \times d$, where c is a constant such as 0.001 (Shi et al., 2019). Despite the success of these methods, the intrinsic dimension-dependent communication cost remains a major bottleneck for current FL systems, especially in the era of large deep learning models. In this work, our DeComFL achieves a constant $\mathcal{O}(1)$ communication cost for both uplink and downlink transmissions by utilizing zeroth-order optimization.

Zeroth-Order Optimization (ZOO): ZOO relies solely on function evaluations, making it ideal for scenarios where explicit gradient computation is impractical, expensive, or unreliable, such as in black-box optimization (Liu et al., 2020; Cai et al., 2021; Nikolakakis et al., 2022) and reinforcement learning (Liu et al., 2020; Jing et al., 2024; Li et al., 2021). Recently, ZOO has shown significant memory advantages in deep learning due to requiring only forward propagation (Malladi et al., 2023; Zhang et al., 2024). However, existing work has not fully exploited ZOO's potential to reduce communication costs in FL, as we propose in this work. For example, FedZO (Fang et al., 2022) applies zeroth-order (ZO) stochastic gradient estimation in the classic FedAvg algorithm, achieving a convergence rate of $\mathcal{O}(\frac{\sqrt{d}}{\sqrt{mKR}})$ in non-convex cases, but its communication complexity remains $\mathcal{O}(d)$ per round, the same as FedAvg. Similarly, BAFFLE (Feng et al., 2023) employs ZOO to achieve $\mathcal{O}(P)$ communication complexity in the uplink, but the downlink communication complexity remains $\mathcal{O}(d)$.

3 Dimension-Free Communication in Federated Learning

3.1 Preliminary of the Zeroth-Order Optimization and Federated Learning

As with most standard FL settings, we assume that there exist M clients in total in our FL system. Our goal is to minimize the global loss function f which can be formulated as,

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} f(\boldsymbol{x}) = \min_{\boldsymbol{x}\in\mathbb{R}^d} \frac{1}{M}\sum_{i=1}^{M} f_i(\boldsymbol{x}) \quad \text{where } f_i(\boldsymbol{x}) := \mathbb{E}\left[f(\boldsymbol{x};\xi_i)\right], \tag{1}$$

where $\boldsymbol{x}$ is a d-dimensional model parameter and f_i represents the loss function on client i. The loss function is the expectation of a stochastic loss function $f_i(\boldsymbol{x};\xi_i)$, where ξ_i is sampled from different local data distributions known as data heterogeneity in FL. The typical FL algorithm comprises three steps in each round: 1) The server initially samples a set of clients and sends the current global model to them. 2) Upon receiving the global model, each client performs multiple local updates based on this model and then transmits the updated local model back to the server. 3) The server aggregates all the returned local models from the clients and updates the global model accordingly.

More specifically, when applying the (stochastic) ZO method for the local update in the classical FedAvg (McMahan et al., 2017), we arrive at the FedZO (Fang et al., 2022). The main recursion of server model $\boldsymbol{x}_r$ and client models $\{\boldsymbol{x}_{i,r}^k\}$ can be summarized into the following forms:

$$\boldsymbol{x}_{i,r}^{1} = \boldsymbol{x}_r, \ \ \forall i \in C_r \qquad \text{(Pull Model)} \tag{2a}$$

$$\boldsymbol{x}_{i,r}^{k+1} = \boldsymbol{x}_{i,r}^{k} - \eta g_{i,r}^{k} \cdot \boldsymbol{z}_{i,r}^{k}, \quad k = 1, 2, \cdots, K, \qquad \text{(Local Update)} \tag{2b}$$

$$\boldsymbol{x}_{r+1} = \frac{1}{|C_r|}\sum_{i\in C_r} \boldsymbol{x}_{i,r}^{K}, \qquad \text{(Aggregate Model)} \tag{2c}$$

where we use the superscript k for the local update step, r for the communication round, i for the client index, and C_r for a set of sampled client indices, $\boldsymbol{z}_{i,r}^k$ typically for a random direction vector drawing either from either Gaussian or uniform ball distribution. $g_{i,r}^k$ is a gradient scalar calculated as

$$g_{i,r}^{k} = \frac{1}{\mu}\big(f_i(\boldsymbol{x}_{i,r}^k + \mu \boldsymbol{z}_{i,r}^k; \xi_{i,r}^k) - f_i(\boldsymbol{x}_{i,r}^k; \xi_{i,r}^k)\big), \tag{3}$$

where $\mu > 0$ is the smooth parameter. Intuitively, when μ is sufficiently small, the scalar $g_{i,r}^k$ approximates the gradient $\nabla f_i(\boldsymbol{x}_{i,r}^k; \xi_{i,r}^k)$ inner product with the random direction $\boldsymbol{z}_{i,r}^k$. There are other types of ZO gradient estimators, but in this paper, we only focus on this (3) form called Simultaneous Perturbation Stochastic Approximation (SPSA) (Spall, 1992) with the forward style. See (Nesterov & Spokoiny, 2017; Liu et al., 2020) for other forms and convergence properties.

When we examine the communication costs incurred between the client and the server within the framework outlined in equations (2a) to (2c), it becomes apparent that a vector of length $2d$ is transmitted during each round. Specifically, the first d-length vector is transmitted during the pull model step via the downlink, while the second d-length vector is sent after finishing all local update steps and before aggregating at the server via the uplink. In the era of LLMs, this $2d$ communication cost can be a huge burden or even prohibitively expensive for scenarios requiring low latency. This observation motivates us to design a novel FL framework wherein the lengths of the vectors communicated via both the uplink and downlink are independent of the model dimension d.

3.2 Eliminating Dimension-Dependent Communication in the Uplink

The equations (2a)-(2c) are merely a straightforward substitution of the first-order method with its zeroth-order counterpart. We can further exploit the zeroth-order property to lower the uplink communication cost. It is worth noting that the random vector $\boldsymbol{z}_{i,r}^k$ is generated using a pseudo-random number generator. Consequently, given a specific seed, $\boldsymbol{z}_{i,r}^k$ can be reproduced for a vector of any length. To exploit this property, we can reformulate (2b)-(2c) as

$$\boldsymbol{x}_{i,r}^{k+1} = \boldsymbol{x}_{i,r}^k - \eta g_{i,r}^k \cdot \boldsymbol{z}_r^k, \quad k = 1, 2, \cdots, K, \qquad \text{(Local Update)} \qquad (4a)$$

$$\boldsymbol{x}_{r+1} = \boldsymbol{x}_r - \eta \sum_{k=0}^{K-1} \left(\frac{1}{|C_r|} \sum_{i \in C_r} g_{i,r}^k \right) \cdot \boldsymbol{z}_r^k \qquad \text{(Aggregate Model Update)} \qquad (4b)$$

Two key modifications are introduced here: 1) $\boldsymbol{z}_{i,r}^k$ becomes $\boldsymbol{z}_r^k$; 2) Model aggregation is now computed using the model update (i.e., the difference observed during the local update step). The first modification is feasible if all clients agree on one common seed, and this modification paves the way for grouping the $g_{i,r}^k$ in the second modification. The second modification is crucial to save communication because only $\frac{1}{|C_r|} \sum_{i \in C_r} g_{i,r}^k$ is unknown to the server, which requires the transmission, but this quantity is merely a scalar!

With this seemingly minor adjustment, we significantly reduce the second d-length vector within the uplink, transforming it into a small constant quantity. However, this improvement is insufficient to achieve dimension-free communication due to the inherent requirements of the pull-model step.

3.3 Eliminating Dimension-Dependent Communication in the Downlink

The challenge remains to eliminate the full model transfer that occurs during the pull-model step in equation (2a). The solution is similar as the modification of model update in (4b), albeit with greater subtlety. **Presuming** that the client model $\boldsymbol{x}_{i,r'}^K$ is the same as the server model $\boldsymbol{x}_{r'}$, where r' is the last participated round, then the process of pulling the model from the server at the r-th round can be expressed as

$$\boldsymbol{x}_{i,r}^1 = \boldsymbol{x}_{i,r'}^K - \eta \sum_{j=r'}^{r-1} \sum_{k=1}^{K} g_j^k \cdot \boldsymbol{z}_j^k, \qquad \text{(Reconstruct Model)} \qquad (5)$$

where $g_j^k = \frac{1}{|C_r|} \sum_{i \in C_r} g_{i,j}^k$ is the average gradient scalar. A crucial observation is that, at the end of the local update, the client model $\boldsymbol{x}_{i,r'}^K$ deviates from the server model in equation (4b). This discrepancy poses a problem because, instead of directly transmitting the updated model, our approach relies on communicating the gradient via scalar values. One straightforward solution is to take a snapshot of the client model at the beginning of the local update and **revert** to it after the local update is completed. This ensures consistency because, as implied by equation (5), the client model $\boldsymbol{x}_{i,r}^1$ at the beginning of the local update is identical to the server model $\boldsymbol{x}_r$.

The data communicated between the server and clients in (5) is reduced to just a few gradient scalars and random seeds. That is dimension-free again! We refer to this step as "Reconstruct Model" rather than "Pulling Model" because it builds the model based on the local model instead of relying on the server model. In fact, due to this, we do not even need the server model $\boldsymbol{x}_r$ to be stored on the server.

