# Federated Learning with Projected Trajectory Regularization

**Anonymous authors**
**Paper under double-blind review**

## Abstract

Federated learning enables joint training of machine learning models from distributed clients without sharing their local data. One key challenge in federated learning is to handle non-identically distributed data across the clients, which leads to deteriorated model training performance. Prior works in this line of research mainly focus on utilizing last-step global model parameters/gradients or the linear combinations of the past model parameters/gradients, which do not fully exploit the potential of global information from the model training trajectory. In this paper, we propose a novel federated learning framework with projected trajectory regularization (FedPTR) for tackling the data heterogeneity issue, which proposes a unique way to better extract the essential global information from the model training trajectory. Specifically, FedPTR allows local clients or the server to optimize an auxiliary (synthetic) dataset that mimics the learning dynamics of the recent model update and utilizes it to project the next-step model trajectory for local training regularization. We conduct rigorous theoretical analysis for our proposed framework under nonconvex stochastic settings to verify its fast convergence under heterogeneous data distributions. Experiments on various benchmark datasets and non-i.i.d. settings validate the effectiveness of our proposed framework.

## 1 Introduction

Federated Learning (FL) (McMahan et al., 2017; Konečnỳ et al., 2016) has recently emerged as an attractive learning paradigm where a large number of remote clients (e.g., mobile devices, institutions, organizations) cooperate to jointly learn a machine learning model without data sharing. Specifically, Federated Learning works by performing local training on each client based on their own data, while having the clients communicate their local model updates to a global server (McMahan et al., 2017; Konečnỳ et al., 2016). Enjoying privacy protection and efficient communication, Federated Learning has been widely applied in various domains such as edge computing (Wang et al., 2019), industrial engineering (Hu et al., 2018), and healthcare (Lee et al., 2018).

As a prevalent Federated Learning algorithm, FedAvg (McMahan et al., 2017) updates the global model by averaging multiple steps of local stochastic gradient descent (SGD) updates. Though FedAvg has demonstrated empirical success on independent and identically distributed (i.i.d) client data, when facing the more practical non-i.i.d data distributions across the clients, it usually leads to diverging updates and severe drift between the local and the global training objective in each client, which largely deteriorates the model training performance (Khaled et al., 2020; Karimireddy et al., 2020b; Wang et al., 2020b). For instance, imagine a group of hospitals that aims to collaboratively train a cancer diagnosis model together while keeping their own patients' data private through FedAvg. Due to different imaging machines and protocols used in hospitals, the medical images usually exhibit varying visual features (e.g., distinct intensity and contrast) (Li et al., 2021c) and thus the locally trained model may easily overfit and drift away from the global model. Similarly, in the task of anomaly detection on edge devices, the data collected on each edge device may only partially cover the entire normal space due to environmental conditions (Zhu et al., 2022). In such a case, simply training models based on FedAvg may obtain a global model that collapses to certain normal modes. As a result, many normal instances would be falsely detected as anomalies (Zhu et al., 2022).

To tackle this data heterogeneity issue, a line of research mainly focuses on utilizing or regularizing with the last-step global model parameters/gradients or the linear combinations of past model parameters/gradients. Specifically, FedProx (Li et al., 2020) makes a simple modification to FedAvg by adding a proximal regularization term in the local training objective to restrict the local updates to be closer to the last-step global model. SCAFFOLD (Karimireddy et al., 2020b) reduces client drift with a control variate that linearly accumulates past gradients and corrects for the drift. FedDC (Gao et al., 2022) follows a similar idea as SCAFFOLD with a local drift term and introduces a linear gradient correction loss, attempting to further eliminate the drift at local training.

However, such methods mentioned above may not be sufficient enough to fully exploit global information to tackle extreme data heterogeneity. In order to better extract the essential global information from the model training trajectory, we propose a novel federated learning framework with projected trajectory regularization (FedPTR), to help guide the local model update. Specifically, during the local training, FedPTR allows local clients or the server to optimize an auxiliary dataset that mimics the learning dynamics of the recent model update and then utilizes it to project the next-step model trajectory for local training regularization. To achieve the above goal and optimize the auxiliary dataset, we adopt the Matching Training Trajectories (MTT) technique (Cazenavette et al., 2022), which was originally proposed for the task of dataset distillation. Here we want to emphasize that our use of MTT is actually different from traditional dataset distillation whose goal is to synthesize a small dataset such that models trained on the synthetic dataset can achieve similar test accuracy as if trained on the actual dataset. Firstly, most works in dataset distillation cannot fit in here as they usually require access to the entire training dataset which is not practical for federated learning. Some existing works (Zhou et al., 2020; Song et al., 2022) aim to get around this by letting local clients distill their own data and send it to the server. However, this could incur extra communication costs and raise data privacy concerns. Secondly, the goal of dataset distillation is quite ambitious and usually requires obtaining multiple model training trajectories (or re-starts) in order to obtain barely satisfactory performance. In sharp contrast, here we only require the extracted auxiliary dataset to mimic the learning dynamics of recent model updates for projecting the next-step update, which is a much easier task that can be achieved with only one global model training trajectory. We summarize our contribution as follows:

- We propose a novel federated learning framework FedPTR that handles extreme data heterogeneity across local clients. The proposed method allows each client or the server to extract essential global information from the observed global model training trajectory and utilize it to project the next-step model trajectory for local training regularization.
- We provide rigorous theoretical analysis and prove that FedPTR can achieve a convergence rate of $\mathcal{O}(1/T)$ w.r.t total communication rounds $T$ under stochastic non-convex optimization settings, which matches the best existing results.
- We conduct extensive experiments on various benchmark datasets and non-i.i.d settings to demonstrate the effectiveness of our proposed FedPTR in training real-world machine learning models.

## 2 Related Work

**Federated Learning for Heterogeneous Data.** Federated Learning (McMahan et al., 2017) has been widely applied to a wide range of clients (e.g., mobile devices, institutions, and organizations) to collaboratively train a global model without sharing their private data. However, the heterogeneous data distribution across different devices has become one of the major bottlenecks for FL algorithms. A prominent line of work addresses this by regularizing local training objectives such as FedProx (Li et al., 2020) and FedDyn (Acar et al., 2021), and client drift correction via control variates such as SCAFFOLD (Karimireddy et al., 2020b), Mime (Karimireddy et al., 2020a) and FedDC (Gao et al., 2022). One recent line of work incorporates sharpness-aware minimization (SAM) into federated optimization to find flatter minima and improve generalization under data heterogeneity, such as FedSAM, MoFedSAM (Qu et al., 2022), FedSMOO (Sun et al., 2023a), FedLESAM (Fan et al., 2024) and FedGMT (Li et al., 2025). Another recent line to address the data heterogeneity is adopting client-level optimization methods such as MOON (Li et al., 2021a), FedSpeed (Sun et al., 2023b) and FedPD (Zhang et al., 2020). Moreover, personalized federated learning (Fallah et al., 2020;

Smith et al., 2017; Deng et al., 2020; Li et al., 2021b) has also been explored to better adapt global models to heterogeneous local distributions.

**Dataset Distillation.** Dataset distillation is proposed to learn a small synthetic dataset out of the original large dataset, such that the model trained on this small dataset can have a similar performance as the model trained on the original dataset. Wang et al. (2018) first introduced dataset distillation by gradient-based hyperparameter optimization. Bohdal et al. (2020) proposed a label distillation method to get soft labels for distilled images with the meta-learning framework. Zhao et al. (2021); Zhao & Bilen (2021b) proposed to match the gradient of the real dataset and the auxiliary dataset, and use data augmentation to boost the performance. Wang et al. (2022a); Zhao & Bilen (2021a) considered matching the distribution based on neural networks and largely reduced the memory cost. Matching Training Trajectories (Cazenavette et al., 2022) and its variants (Cazenavette et al., 2022; Liu et al., 2022; Du et al., 2022) are proposed to optimize the synthetic data by minimizing the distance between the synthetically trained parameters and the actual model training trajectories. Recent advances further push this direction: DATM (Guo et al., 2024) achieves lossless distillation by dynamically aligning trajectory difficulty to synthetic dataset size, and SeqMatch (Du et al., 2023) addresses accumulated trajectory error for more stable distillation.

**Federated Learning with Dataset Distillation.** Several recent works have tried to combine federated learning with dataset distillation. Specifically, Zhou et al. (2020) and Song et al. (2022) proposed a one-shot federated learning framework that allows all clients to distill their own dataset by local training and send the synthetic dataset to the server. The server will then use the collected distilled dataset for centralized training. Xiong et al. (2022) proposes a multi-shot framework that clients will distill their dataset for every communication round. Note that the above-mentioned works require (distilled) data transmission between clients and server: clients need to send their local distilled data to the server. Yet this incurs extra communication costs and data privacy concerns. Beyond reducing data transmission, a more fundamental question is how to effectively extract and utilize global information within the federated framework without any data sharing. FMKE (Liu et al., 2023) addresses this by condensing client knowledge into meta representations, though transmission of these representations is still required. DynaFed (Pi et al., 2023) takes a different angle by leveraging global model dynamics to synthesize data on the server side, yet the synthesized data is used to directly augment server-side training, rather than to guide local client updates.

## 3 Preliminaries on Federated Learning

In this section, we will briefly introduce the basics of federated learning. Consider a federated system with $N$ total clients participated, the goal of federated learning is to optimize the following objective function:

$$\min_{\mathbf{w}\in\mathbb{R}^d} \sum_{i=1}^{N} \frac{m_i}{M} \mathcal{L}(\mathbf{w}; \mathcal{D}_i), \tag{3.1}$$

where $\mathcal{L}$ denotes the loss function such as Cross Entropy loss, $\mathcal{D}_i$ denotes the training data on client $i$. Let $m_i = |\mathcal{D}_i|$ be the cardinality of $\mathcal{D}_i$ and we have $M = \sum_{i=1}^{N} m_i$ as total size of the training dataset.