3.4 DeComFL Algorithm

Algorithm 1 Dimension-Free Communication in Federated Learning (DeComFL) [Server-side]

```
 1: Initialize: $\{g_0^k\}_{k=1}^K$, learning rate $\eta$, local update steps $K$, communication rounds $R$.
 2: Allocate: memory for recording three states : 1) state set $\{t_i\}_{i=1}^N$ storing the last round that client $i$
    participated in, 2) seed set $\{\{s_r^k\}_{k=1}^K\}_{r=0}^{R-1}$, 3) gradient set$\{\{g_r^k\}_{k=1}^K\}_{r=0}^{R-1}$.
 3:
 4: for $r = 0, 1, \ldots, R-1$ do
 5:     Uniformly sample a client set $C_r$ with cardinality $m$ and sample $K$ seeds $\{s_r^k\}_{k=1}^K$
 6:     for each client $i \in C_r$ in parallel do
 7:         ClientRebuildModel($\{\{g_{r'}^k\}_{k=1}^K\}_{r'=t_i}^{r-1}, \{\{s_{r'}^k\}_{k=1}^K\}_{r'=t_i}^{r-1}$)    ▷ send $g$ and $s$ to client
 8:         $\{g_{i,r}^k\}_{k=1}^K =$ ClientZOLocalUpdate($\{s_r^k\}_{k=1}^K, r$)    ▷ send $s$ to client and receive $g$
 9:     end for
10:     Compute the global gradient scalars $\{g_r^k\}_{k=1}^K = \left\{ \frac{1}{|C_r|} \sum\limits_{i \in C_r} g_{i,r}^k \right\}_{k=1}^K$
11:     Store $\{g_r^k\}_{k=1}^K$ and $\{s_r^k\}_{k=1}^K$ and update the client's last update record $t_i = r$.
12:     $\boldsymbol{x}_{r+1} = \boldsymbol{x}_r - \eta \sum_{k=1}^K \cdot g_r^k \cdot \boldsymbol{z}_r^k$    ▷ This step is optional.
13: end for
```

Algorithm 2 Dimension-Free Communication in Federated Learning (DeComFL) [Client-side]

```
 1: Initialize: Maintain a local model $\boldsymbol{x}_{i,0}^1$ and standby until the following procedures triggered by server.
 2: procedure 1. ClientRebuildModel($\{\{g_{r'}^k\}_{k=1}^K\}_{r'=t_i}^{r-1}, \{\{s_{r'}^k\}_{k=1}^K\}_{r'=t_i}^{r-1}$)
 3:     for $r' = t_i, \ldots, r-1$ do    ▷ Equivalent to Pull-model step.
 4:         for $k = 1, \ldots, K$ do
 5:             Generate $\boldsymbol{z}_{r'}^k \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$ by random seed $s_{r'}^k$.
 6:             $\boldsymbol{x}_{i,r'}^{k+1} = \boldsymbol{x}_{i,r'}^k - \eta \sum_{r'=t_i}^{r-1} g_{r'}^k \cdot \boldsymbol{z}_{r'}^k$    ▷ $\boldsymbol{x}_{i,t_i}^0$ is the local model.
 7:         end for
 8:     end for
 9: end procedure
10:
11: procedure 2. ClientZOLocalUpdate($\{s_r^k\}_{k=1}^K, r$)    ▷ Can be replaced by other ZO methods.
12:     for $k = 1, \ldots, K$ do
13:         Generate $\boldsymbol{z}_r^k \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$ by random seed $s_r^k$
14:         $g_{i,r}^k = \frac{1}{\mu}\big(f_i(\boldsymbol{x}_{i,r}^k + \mu \boldsymbol{z}_r^k; \xi_{i,r}^k) - f_i(\boldsymbol{x}_{i,r}^k; \xi_{i,r}^k)\big)$    ▷ Forward difference style.
15:         $\boldsymbol{x}_{i,r}^{k+1} = \boldsymbol{x}_{i,r}^k - \eta g_{i,r}^k \cdot \boldsymbol{z}_r^k$    ▷ Standard ZO-SGD
16:     end for
17:     revert the local model back to $x_{i,r}^1$.
18:     Return $\{g_{i,r}^k\}_{k=1}^K$
19: end procedure
```

With the mathematical groundwork established, we are now prepared to present the DeComFL algorithm. A comprehensive description is provided in Algorithm 1 (from the server's perspective) and Algorithm 2 (from the client's perspective), with a high-level illustration depicted in Fig. 1.

As shown in the Algorithm tables, DeComFL deviates significantly from the traditional FL framework. We transform the standard three-step process (pulling, local update, and aggregation) as a new three-step approach: reconstructing, local update with revert, and global aggregate of gradient scalars. This revised framework necessitates several additional details to ensure the implementation of the algorithm in practice. To highlight a few:

a) **Allocation** (Line 2 in Alg. 1): The server is required to maintain some states to keep track of the client's last participation round, gradient scalar history and random seeds.

b) **ClientRebuildModel**: (Line 2-9 in Alg. 2) Assume that the current round is r-th round. For each sampled client, before executing the local update procedure, they need to reconstruct their own model because they may not participate in training in the $(r-1)$-th round. Hence, the

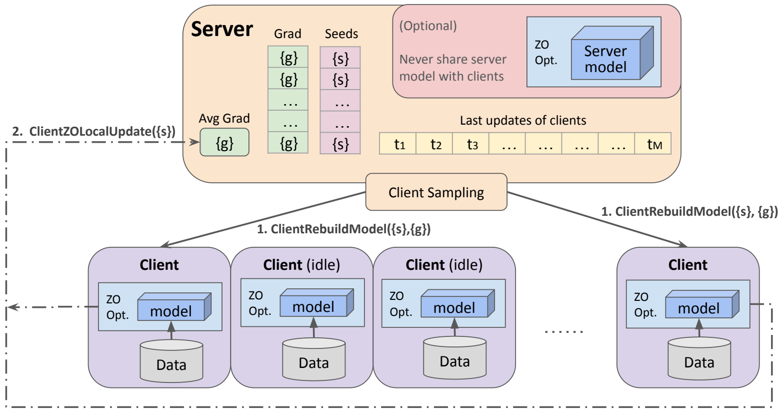

Figure 1: Illustration of DeComFL and components used in the server and clients.

clients need to fill the gap of model update between the current round and the last round where they participate in the training. It is corresponding to equation (5).

c) **ClientZOLocalUpdate**: (Line 11-19 in Alg. 2) After each sampled client finishes rebuilding its model, local update steps begin. Specifically, the client uses shared random seeds sent by the server to generate perturbations for each local update step. Each perturbation is used for one corresponding local update step. Then, they execute local update steps for K times to train their own local models by ZO-SGD. Finally, they revert the model as discussed in subsection 3.3 and send scalars $\{g_{i,r}^k\}_{k=1}^K$ back to the server. It is corresponding to equation (4a).

We emphasize that all information transmitted between the client and server consists of a few scalars and random seeds, representing "Dimension-Free" for DeComFL. For clarity, we only present the simplest case of DeComFL in the main paper. For instance, the presented algorithm uses a single perturbation vector. Extending it to incorporate multiple perturbation vectors is straightforward. *All subsequent theorems and experiments are based on this multi-perturbation version.* Moreover, as an algorithmic framework, DeComFL can be readily improved and extended into several variants. We defer this to Appendix A.

4 Convergence Analysis

This section presents the convergence rate of DeComFL under standard assumptions. Due to limited space, all proofs are deferred to Appendix C. First, we list the following standard assumptions 1, 2, 3, which are commonly used in the existing literature.

Assumption 1 (Unbiased Stochastic Gradient with Bounded Variance) *For any $r \geq 1$, we have*

$$\mathbb{E}\left[\nabla f_i(\boldsymbol{x}_r;\xi_r)\right] = \nabla f_i(\boldsymbol{x}_r) \text{ and } \mathbb{E}\left[\|\nabla f_i(\boldsymbol{x}_r;\xi_r) - \nabla f_i(\boldsymbol{x}_r)\|^2\right] \leq \sigma^2, \;\; \forall i.$$

Assumption 2 (Bounded Gradient Dissimilarity) *For any $i \in [M]$, $\|\nabla f(\boldsymbol{x}) - \nabla f_i(\boldsymbol{x})\|^2 \leq \sigma_G^2$.*

Assumption 3 (L-Lipschitz Continuous Gradient) *$f \in \mathcal{C}_L^{1,1}(\mathbb{R}^d)$, i.e., f is continuous and differentiable in first order and satisfies L-smooth condition:*

$$\|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\| \leq L\|\boldsymbol{x} - \boldsymbol{y}\|, \; \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d.$$

The convergence bound of DeComFL under the above standard assumptions is as follows:

Theorem 1 (Standard Convergence Bound of DeComFL) *Under Assumptions 1, 2 and 3, using Gaussian perturbations $\boldsymbol{z}_r^k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$, and if $\eta \leq \min\{\frac{mP}{24L(d+4)}, \frac{2P}{mKL(d+P+4)}, \frac{1}{mK^2L}, \frac{mP(d+3)^3}{2L[3mPK(d+3)^3+(d+6)^3]}\}$, the sequence of iterates generated by DeComFL satisfies:*

$$\frac{1}{R}\sum_{r=0}^{R-1}\mathbb{E}_r\|\nabla f(\boldsymbol{x}_r)\|^2 \leq \frac{4\big(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^\star)\big)}{KR\eta} + \left(\frac{72KL\eta}{m} + \frac{24(d+4)L\eta}{mP}\right)\sigma_G^2$$

$$+\frac{16L(d+4)\eta}{mP}\sigma^2 + 2\mu^2 L^2 (d+3)^3, \tag{6}$$

where $\boldsymbol{x}_r$ is the model parameter in the r-th round, P is the number of perturbations, $f(\boldsymbol{x}^\star)$ is the optimal loss value, K is the number of local update steps, R is the number of communication rounds, d is the dimension of model parameters, and m is the number of sampled clients in each round. ■

Remark 1 The right side of equation (6) comprises terms with distinct interpretations. The first term represents the decrease in the loss value at the initial point, the second term quantifies the impact of data heterogeneity, the third term arises from the stochastic gradient estimate, and the finite difference approximation introduces the fourth that is often negligible since μ is typically very small. The crucial terms are the middle two, depending on the dimension of the model parameters.

Corollary 1 (Standard Convergence Rate of DeComFL) *Further, based on theorem 1, supposing that $\mu \leq \frac{1}{(d+3)\sqrt{PKR}}$ and $\eta = \mathcal{O}\left(\frac{\sqrt{mP}}{\sqrt{dRK}}\right)$, the convergence rate of DeComFL is $\mathcal{O}\left(\frac{\sqrt{d}}{\sqrt{mPKR}}\right)$ when the algorithm runs with sufficient large communication round R.* ■

Remark 2 Both the number of local updates, K, and the number of perturbations, P, appear in the denominator of the final convergence rate, indicating a linear speedup with increasing client numbers and local steps. However, these parameters have opposing effects on the learning rate. A larger number of perturbations allows for a larger learning rate, while a larger number of local updates necessitates a smaller learning rate. This is intuitive, as more perturbations reduce variance between clients, while more local updates increase the dissimilarity between client models.

5 Experiments

DeComFL can achieve tremendous communication savings in LLM fine-tuning tasks. To further verify the DeComFL's effectiveness on LLMs, we execute fine-tuning tasks on a series of NLP datasets[1]. The models we used are OPT-125M and OPT-1.3B (Zhang et al., 2022). Due to the model size, we sample 2 clients from 8 clients to participate in each round to illustrate the core concept. In Table 1, we compare DeComFL with varying numbers of perturbations against MeZO (single agent setting) and FedZO (multi-agent setting) fine-tuning as baselines. All parameter settings and the definition of tasks are described in Sec B in Appendix. Although the number of rounds required for DeComFL convergence varies across different tasks (see appendix for details), the convergence consistently occurs within thousands of rounds, significantly fewer than the model's billions of dimensions and only slightly greater than that of first-order counterparts. This observation supports our low effective-rank assumption. The tables clearly demonstrate that DeComFL can match or even excel MeZO's performance. When using the same P, the performances of DeComFL and FedZO are almost the same, but the communication cost of DeComFL is dramatically lower than the one of FedZO. Besides, the results in Table 1 also supports our earlier claim that increasing p improves the performance. Lastly, the most important observation is that the communication costs for both model sizes are nearly identical, highlighting the dimension-free communication achieved by DeComFL!

Table 1: Test accuracy and communication cost on fine-tuning tasks

Model	Dataset \ Task	MeZO	FedZO with P=5	DeComFL with P = 5	DeComFL with P = 10
OPT-125M	SST-2	83.99%	84.11% (0.27 TB)[2]	84.02% (0.18 MB)	**85.08%** (0.36 MB)
	CB	72.49%	73.97% (0.09 TB)	74.28% (0.06 MB)	**75.00%** (0.12 MB)
	WSC	55.18%	59.43% (0.27 TB)	59.13% (0.18 MB)	**59.59%** (0.36 MB)
	WIC	53.25%	53.31% (0.27 TB)	53.28% (0.18 MB)	**53.38%** (0.36 MB)
	RTE	52.91%	53.42% (0.18 TB)	54.33% (0.12 MB)	**57.05%** (0.24 MB)
	BoolQ	61.46%	61.20% (0.18 TB)	61.36% (0.12 MB)	**61.60%** (0.24 MB)
OPT-1.3B	SST-2	90.23%	90.17% (1937.15 TB)	90.02% (0.12 MB)	**90.78%** (0.24 MB)
	CB	74.01%	74.41% (2905.73 TB)	74.40% (0.18 MB)	**75.71%** (0.36 MB)
	WSC	58.21%	59.95% (2905.73 TB)	60.41% (0.18 MB)	**64.16%** (0.36 MB)
	WIC	55.95%	56.06% (1937.15 TB)	55.97% (0.12 MB)	**56.14%** (0.24 MB)
	RTE	57.57%	58.88% (1452.86 TB)	59.42% (0.90 MB)	**60.89%** (1.80 MB)
	BoolQ	61.98%	62.01% (1452.86 TB)	62.17% (0.90 MB)	**62.50%** (1.80 MB)

[1] Loading and splitting datasets are based on `https://huggingface.co/datasets/super_glue`.

Acknowledgement

We sincerely thank Julie Huang from Google for her meaningful advice and help on the illustrations of this work.

References

Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signsgd: Compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, pp. 560–569. PMLR, 2018.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Pavlos S Bouzinis, Panagiotis D Diamantoulakis, and George K Karagiannidis. Wireless quantized federated learning: a joint computation and communication design. *IEEE Transactions on Communications*, 2023.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*, 2015.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

HanQin Cai, Yuchen Lou, Daniel McKenzie, and Wotao Yin. A zeroth-order block coordinate descent algorithm for huge-scale black-box optimization. In *International Conference on Machine Learning*, pp. 1193–1203. PMLR, 2021.

Xiangyi Chen, Sijia Liu, Kaidi Xu, Xingguo Li, Xue Lin, Mingyi Hong, and David Cox. Zo-adamm: Zeroth-order adaptive momentum method for black-box optimization. *Advances in neural information processing systems*, 32, 2019.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *NAACL*, 2019.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. The CommitmentBank: Investigating projection in naturally occurring discourse. 2019. To appear in proceedings of Sinn und Bedeutung 23. Data can be found at https://github.com/mcdm/CommitmentBank/.

Wenzhi Fang, Ziyi Yu, Yuning Jiang, Yuanming Shi, Colin N Jones, and Yong Zhou. Communication-efficient stochastic zeroth-order optimization for federated learning. *IEEE Transactions on Signal Processing*, 70: 5058–5073, 2022.

Haozhe Feng, Tianyu Pang, Chao Du, Wei Chen, Shuicheng Yan, and Min Lin. Does federated learning really need backpropagation? *arXiv preprint arXiv:2301.12195*, 2023.

Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

Farzin Haddadpour, Mohammad Mahdi Kamani, Aryan Mokhtari, and Mehrdad Mahdavi. Federated learning with compression: Unified analysis and sharp guarantees. In *International Conference on Artificial Intelligence and Statistics*, pp. 2350–2358. PMLR, 2021.

Pengchao Han, Shiqiang Wang, and Kin K Leung. Adaptive gradient sparsification for efficient federated learning: An online learning approach. In *2020 IEEE 40th international conference on distributed computing systems (ICDCS)*, pp. 300–310. IEEE, 2020.

Robert Hönig, Yiren Zhao, and Robert Mullins. Dadaquant: Doubly-adaptive quantization for communication-efficient federated learning. In *International Conference on Machine Learning*, pp. 8852–8866. PMLR, 2022.

Xinmeng Huang, Ping Li, and Xiaoyun Li. Stochastic controlled averaging for federated learning with communication compression. *arXiv preprint arXiv:2308.08165*, 2023.

[2] The value enclosed in parentheses represents the total bytes of the vector transferred between the server and a single client throughout the entire fine-tuning phase.

Divyansh Jhunjhunwala, Advait Gadhikar, Gauri Joshi, and Yonina C Eldar. Adaptive quantization of model updates for communication-efficient federated learning. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3110–3114. IEEE, 2021.

Gangshan Jing, He Bai, Jemin George, Aranya Chakrabortty, and Piyush K Sharma. Asynchronous distributed reinforcement learning for lqr control via zeroth-order block coordinate descent. *IEEE Transactions on Automatic Control*, 2024.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

Vid Kocijan, Thomas Lukasiewicz, Ernest Davis, Gary Marcus, and Leora Morgenstern. A review of winograd schema challenge datasets and approaches. *arXiv preprint arXiv:2004.13831*, 2020.

Shiqi Li, Qi Qi, Jingyu Wang, Haifeng Sun, Yujian Li, and F Richard Yu. Ggs: General gradient sparsification for federated learning in edge computing. In *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pp. 1–7. IEEE, 2020.

Xiaoyun Li and Ping Li. Analysis of error feedback in federated non-convex optimization with biased compression: Fast convergence and partial participation. In *International Conference on Machine Learning*, pp. 19638–19688. PMLR, 2023.

Yingying Li, Yujie Tang, Runyu Zhang, and Na Li. Distributed reinforcement learning for decentralized linear quadratic control: A derivative-free policy optimization approach. *IEEE Transactions on Automatic Control*, 67(12):6429–6444, 2021.

Heting Liu, Fang He, and Guohong Cao. Communication-efficient federated learning for heterogeneous edge devices based on adaptive gradient quantization. In *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*, pp. 1–10. IEEE, 2023.

Sijia Liu, Pin-Yu Chen, Xiangyi Chen, and Mingyi Hong. signsgd via zeroth-order oracle. In *International Conference on Learning Representations*, 2018a.

Sijia Liu, Bhavya Kailkhura, Pin-Yu Chen, Paishun Ting, Shiyu Chang, and Lisa Amini. Zeroth-order stochastic variance reduction for nonconvex optimization. *Advances in Neural Information Processing Systems*, 31, 2018b.

Sijia Liu, Pin-Yu Chen, Bhavya Kailkhura, Gaoyuan Zhang, Alfred O Hero III, and Pramod K Varshney. A primer on zeroth-order optimization in signal processing and machine learning: Principals, recent advances, and applications. *IEEE Signal Processing Magazine*, 37(5):43–54, 2020.

Sadhika Malladi, Tianyu Gao, Eshaan Nichani, Alex Damian, Jason D Lee, Danqi Chen, and Sanjeev Arora. Fine-tuning language models with just forward passes. *Advances in Neural Information Processing Systems*, 36:53038–53075, 2023.

Charles H Martin and Michael W Mahoney. Implicit self-regularization in deep neural networks: Evidence from random matrix theory and implications for learning. *Journal of Machine Learning Research*, 22(165):1–73, 2021.

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

Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

Yurii Nesterov and Vladimir Spokoiny. Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics*, 17:527–566, 2017.

Dinh C Nguyen, Ming Ding, Quoc-Viet Pham, Pubudu N Pathirana, Long Bao Le, Aruna Seneviratne, Jun Li, Dusit Niyato, and H Vincent Poor. Federated learning meets blockchain in edge computing: Opportunities and challenges. *IEEE Internet of Things Journal*, 8(16):12806–12825, 2021.

Konstantinos Nikolakakis, Farzin Haddadpour, Dionysis Kalogerias, and Amin Karbasi. Black-box generalization: Stability of zeroth-order learning. *Advances in Neural Information Processing Systems*, 35:31525–31541, 2022.

Emre Ozfatura, Kerem Ozfatura, and Deniz Gündüz. Time-correlated sparsification for communication-efficient federated learning. In *2021 IEEE International Symposium on Information Theory (ISIT)*, pp. 461–466. IEEE, 2021.

Mohammad Taher Pilehvar and Jose Camacho-Collados. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. *arXiv preprint arXiv:1808.09121*, 2018.

Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In *International conference on artificial intelligence and statistics*, pp. 2021–2031. PMLR, 2020.

Monica Ribero and Haris Vikalo. Communication-efficient federated learning via optimal client sampling. *arXiv preprint arXiv:2007.15197*, 2020.

Shaohuai Shi, Xiaowen Chu, Ka Chun Cheung, and Simon See. Understanding top-k sparsification in distributed deep learning. *arXiv preprint arXiv:1911.08772*, 2019.

Nir Shlezinger, Mingzhe Chen, Yonina C Eldar, H Vincent Poor, and Shuguang Cui. Federated learning with quantization constraints. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8851–8855. IEEE, 2020.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

James C Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE transactions on automatic control*, 37(3):332–341, 1992.

Sebastian U Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified sgd with memory. *Advances in neural information processing systems*, 31, 2018.

Zhenheng Tang, Shaohuai Shi, Bo Li, and Xiaowen Chu. Gossipfl: A decentralized federated learning framework with sparsified and adaptive communication. *IEEE Transactions on Parallel and Distributed Systems*, 34(3): 909–922, 2022.

Thijs Vogels, Sai Praneeth Karimireddy, and Martin Jaggi. Powersgd: Practical low-rank gradient compression for distributed optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

Bin Wang, Jun Fang, Hongbin Li, and Bing Zeng. Communication-efficient federated learning: A variance-reduced stochastic approach with adaptive sparsification. *IEEE Transactions on Signal Processing*, 2023a.

Jianyu Wang, Zachary Charles, Zheng Xu, Gauri Joshi, H Brendan McMahan, Maruan Al-Shedivat, Galen Andrew, Salman Avestimehr, Katharine Daly, Deepesh Data, et al. A field guide to federated optimization. *arXiv preprint arXiv:2107.06917*, 2021.

Jue Wang, Yucheng Lu, Binhang Yuan, Beidi Chen, Percy Liang, Christopher De Sa, Christopher Re, and Ce Zhang. Cocktailsgd: Fine-tuning foundation models over 500mbps networks. In *International Conference on Machine Learning*, pp. 36058–36076. PMLR, 2023b.

Yujia Wang, Lu Lin, and Jinghui Chen. Communication-efficient adaptive federated learning. In *International Conference on Machine Learning*, pp. 22802–22838. PMLR, 2022.

Jie Xu, Benjamin S Glicksberg, Chang Su, Peter Walker, Jiang Bian, and Fei Wang. Federated learning for healthcare informatics. *Journal of healthcare informatics research*, 5:1–19, 2021.

Haibo Yang, Jia Liu, and Elizabeth S Bentley. Cfedavg: achieving efficient communication and fast convergence in non-iid federated learning. In *2021 19th International Symposium on Modeling and Optimization in Mobile, Ad hoc, and Wireless Networks (WiOpt)*, pp. 1–8. IEEE, 2021.

Kai Yi, Georg Meinhardt, Laurent Condat, and Peter Richtárik. Fedcomloc: Communication-efficient distributed training of sparse and quantized models. *arXiv preprint arXiv:2403.09904*, 2024.

Hossein Zakerinia, Shayan Talaei, Giorgi Nadiradze, and Dan Alistarh. Communication-efficient federated learning with data and client heterogeneity. In *International Conference on Artificial Intelligence and Statistics*, pp. 3448–3456. PMLR, 2024.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Yihua Zhang, Pingzhi Li, Junyuan Hong, Jiaxiang Li, Yimeng Zhang, Wenqing Zheng, Pin-Yu Chen, Jason D Lee, Wotao Yin, Mingyi Hong, et al. Revisiting zeroth-order optimization for memory-efficient llm fine-tuning: A benchmark. *arXiv preprint arXiv:2402.11592*, 2024.

Table 2: Notations in this paper

Notation	Meaning
d	Model parameter dimension
m	Number of clients participating in each round
i, M	Index, number of clients
r, R	Index, number of communication round
p, P	Index, number of perturbations
k, K	Index, number of local update iterations
$\boldsymbol{x}_r$	Global model parameters in the r-th round
$\boldsymbol{x}_{i,r}^k$	Local model parameters in the k-th iteration and r-th round at client i
$\xi_{i,r}^k$	Data sample used in the k-th iteration and r-th round at client i
$g_{i,r}^k$	Zeroth-order gradient estimate scalar
$\boldsymbol{z}_r^k$	Perturbation in the k-th iteration and r-round
f	Global loss function
f_i	Local loss function at client i
C_r	Set of clients participating in r-th round

A Improvements and Variations of Algorithm Implementation

DeComFL discussed in the main paper is a quite general framework. In practice, several directions exist to extend further and improve the Algorithm 1.

A.1 Improve the Performance

One well-known issue of ZO methods is that the variance of stochastic gradient introduced by the random perturbation is so significant that it takes a longer time to converge and requires a much smaller learning rate than the FO counterparts.

Multiple Perturbations. One common variation in zeroth-order optimization is to use multiple perturbations. Suppose $\boldsymbol{z}_{r,p}^k$ is a Gaussian random vector generated for r-th round, k-th local update step and p-th perturbation (i.i.d. for any r, k, p). One step of SGD local update becomes:

$$\boldsymbol{x}_{i,r}^{k+1} = \boldsymbol{x}_{i,r}^k - \eta \sum_{p=1}^{P} g_{i,r,p}^k \cdot \boldsymbol{z}_{r,p}^k, \quad \text{where } g_{i,r,p}^k = \frac{1}{\mu}\big(f_i(\boldsymbol{x}_{i,r}^k + \mu \boldsymbol{z}_{r,p}^k; \xi_{i,r}^k) - f_i(\boldsymbol{x}_{i,r}^k; \xi_{i,r}^k)\big) \tag{7}$$

That is we perturb the model P times and calculate P gradient scalars $\{g_{i,r,p}^k\}$. The effective update then becomes a weighted average of these perturbations. While using multiple perturbations increases the communication cost linearly with P since all gradient scalars must be transmitted to the server for global aggregation, it can significantly reduce the variance of the stochastic gradient estimate so that decrease the required rounds. Also, unlike large mini-batch size, this *does not* increase the memory consumption since we calculate the corresponding gradient scalar $\{g_{i,r,p}^k\}_{p=1}^P$ sequentially.

Advanced Optimizers. We can use more advanced optimizers to improve the performance or accelerate the convergence of algorithm. For clarity, Algorithm 1 is based on vanilla ZO-SGD, but it should be straightforward to extend to other zeroth-order optimization methods, such as ZO-SignSGD (Liu et al., 2018a), ZO-SVRG (Liu et al., 2018b), ZO-Adam(Chen et al., 2019), etc. A full discussion and analysis of these optimizers are beyond the scope of this paper. However, to illustrate the necessary modifications, we present momentum SGD as an example:

$$\boldsymbol{m}_{i,r}^{k+1} = \beta \boldsymbol{m}_{i,r}^{k+1} + (1-\beta) g_{i,r}^k \cdot \boldsymbol{z}_r^k \tag{8}$$

$$\boldsymbol{x}_{i,r}^{k+1} = \boldsymbol{x}_{i,r}^k - \eta \cdot \boldsymbol{m}_{i,r}^{k+1} \tag{9}$$

where β is a momentum factor between 0 and 1. One pitfall is the optimizer is no longer stateless as ZO-SGD. Hence, after the local update step, we have to revert both the model parameter $\boldsymbol{x}_{i,r}^K$ but also the optimizer state $\boldsymbol{m}_{i,r}^K$. In the model reconstruction step, we maintain the optimizer states and model update at the same time as well.

A.2 Decrease the Memory Consumption

Reduce the Memory in the Server Side. One simple improvement is that the server no longer stores the model if there is no such necessity. This allows the server to function as a simple aggregator of scalar values. In a cloud service environment, this translates to utilizing less expensive CPU-only instances instead of costly GPU instances, resulting in substantial cost savings.

In our base algorithm (Algorithm 1), the server stores the entire history of selected random seeds and computed gradient scalars, potentially leading to significant memory consumption over many rounds. However, we can optimize this by recognizing that only information from the most recent round of each client is necessary. Since the server tracks each client's last updated round, a queue can efficiently manage this history. After clients complete their local updates, the server discards outdated information, minimizing memory usage. Further memory optimization can be achieved by having clients track and send their last participation round to the server, eliminating the need for the server to store this information.

Reduce the Memory in the Client Side. One significant memory consumption in the client side is we have to take a snapshot before the local update step. There is alternative more memory-efficient solution to ensuring the prerequisite for equation (5) is satisfied - namely, that the client model $\boldsymbol{x}_{i,r'}^{K}$ at the end of the local update is identical to the server model $\boldsymbol{x}_{r'}$. Subtracting (4b) from (4a)[3], then we get

$$\boldsymbol{x}_{r+1} - \boldsymbol{x}_{i,r}^{K} = \boldsymbol{x}_{r} - \boldsymbol{x}_{i,r}^{1} - \eta \sum_{k=1}^{K} \left(g_r^k - g_{i,r}^k\right) \cdot \boldsymbol{z}_r^k \tag{10}$$

Since $\boldsymbol{x}_r = \boldsymbol{x}_{i,r}^1$ after the reconstructing step, the quantity $\sum_{k=1}^{K} \left(g_r^k - g_{i,r}^k\right) \cdot \boldsymbol{z}_r^k$ represents the divergence between the local client model and the global server model. By compensating for this divergence, we can synchronize the two models. Crucially, this quantity can be generated using only gradient scalars and random seeds, eliminating the need to store a snapshot of the entire model. However, this technique is specific to SGD optimization.

A.3 Others Variants

Model Pulling for Excessive Lagging If Necessary. If a client remains unsampled for an extended period, the model pulling step requires retrieving all historical seeds and gradient scalars to update the model. This can be computationally demanding. In contrast, directly pulling the server's model has a fixed cost regardless of the client's lag time. This introduces a trade-off: if a client has limited computational resources and can tolerate communicating full model parameters, it might be preferable to simply pull the server's model.

Enhanced Privacy through Private Seed Shifting. Data privacy is paramount in FL. While our proposed DeComFL, like other FL algorithms, does not share local raw data with the server, we can further enhance privacy protection by ensuring that the server remains unaware of how the model evolves on the client side. Notice that even without direct access to local data, the server could potentially infer some information about local data distribution by comparing model updates between rounds. To address this, we introduce a simple and effective improvement: a private shift or function, known only to the clients, applies to the random seeds. Upon receiving a seed to generate a perturbation, the client first applies this private shift function to alter the seed. Since this shift is deterministic, it is easy to see that this modification does not affect the functionality of our algorithm while it prevents the server from reconstructing the model updates (This shift can be established via another server or a consensus protocol among clients). As a result, the random gradient scalars transmitted to the server cannot convey any information about the local data distribution, further enhancing privacy protection.

B Additional Experiment Details

Datasets for LLM Fine-tuning Tasks. We utilize a series of Natural Language Processing(NLP) datasets to execute fine-tuning tasks on LLMs, such as SST-2 (Socher et al., 2013; Wang et al., 2018)

[3] (4a) is K recursions. We expand the equation from K to 1 before the subtraction.

for the sentiment classification task, CB (De Marneffe et al., 2019) for hypothesis inference problem, WSC (Kocijan et al., 2020) for commonsense reasoning task, WIC (Pilehvar & Camacho-Collados, 2018) for word sense disambiguation task, RTE (Bowman et al., 2015) for natural language inference task, and BoolQ (Clark et al., 2019) for question answering.

Hyper-parameter Settings. In our FL system, for the experiments on LLMs, there are 8 clients in total, and in each communication round, only 2 clients are sampled to participate in the training. In Table 3, we show the specific parameter settings about learning rate and total communication rounds. For other shared parameters, we set smooth parameter $\mu = 1e-3$, Dirichlet concentration parameter $\alpha = 1$, local update step $K = 1$. For DeComFL's experiments, we set train batch size$= 32$ and test batch size$= 64$.

Table 3: Experiment setting of DeComFL

Model+Fine Tuning	Parameter \ Dataset	SST-2	CB	WSC	WIC	MultiRC	RTE	BoolQ
OPT-125M+FP[4]	Learning rate	5e-6	2e-6	5e-6	2e-7	5e-6	2e-6	2e-6
	Comm. rounds	3k	1k	3k	3k	2k	2k	2k
OPT-1.3B+FP	Learning rate	2e-6	5e-6	5e-6	2e-7	1e-5	2e-6	2e-6
	Comm. rounds	2k	3k	3k	2k	2k	1.5k	1.5k

C Theoretical Proof

C.1 Main Recursion

We can focus on the server-side model's evolution only because after the revert of the model (or synchronize step), all sampled clients are the same as server-side model. For other clients that are not sampled, they will sync to the server-side model after the reconstruction step so that we can virtually assume all clients and servers are iterated with the same server-side model.

The recursion of the server-side model can be written as

$$\boldsymbol{x}_{r+1} = \boldsymbol{x}_r - \eta \sum_{k=1}^{K} g_r^k \boldsymbol{z}_r^k = \boldsymbol{x}_r - \frac{\eta}{m} \sum_{k=1}^{K} \sum_{i \in C_r} g_{i,r}^k \boldsymbol{z}_r^k = \frac{1}{m} \sum_{i \in C_r} \underbrace{\left(\boldsymbol{x}_{i,r} - \eta \sum_{k=1}^{K} g_{i,r}^k \boldsymbol{z}_r^k \right)}_{:= \boldsymbol{x}_{i,r+1}}.$$

where we just denote $\boldsymbol{x}_{i,r} = \boldsymbol{x}_r$ for the client's model. It is now clear that our algorithm follows the same routine as the Federated Average framework that it combines the models after each client runs K local-update steps in their local model $\boldsymbol{x}_{i,r+1}$.

C.2 Lemmas for the Zeroth-Order Optimization

Before we present the proof of our main theorem, we first list several well-known lemmas about the zeroth-order optimization, which is the foundation for all following proofs.

Lemma 1 *(Nesterov, 2013) $f \in \mathcal{C}_L^{1,1}(\mathbb{R}^n)$ if it is differentiable and satisfies*

$$|f(\boldsymbol{y}) - f(\boldsymbol{x}) - \langle \nabla f(\boldsymbol{x}), \boldsymbol{y} - \boldsymbol{x} \rangle| \leq \frac{L}{2} \|\boldsymbol{y} - \boldsymbol{x}\|^2. \tag{11}$$

Lemma 2 *(Nesterov & Spokoiny, 2017; Ghadimi & Lan, 2013) We define a smooth approximation of objective function f_i as $f_i^{\mu}(\cdot)$ that can be formulated as*

$$f_i^{\mu}(\boldsymbol{x}) := \frac{1}{(2\pi)^{\frac{P}{2}}} \int f_i(\boldsymbol{x} + \mu \boldsymbol{z}) e^{-\frac{1}{2}\|\boldsymbol{z}\|^2} d\boldsymbol{z} = \mathbb{E}\left[f_i(\boldsymbol{x} + \mu \boldsymbol{z})\right], \tag{12}$$

where $\mu > 0$ is the smoothing parameter, and $\boldsymbol{z}$ is one n-dimensional standard Gaussian random vector. Then, for any $f_i \in \mathcal{C}_L^{1,1}$, the following statements hold.

[4] FP means full-parameter fine tuning in LLMs.

(a) The gradient of $f_i^\mu(\cdot)$ is L_μ-Lipschitz continuous where $L_\mu \leq L$. $\nabla f_i^\mu(\boldsymbol{x})$ can be shown as

$$\nabla f_i^\mu(\boldsymbol{x}) = \frac{1}{(2\pi)^{\frac{P}{2}}} \int \frac{f_i(\boldsymbol{x}+\mu\boldsymbol{z}) - f_i(\boldsymbol{x})}{\mu} \boldsymbol{z} e^{-\frac{1}{2}\|\boldsymbol{z}\|^2} d\boldsymbol{z}. \tag{13}$$

(b) For any $\boldsymbol{x} \in \mathbb{R}^d$,

$$|f_i^\mu(\boldsymbol{x}) - f_i(\boldsymbol{x})| \leq \frac{1}{2}\mu^2 L d \tag{14}$$

$$\|\nabla f_i^\mu(\boldsymbol{x}) - \nabla f_i(\boldsymbol{x})\| \leq \frac{1}{2}\mu L (d+3)^{\frac{3}{2}} \tag{15}$$

(c) For any $\boldsymbol{x} \in \mathbb{R}^d$,

$$\frac{1}{\mu^2}\mathbb{E}_{\boldsymbol{z}}\left[\Big(f_i(\boldsymbol{x}+\mu\boldsymbol{z}) - f_i(\boldsymbol{x})\Big)^2 \|\boldsymbol{z}\|^2\right] \leq \frac{\mu^2}{2}L^2(d+6)^3 + 2(d+4)\|\nabla f_i(\boldsymbol{x})\|^2 \tag{16}$$

Following form (15) and utilizing Jensen's inequality $\|a\|^2 \leq 2\|a-b\|^2 + 2\|b\|^2$, we have

$$\|\nabla f_i^\mu(\boldsymbol{x})\|^2 \leq 2\|\nabla f_i(\boldsymbol{x})\|^2 + \frac{1}{2}\mu^2 L^2 (d+3)^3, \tag{17}$$

$$\|\nabla f_i(\boldsymbol{x})\|^2 \leq 2\|\nabla f_i^\mu(\boldsymbol{x})\|^2 + \frac{1}{2}\mu^2 L^2 (d+3)^3. \tag{18}$$

Moreover, we denote $f_i^\mu(\boldsymbol{x}^*) := \min_{\boldsymbol{x}\in\mathbb{R}^d} f_i^\mu(\boldsymbol{x})$ and conclude $|f_i^\mu(\boldsymbol{x}^*) - f_i(\boldsymbol{x}^*)| \leq \frac{\mu^2 L d}{2}$ from (14). Then, we further conclude that

$$-\mu^2 L d \leq [f_i^\mu(\boldsymbol{x}) - f_i^\mu(\boldsymbol{x}^*)] - [f_i(\boldsymbol{x}) - f_i(\boldsymbol{x}^*)] \leq \mu^2 L d. \tag{19}$$

C.3 Proof of Theorem 1

Our main theorem is based on multiple perturbations. To light the notation, we first introduce $G_{i,r}^k$ that stands for the stochastic zeroth-order gradient estimate on $\boldsymbol{x}_{i,r}^k$ averaging over P-perturbation directions:

$$G_{i,r}^k := \frac{1}{P}\sum_{p=1}^P G_{i,r,p}^k = \frac{1}{P}\sum_{p=1}^P \frac{f_i(\boldsymbol{x}_{i,r}^k + \mu\boldsymbol{z}_{r,p}^k; \xi_{i,r}^k) - f_i(\boldsymbol{x}_{i,r}^k; \xi_{i,r}^k)}{\mu}\boldsymbol{z}_{r,p}^k = \frac{1}{P}\sum_{p=1}^P g_{i,r,p}^k \cdot \boldsymbol{z}_{r,p}^k \tag{20}$$

To begin with, we start with a few lemmas about the property about $G_{i,r}^k$.

Lemma 3 (Bounds on the Stochastic Zeroth-Order Gradient Variance) *The variance of the stochastic zeroth-order gradient $\mathbb{E}\|G_{i,r}^k - \nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)\|^2$ can be bounded by the true gradient $\|\nabla f(\boldsymbol{x}_r)\|^2$ on the starting point of round r, the local update distance $\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\|^2$ and several constants:*

$$\begin{aligned}\mathbb{E}_r\left\|G_{i,r}^k - \nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)\right\|^2 \leq & \frac{6(d+4)}{P}\|\nabla f(\boldsymbol{x}_r)\|^2 + \frac{6L^2(d+4)}{P}\mathbb{E}_r\left\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\right\|^2 + \frac{6(d+4)}{P}\sigma_G^2 \\ & + \frac{2(d+4)}{P}\sigma^2 + \frac{\mu^2 L^2 (d+6)^3}{2P}.\end{aligned} \tag{21}$$

Proof. For any independent and identically distributed random variables $\{\boldsymbol{y}_p\}_{p=1}^P$ with the mean $\bar{y}$, we know

$$\mathbb{E}\left\|\frac{1}{P}\sum_{p=1}^P \boldsymbol{y}_p - \bar{y}\right\|^2 = \frac{1}{P^2}\sum_{p=1}^P \mathbb{E}\left\|\boldsymbol{y}_p - \bar{y}\right\|^2 \tag{22}$$

Recall that $G_{i,r}^k = \frac{1}{P}\sum_{p=1}^P G_{i,r,p}^k$, $\mathbb{E}[G_{i,r,p}^k|\boldsymbol{x}_{i,r}^k] = \nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)$, and lemma 2 shows that

$$\mathbb{E}_r\left\|G_{i,r,p}^k - \nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)\right\|^2 \leq 2(d+4)\left\|\nabla f_i(\boldsymbol{x}_{i,r}^k; \xi_{i,r}^k)\right\|^2 + \frac{\mu^2}{2}L^2(d+6)^3.$$

Substituting $G_{i,r}^k$ and above properties into (22), we establish

$$\begin{aligned}\mathbb{E}_r \left\|G_{i,r}^k - \nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)\right\|^2 \leq & \frac{2(d+4)}{P}\mathbb{E}_r \left\|\nabla f_i(\boldsymbol{x}_{i,r}^k;\xi_{i,r}^k)\right\|^2 + \frac{\mu^2L^2(d+6)^3}{2P} \\ \leq & \frac{2(d+4)}{P}\mathbb{E}_r \left\|\nabla f_i(\boldsymbol{x}_{i,r}^k)\right\|^2 + \frac{2(d+4)}{P}\sigma^2 + \frac{\mu^2L^2(d+6)^3}{2P}\end{aligned}$$

Next, we bound the $\mathbb{E}_r \left\|\nabla f_i(\boldsymbol{x}_{i,r}^k)\right\|^2$ via the Jensen's inequality:

$$\begin{aligned}\mathbb{E}_r \left\|\nabla f_i(\boldsymbol{x}_{i,r}^k)\right\|^2 = & \mathbb{E}_r \left\|\nabla f_i(\boldsymbol{x}_{i,r}^k) - \nabla f_i(\boldsymbol{x}_r) + \nabla f_i(\boldsymbol{x}_r) - \nabla f(\boldsymbol{x}_r) + \nabla f(\boldsymbol{x}_r)\right\|^2 \\ \leq & 3\mathbb{E}_r \left\|\nabla f_i(\boldsymbol{x}_{i,r}^k) - \nabla f_i(\boldsymbol{x}_r)\right\|^2 + 3\mathbb{E}_r\|\nabla f_i(\boldsymbol{x}_r) - \nabla f(\boldsymbol{x}_r)\|^2 + 3\|\nabla f(\boldsymbol{x}_r)\|^2 \\ \leq & 3L^2\mathbb{E}_r \left\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\right\|^2 + 3\sigma_G^2 + 3\|\nabla f(\boldsymbol{x}_r)\|^2\end{aligned}$$

Lastly, plugging back, we finish the proof of lemma

$$\begin{aligned}\mathbb{E}_r \left\|G_{i,r}^k - \nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)\right\|^2 \leq & \frac{2(d+4)}{P}\Big(3L^2\mathbb{E}_r \left\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\right\|^2 + 3\sigma_G^2 + 3\|\nabla f(\boldsymbol{x}_r)\|^2\Big) \\ & + \frac{2(d+4)}{P}\sigma^2 + \frac{\mu^2L^2(d+6)^3}{2P} \\ = & \frac{6(d+4)}{P}\|\nabla f(\boldsymbol{x}_r)\|^2 + \frac{6L^2(d+4)}{P}\mathbb{E}_r \left\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\right\|^2 \\ & + \frac{6(d+4)}{P}\sigma_G^2 + \frac{2(d+4)}{P}\sigma^2 + \frac{\mu^2L^2(d+6)^3}{2P}\end{aligned}$$

■

Similarly, we can also bound the second-order moments of $\mathbb{E}_r\|G_{i,r}^k\|^2$ as follows.

Lemma 4 (Bounds on the Stochastic Zeroth-Order Gradient Second-Order Moments) *$\mathbb{E}\|G_{i,r}^k\|^2$ can be bounded by the true gradient $\|\nabla f(\boldsymbol{x}_r)\|^2$ on the starting point of round r, the local update distance $\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\|^2$ and several constants:*

$$\begin{aligned}\mathbb{E}_r \left\|G_{i,r}^k\right\|^2 \leq & \frac{6(d+P+4)}{P}\|\nabla f(\boldsymbol{x}_r)\|^2 + \frac{6L^2(d+P+4)}{P}\mathbb{E}_r \left\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\right\|^2 \\ & + \frac{6(d+P+4)}{P}\sigma_G^2 + \frac{2(d+4)}{P}\sigma^2 + \frac{\mu^2L^2(d+6)^3}{2P} + \frac{1}{2}\mu^2L^2(d+3)^3 \end{aligned} \tag{23}$$

Proof. Using Jensen's inequality, we know

$$\mathbb{E}_r \left\|G_{i,r}^k\right\|^2 = \mathbb{E}_r \left\|G_{i,r}^k - \nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)\right\|^2 + \mathbb{E}_r \left\|\nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)\right\|^2 \tag{24}$$

From Lemma 2, we have

$$\begin{aligned}& \mathbb{E}_r \left\|\nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)\right\|^2 \\ \leq & 2\mathbb{E}_r \left\|\nabla f_i(\boldsymbol{x}_{i,r}^k)\right\|^2 + \frac{1}{2}\mu^2L^2(d+3)^3 \\ = & 2\mathbb{E}_r \left\|\nabla f_i(\boldsymbol{x}_{i,r}^k) - \nabla f_i(\boldsymbol{x}_r) + \nabla f_i(\boldsymbol{x}_r) - \nabla f(\boldsymbol{x}_r) + \nabla f(\boldsymbol{x}_r)\right\|^2 + \frac{1}{2}\mu^2L^2(d+3)^3 \\ \leq & 6L^2\mathbb{E}_r \left\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\right\|^2 + 6\sigma_G^2 + 6\|\nabla f(\boldsymbol{x}_r)\|^2 + \frac{1}{2}\mu^2L^2(d+3)^3\end{aligned}$$

Combining with the result (21), we conclude the proof of lemma. ■

Furthermore, we denote $\chi_r = \mathbb{E}_r\left[\frac{1}{M}\sum_{i=1}^M\sum_{k=1}^K \left\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\right\|^2\right]$ for the local update steps, which is closely related to $\mathbb{E}_r \left\|G_{i,r}^k\right\|^2$. Using the previous lemma, we can easily establish the upper-bound on χ_r.

Lemma 5 (Bounds on Local Update Steps) *With Assumptions 1-3 and the learning rate satisfying $\eta \le \frac{2P}{\sqrt{6}LK\sqrt{d+P+4}}$, the local update distance χ_r satisfies*

$$
\begin{aligned}
\chi_r \le & \frac{6K^3(d+P+4)\eta^2}{P}\|\nabla f(\boldsymbol{x}_r)\|^2 + \frac{6K^3(d+P+4)\eta^2}{2P}\sigma_G^2 + \frac{2K^3(d+4)\eta^2}{P}\sigma^2 \\
& + \frac{\mu^2L^2K^3(d+6)^3\eta^2}{P} + \frac{1}{2}\mu^2L^2K^3(d+3)^3\eta^2
\end{aligned}
$$

Proof. Utilizing the relationship $\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r = \eta\sum_{\tau=1}^k G_{i,r}^\tau$, we have

$$
\begin{aligned}
\chi_r =& \mathbb{E}_r\left[\frac{\eta^2}{M}\sum_{i=1}^M\sum_{k=1}^K\left\|\sum_{\tau=1}^k G_{i,r}^\tau\right\|^2\right] \\
\le& \frac{\eta^2}{M}\sum_{i=1}^M\sum_{k=1}^K\sum_{\tau=1}^k k\mathbb{E}_r\left\|G_{i,r}^\tau\right\|^2 \\
\le& \frac{K^2\eta^2}{2M}\sum_{i=1}^M\sum_{k=1}^K\left\|G_{i,r}^k\right\|^2,
\end{aligned}
$$

where the last inequality holds since $\sum_{k=1}^K\sum_{\tau=1}^k kX_\tau = \sum_{\tau=1}^K(\sum_{k=\tau}^K k)X_\tau$. Substituting (23), we get

$$
\begin{aligned}
\chi_r \le& \frac{K^2\eta^2}{2M}\sum_{i=1}^M\sum_{k=1}^K\Bigg(\frac{6(d+P+4)}{P}\|\nabla f(\boldsymbol{x}_r)\|^2 + \frac{6L^2(d+P+4)}{P}\mathbb{E}\left\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\right\|^2 \\
& + \frac{6(d+P+4)}{P}\sigma_G^2 + \frac{2(d+4)}{P}\sigma^2 + \frac{\mu^2L^2(d+6)^3}{2P} + \frac{1}{2}\mu^2L^2(d+3)^3\Bigg) \qquad (25)
\end{aligned}
$$

Moving the term $\mathbb{E}\left\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\right\|^2$, which is χ_r again after the double summations, to the left hand side, we have

$$
\begin{aligned}
\left(1 - \frac{6L^2K^2(d+P+4)\eta^2}{2P}\right)\chi_r \le& \frac{3K^3(d+P+4)\eta^2}{P}\|\nabla f(\boldsymbol{x}_r)\|^2 + \frac{3K^3(d+P+4)\eta^2}{2P}\sigma_G^2 \\
& + \frac{K^3(d+4)\eta^2}{P}\sigma^2 + \frac{\mu^2L^2K^3(d+6)^3\eta^2}{2P} \\
& + \frac{1}{4}\mu^2L^2K^3(d+3)^3\eta^2 \qquad (26)
\end{aligned}
$$

When $\eta \le \frac{2P}{\sqrt{6}LK\sqrt{d+P+4}}$, the coefficient on the l.h.s. is larger than $\frac{1}{2}$. Plugging back, we complete the proof of this lemma. ■

Now we are ready to present the proof of main theorem with above lemmas. To ease the reference, we restate the theorem here again:

Theorem 2 (Restated; Standard Convergence Bound of DeComFL) *Under the assumptions 1, 2 and 3, supposing that the perturbation $\boldsymbol{z}_r^k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$, i.e., follows the Gaussian distribution, and the learning rate satisfies $\eta \le \min\{\frac{mP}{24L(d+4)}, \frac{2P}{mKL(d+P+4)}, \frac{1}{mK^2L}, \frac{mP(d+3)^3}{2L[3mPK(d+3)^3+(d+6)^3]}\}$, then it holds*

$$
\begin{aligned}
\frac{1}{R}\sum_{r=0}^{R-1}\mathbb{E}_r\|\nabla f(\boldsymbol{x}_r)\|^2 \le& \frac{4\big(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^\star)\big)}{KR\eta} + \left(\frac{72KL\eta}{m} + \frac{24(d+4)L\eta}{mP}\right)\sigma_G^2 \\
& + \frac{16L(d+4)\eta}{mP}\sigma^2 + 2\mu^2L^2(d+3)^3, \qquad (27)
\end{aligned}
$$

where $\boldsymbol{x}_r$ is the model parameter in the r-th round, P is the number of perturbations, $\boldsymbol{x}^*$ is the optimal point, K is the number of local update steps, R is the number of communication rounds, d is the dimension of model parameters, and m is the number of sampled clients in each round.

Proof. First, applying the $L-$Lipschitz smooth property on the global loss function f, we have

$$f(\boldsymbol{x}_{r+1}) \leq f(\boldsymbol{x}_r) + \langle \nabla f(\boldsymbol{x}_r), \boldsymbol{x}_{r+1} - \boldsymbol{x}_r \rangle + \frac{L}{2}\|\boldsymbol{x}_{r+1} - \boldsymbol{x}_r\|^2 \tag{28}$$

$$= f(\boldsymbol{x}_r) - \eta \left\langle \nabla f(\boldsymbol{x}_r), \frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K} G_{i,r}^k \right\rangle + \eta^2 \frac{L}{2}\left\| \frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K} G_{i,r}^k \right\|^2, \tag{29}$$

Taking conditional expectation $\mathbb{E}_r$ given the filtration $\boldsymbol{x}_r$ and information before round r, we obtain

$$\mathbb{E}_r[f(\boldsymbol{x}_{r+1})] \leq f(\boldsymbol{x}_r) \underbrace{-\eta \mathbb{E}_r \left\langle \nabla f(\boldsymbol{x}_r), \frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K} G_{i,r}^k \right\rangle}_{A_1} + \underbrace{\frac{L}{2}\eta^2 \mathbb{E}_r \left\| \frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K} G_{i,r}^k \right\|^2}_{A_2} \tag{30}$$

Observing $\mathbb{E}_{r,\xi}\left[\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\left(G_{i,r}^k - \nabla f_i^{\mu}(\boldsymbol{x}_{i,r}^k)\right)\right] = 0$, the cross product term A_1 satisfies

$$A_1 = -K\eta \mathbb{E}_r \left[\left\langle \nabla f(\boldsymbol{x}_r), \frac{1}{MK}\sum_{i=1}^{M}\sum_{k=1}^{K} \nabla f_i^{\mu}(\boldsymbol{x}_{i,r}^k) \right\rangle \right] \tag{31}$$

Utilizing the Parallelogram Identity, we know

$$\begin{aligned}
A_1 = & -\frac{1}{2}K\eta\|\nabla f(\boldsymbol{x}_r)\|^2 - \frac{1}{2}K\eta\mathbb{E}_r\left\|\frac{1}{MK}\sum_{i=1}^{M}\sum_{k=1}^{K}\nabla f_i^{\mu}(\boldsymbol{x}_{i,r}^k)\right\|^2 \\
& + \frac{1}{2}K\eta\mathbb{E}_r\left\|\frac{1}{MK}\sum_{i=1}^{M}\sum_{k=1}^{K}\left[\nabla f_i^{\mu}(\boldsymbol{x}_{i,r}^k) - \nabla f_i(\boldsymbol{x}_r)\right]\right\|^2 \\
\leq & -\frac{1}{2}K\eta\|\nabla f(\boldsymbol{x}_r)\|^2 - \frac{1}{2}K\eta\mathbb{E}_r\left\|\frac{1}{MK}\sum_{i=1}^{M}\sum_{k=1}^{K}\nabla f_i^{\mu}(\boldsymbol{x}_{i,r}^k)\right\|^2 \\
& + \frac{1}{2}K\eta\frac{1}{MK}\mathbb{E}_r\sum_{i=1}^{M}\sum_{k=1}^{K}\left\|\nabla f_i^{\mu}(\boldsymbol{x}_{i,r}^k) - \nabla f_i(\boldsymbol{x}_r)\right\|^2 \\
\leq & -\frac{1}{2}K\eta\|\nabla f(\boldsymbol{x}_r)\|^2 - \frac{1}{2}K\eta\mathbb{E}_r\left\|\frac{1}{MK}\sum_{i=1}^{M}\sum_{k=1}^{K}\nabla f_i^{\mu}(\boldsymbol{x}_{i,r}^k)\right\|^2 \\
& + \frac{\eta}{M}\mathbb{E}_r\sum_{i=1}^{M}\sum_{k=1}^{K}\left\|\nabla f_i^{\mu}(\boldsymbol{x}_{i,r}^k) - \nabla f_i(\boldsymbol{x}_{i,r}^k)\right\|^2 + \frac{\eta}{M}\mathbb{E}_r\sum_{i=1}^{M}\sum_{k=1}^{K}\|\nabla f_i(\boldsymbol{x}_{i,r}^k) - \nabla f_i(\boldsymbol{x}_r)\|^2 \\
\leq & -\frac{1}{2}K\eta\|\nabla f(\boldsymbol{x}_r)\|^2 - \frac{1}{2}K\eta\mathbb{E}_r\left\|\frac{1}{MK}\sum_{i=1}^{M}\sum_{k=1}^{K}\nabla f_i^{\mu}(\boldsymbol{x}_{i,r}^k)\right\|^2 + \frac{1}{4}\mu^2 KL^2(d+3)^3\eta \\
& + \frac{L^2\eta}{M}\sum_{k=1}^{K}\sum_{i=1}^{M}\mathbb{E}_r\left\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\right\|^2,
\end{aligned}$$

where we utilize Jensen's Inequality in the first two inequalities and apply L-smoothness and 15 to get the last inequality. Next, we focus on the quadratic term A_2.

$$\begin{aligned}
A_2 = & \frac{L}{2}\eta^2\mathbb{E}_r\left\|\frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K}G_{i,r}^k\right\|^2 \\
\leq & L\eta^2\mathbb{E}_r\left\|\frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K}[G_{i,r}^k - \nabla f_i^{\mu}(\boldsymbol{x}_{i,r}^k)]\right\|^2 + L\eta^2\mathbb{E}_r\left\|\frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K}\nabla f_i^{\mu}(\boldsymbol{x}_{i,r}^k)\right\|^2
\end{aligned}$$

$$
\begin{aligned}
&= \frac{L\eta^2}{mM}\sum_{i=1}^{M}\sum_{k=1}^{K}\mathbb{E}_r\left\|G_{i,r}^k-\nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)\right\|^2+L\eta^2\mathbb{E}_r\left\|\frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K}\nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)\right\|^2\\
&\le \frac{6L^3(d+4)\eta^2}{mMP}\sum_{i=1}^{M}\sum_{k=1}^{K}\mathbb{E}_r\left\|\boldsymbol{x}_{i,r}^k-\boldsymbol{x}_r\right\|^2+\frac{KL\eta^2}{m}\left[\frac{6(d+4)}{P}\|\nabla f(\boldsymbol{x}_r)\|^2+\frac{6(d+4)}{P}\sigma_G^2\right]\\
&\quad+\frac{KL\eta^2}{m}\left[\frac{2(d+4)}{P}\sigma^2+\frac{\mu^2L^2(d+6)^3}{2P}\right]+\underbrace{L\eta^2\mathbb{E}_r\left\|\frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K}\nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)\right\|^2}_{A_3},
\end{aligned}
\tag{32}
$$

where we applied Jensen's inequality in the first inequality; the second equality holds since each term $\left[G_{i,r}^k-\nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)\right]$ is zero mean and independent to each other; the last inequality utilized the Lemma 3. For A_3, it can be bounded as follows

$$
\begin{aligned}
A_3 =& L\eta^2\mathbb{E}_r\left\|\frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K}\nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)\right\|^2\\
=& L\eta^2\mathbb{E}_r\left\|\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)\right\|^2+L\eta^2\underbrace{\mathbb{E}_r\left\|\frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K}\nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)-\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)\right\|^2}_{A_4}
\end{aligned}
\tag{33}
$$

Continuing bounding the A_4 term, we have

$$
A_4 = \mathbb{E}_r\left\|\frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K}\nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)-\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)\right\|^2 \tag{34}
$$

$$
\begin{aligned}
&\overset{(a)}{\le} 3\mathbb{E}_r\left\|\frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K}\left[\nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)-\nabla f_i(\boldsymbol{x}_{i,r}^k)\right]\right\|^2\\
&\qquad+3\mathbb{E}_r\left\|\frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K}\nabla f_i(\boldsymbol{x}_{i,r}^k)-\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\nabla f_i(\boldsymbol{x}_{i,r}^k)\right\|^2\\
&\qquad+3\mathbb{E}_r\left\|\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\left[\nabla f_i(\boldsymbol{x}_{i,r}^k)-\nabla f_i^\mu(\boldsymbol{x}_{i,r}^k)\right]\right\|^2\\
&\overset{(b)}{\le}\frac{3}{2}\mu^2K^2L^2(d+3)^3+\underbrace{3\mathbb{E}_r\left\|\frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K}\nabla f_i(\boldsymbol{x}_{i,r}^k)-\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\nabla f_i(\boldsymbol{x}_{i,r}^k)\right\|^2}_{A_5},
\end{aligned}
\tag{35}
$$

where in the step (a), we plus and minus the $\frac{1}{m}\sum\limits_{i\in C_r}\sum\limits_{k=1}^{K}\nabla f_i(\boldsymbol{x}_{i,r}^k)$ and $\frac{1}{M}\sum\limits_{i=1}^{M}\sum\limits_{k=1}^{K}\nabla f_i(\boldsymbol{x}_{i,r}^k)$ then applies Jensen's inequality; in the step (b), we restore to the lemma 2 on the first and last terms. Next, we use the similar trick to bound A_5:

$$
\begin{aligned}
A_5 =& 3\mathbb{E}_r\Bigg\|\frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K}\nabla f_i(\boldsymbol{x}_{i,r}^k)-\frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K}\nabla f_i(\boldsymbol{x}_r)+\frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K}\nabla f_i(\boldsymbol{x}_r)\\
&\qquad-\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\nabla f_i(\boldsymbol{x}_r)+\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\nabla f_i(\boldsymbol{x}_r)-\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\nabla f_i(\boldsymbol{x}_{i,r}^k)\Bigg\|^2\\
\overset{(a)}{\le}& 9\mathbb{E}_r\left\|\frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K}\left[\nabla f_i(\boldsymbol{x}_{i,r}^k)-\nabla f_i(\boldsymbol{x}_r)\right]\right\|^2
\end{aligned}
$$

$$
\begin{aligned}
&+ 9\mathbb{E}_r \left\| \frac{1}{m}\sum_{i\in C_r}\sum_{k=1}^{K} \nabla f_i(\boldsymbol{x}_r) - \frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K} \nabla f_i(\boldsymbol{x}_r) \right\|^2 \\
&+ 9\mathbb{E}_r \left\| \frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K} \left[\nabla f_i(\boldsymbol{x}_r) - \nabla f_i(\boldsymbol{x}_{i,r}^k)\right] \right\|^2 \\
\overset{(b)}{\leq}& 18KL^2\mathbb{E}_r\left[\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\left\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\right\|^2\right] + 9K^2\mathbb{E}_r\left\|\frac{1}{m}\sum_{i\in C_r}\left[\nabla f_i(\boldsymbol{x}_r) - \nabla f(\boldsymbol{x}_r)\right]\right\|^2 \\
\overset{(c)}{=}& 18KL^2\mathbb{E}_r\left[\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\left\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\right\|^2\right] + \frac{9K^2}{m^2}\mathbb{E}_r\sum_{i\in C_r}\left\|\nabla f_i(\boldsymbol{x}_r) - \nabla f(\boldsymbol{x}_r)\right\|^2 \\
\overset{(d)}{\leq}& 18KL^2\mathbb{E}_r\left[\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\left\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\right\|^2\right] + \frac{9K^2}{m}\sigma_G^2,
\end{aligned}
$$

where step (a) applies Jensen's inequality; step (b) utilizes the $L-$Lipschitze condition; the equality in step (c) holds because each term $\left[\nabla f_i(\boldsymbol{x}_r) - \nabla f_i(\boldsymbol{x}_{i,r}^k)\right]$ is independent and zero-mean; step (d) results from the data heterogeneous assumption.

Plugging A_4 and A_5 into A_3, we establish

$$
\begin{aligned}
A_3 \leq & L\eta^2\mathbb{E}_r\left\|\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\nabla f_i^{\mu}(\boldsymbol{x}_{i,r}^k)\right\|^2 + \frac{3}{2}\mu^2K^2L^3(d+3)^3\eta^2 \\
& + 18KL^3\eta^2\mathbb{E}_r\left[\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\left\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\right\|^2\right] + \frac{9K^2L\eta^2}{m}\sigma_G^2
\end{aligned} \tag{36}
$$

Now, we are ready to put A_3 back to A_2 and group the terms

$$
\begin{aligned}
A_2 \leq & \frac{6KL(d+4)\eta^2}{mP}\|\nabla f(\boldsymbol{x}_r)\|^2 + \frac{6L^3(d+4)\eta^2}{mMP}\sum_{i=1}^{M}\sum_{k=1}^{K}\mathbb{E}_r\left\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\right\|^2 \\
& + \frac{6KL(d+4)\eta^2}{mP}\sigma_G^2 + \frac{2KL(d+4)\eta^2}{mP}\sigma^2 + \frac{\mu^2KL^3(d+6)^3\eta^2}{2mP} \\
& + 18KL^3\eta^2\mathbb{E}_r\left[\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\left\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\right\|^2\right] + \frac{3}{2}\mu^2K^2L^3(d+3)^3\eta^2 + \frac{9K^2L\eta^2}{m}\sigma_G^2 \\
& + L\eta^2\mathbb{E}_r\left\|\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\nabla f_i^{\mu}(\boldsymbol{x}_{i,r}^k)\right\|^2
\end{aligned} \tag{37}
$$

Combining all pieces and denoting $\chi_r = \mathbb{E}_r\left[\frac{1}{M}\sum\limits_{i=1}^{M}\sum\limits_{k=1}^{K}\left\|\boldsymbol{x}_{i,r}^k - \boldsymbol{x}_r\right\|^2\right]$, we have

$$
\begin{aligned}
\mathbb{E}_r[f(\boldsymbol{x}_{r+1})] \leq & f(\boldsymbol{x}_r) - \frac{1}{2}K\eta\|\nabla f(\boldsymbol{x}_r)\|^2 - \frac{1}{2}K\eta\mathbb{E}_r\left\|\frac{1}{MK}\sum_{i=1}^{M}\sum_{k=1}^{K}\nabla f_i^{\mu}(\boldsymbol{x}_{i,r}^k)\right\|^2 + \frac{1}{4}\mu^2KL^2(d+3)^3\eta \\
& + L^2\eta\chi_r + \frac{6(d+4)LK\eta^2}{mP}\|\nabla f(\boldsymbol{x}_r)\|^2 + \frac{6L^3(d+4)\eta^2}{mP}\chi_r \\
& + \frac{6(d+4)LK\eta^2}{mP}\sigma_G^2 + \frac{2(d+4)LK\eta^2}{mP}\sigma^2 + \frac{\mu^2(d+6)^3L^3K\eta^2}{2mP} \\
& + 18KL^3\eta^2\chi_r + \frac{3}{2}\mu^2K^2L^3\eta^2(d+3)^3 \\
& + \frac{9K^2L\eta^2}{m}\sigma_G^2 + L\eta^2\mathbb{E}_r\left\|\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\nabla f_i^{\mu}(\boldsymbol{x}_{i,r}^k)\right\|^2
\end{aligned} \tag{38}
$$

$$
\begin{aligned}
\leq & f(\boldsymbol{x}_r) - \left(\frac{1}{2}K\eta - \frac{6(d+4)LK\eta^2}{mP}\right)\|\nabla f(\boldsymbol{x}_r)\|^2 + \frac{1}{4}\mu^2 KL^2(d+3)^3\eta \\
& + \left(L^2\eta + \frac{6L^3(d+4)\eta^2}{mP} + 18KL^3\eta^2\right)\chi_r + \frac{2KL(d+4)\eta^2}{mP}\sigma^2 \\
& + \left(\frac{9K^2L\eta^2}{m} + \frac{6(d+4)LK\eta^2}{mP}\right)\sigma_G^2 + \frac{\mu^2 KL^3(d+6)^3\eta^2}{2mP} + \frac{3}{2}\mu^2K^2L^3(d+3)^3\eta^2
\end{aligned}
\tag{39}
$$

Plugging lemma 5 into (39) and following the condition $\eta \leq \min\left\{\frac{mP}{24L(d+4)}, \frac{2P}{mKL(d+P+4)}, \frac{1}{mK^2L}, \frac{mP(d+3)^3}{2L[3mPK(d+3)^3+(d+6)^3]}\right\}$, we can further simplified it into

$$
\begin{aligned}
\mathbb{E}_r[f(\boldsymbol{x}_{r+1})] \leq & f(\boldsymbol{x}_r) - \frac{1}{4}K\eta\|\nabla f(\boldsymbol{x}_r)\|^2 + \left(\frac{18K^2L\eta^2}{m} + \frac{6(d+4)LK\eta^2}{mP}\right)\sigma_G^2 \\
& + \frac{4KL(d+4)\eta^2}{mP}\sigma^2 + \frac{1}{2}\mu^2KL^2(d+3)^3\eta
\end{aligned}
\tag{40}
$$

Rearranging the terms, we have

$$
\begin{aligned}
\frac{1}{4}K\eta\|\nabla f(\boldsymbol{x}_r)\|^2 \leq & f(\boldsymbol{x}_r) - \mathbb{E}_r[f(\boldsymbol{x}_{r+1})] + \left(\frac{18K^2L\eta^2}{m} + \frac{6(d+4)LK\eta^2}{mP}\right)\sigma_G^2 \\
& + \frac{4KL(d+4)\eta^2}{mP}\sigma^2 + \frac{1}{2}\mu^2KL^2(d+3)^3\eta
\end{aligned}
\tag{41}
$$

Dividing $\frac{1}{4}K\eta$ on both sides, then we get

$$
\begin{aligned}
\|\nabla f(\boldsymbol{x}_r)\|^2 \leq & \frac{4}{K\eta}\Big(f(\boldsymbol{x}_r) - \mathbb{E}_r[f(\boldsymbol{x}_{r+1})]\Big) + \left(\frac{72KL\eta}{m} + \frac{24(d+4)L\eta}{mP}\right)\sigma_G^2 \\
& + \frac{16L(d+4)\eta}{mP}\sigma^2 + 2\mu^2L^2(d+3)^3
\end{aligned}
\tag{42}
$$

Recursively executing (42) R rounds, we can obtain

$$
\begin{aligned}
\frac{1}{R}\sum_{r=0}^{R-1}\mathbb{E}_r\|\nabla f(\boldsymbol{x}_r)\|^2 \leq & \frac{4\big(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*)\big)}{KR\eta} + \left(\frac{72KL\eta}{m} + \frac{24(d+4)L\eta}{mP}\right)\sigma_G^2 \\
& + \frac{16L(d+4)\eta}{mP}\sigma^2 + 2\mu^2L^2(d+3)^3
\end{aligned}
\tag{43}
$$

■