**FedAvg.** FedAvg (McMahan et al., 2017) is designed to solve Eq (3.1) by coordinating multiple devices via a central server. Specifically, in each round of FedAvg, the server will broadcast the current global model $\mathbf{w}^t$ to the clients. Each client will locally optimize the corresponding component function in Eq (3.1) and obtain the local model $\mathbf{w}_i^{t+1}$, i.e., $\mathbf{w}_i^{t+1} = \operatorname{argmin}_{\mathbf{w}} \mathcal{L}(\mathbf{w}; \mathcal{D}_i)$, through multiple steps of stochastic gradient descent. The server will then aggregate (weighted average) the local models $\{\mathbf{w}_i^{t+1}\}_{i=1}^{N}$ to obtain the new global model.

**FedProx.** FedProx (Li et al., 2020) is one of the representative works which is designed to tackle both the data heterogeneity issue in federated learning and allows flexible performances on each of the local clients. Specifically, it allows the local objectives to be solved inexactly such that the amount of local computation/communication can be adapted for different clients with different amounts of resources. To formally define this inexactness, Li et al. (2020) introduced the following definition:

**Definition 3.1** ($\gamma$-Inexactness)**.** Given a function $h(\mathbf{w}, \mathbf{w}_0) = \mathcal{L}(\mathbf{w}) + \frac{\lambda}{2}\|\mathbf{w} - \mathbf{w}_0\|_2^2$ and $\gamma \in [0, 1]$, $\mathbf{w}^*$ is a $\gamma$-inexact solution of $\min_{\mathbf{w}} h(\mathbf{w}, \mathbf{w}_0)$ if $\|\nabla h(\mathbf{w}^*, \mathbf{w}_0)\|_2 \leq \gamma \|\nabla h(\mathbf{w}_0, \mathbf{w}_0)\|_2$.

Additionally, FedProx introduced a proximal term to the local objective to limit the local updates to be not too far away from its initial global model $\mathbf{w}^t$. In particular, the local objectives of FedProx become:

$$\mathbf{w}_i^{t+1} = \underset{\mathbf{w}}{\operatorname{argmin}}\, \mathcal{L}(\mathbf{w}; \mathcal{D}_i) + \frac{\lambda}{2}\|\mathbf{w} - \mathbf{w}^t\|_2^2, \tag{3.2}$$

where $\lambda$ is the hyperparameter that controls the strength of the proximal term. We can observe that FedProx handles data heterogeneity by restricting the local updates to stay close to the initial global model.

## 4 Proposed Method

In this section, we introduce our proposed federated learning framework with Projected Trajectory Regularization (FedPTR) for tackling the data heterogeneity issue in federated learning. Due to the privacy-preserving design of federated learning, each client can only access their own training data. Therefore, the global model training trajectories contain all the information each client could possibly access with regard to other clients' local training data. Naturally, most existing works seek help from global models or global variables to mitigate the data heterogeneity issue. Different from existing works which mainly focus on utilizing only the last-step global model or the linear combination of past global models, FedPTR aims to provide a better way to exploit the global information hidden in the model training trajectories via Matching Training Trajectories (MTT) (Cazenavette et al., 2022). More specifically, we aim to extract an auxiliary dataset from the recent model training trajectories that could mimic the learning dynamics of the recent model update. Thus we can utilize such an auxiliary dataset for projecting the next-step training trajectory and provide better guidance to the local model training.

**Extract Global Information via MTT.** We found MTT (Cazenavette et al., 2022) to be a perfect tool for our global information extraction purposes. The general goal of MTT is to learn an informative but small synthetic dataset that approximates the true learning trajectories. Moreover, it only requires model parameters from previous training epochs which naturally fit the setting of federated learning. We summarize the detailed MTT process in Algorithm 2.

Specifically, we treat the last model in the actual training trajectories (denoted by $\mathbf{w}^{\text{end}}$) as the teacher model in MTT and build a student model $\widehat{\mathbf{w}}$, which is obtained by multiple steps of gradient descent training on the synthetic dataset $\widetilde{\mathcal{D}}$ starting from $\mathbf{w}^{\text{start}}$ in the trajectory. By requiring the student model to stay close to the teacher model, we force MTT to learn a synthetic dataset $\widetilde{\mathcal{D}}$ to that well-represents the recent learning dynamics. In detail, we have the following MTT loss:

$$\mathcal{L}_{\text{MTT}}(\widetilde{\mathcal{D}}) = \|\widehat{\mathbf{w}} - \mathbf{w}^{\text{start}}\|_2^2 / \|\mathbf{w}^{\text{end}} - \mathbf{w}^{\text{start}}\|_2^2, \tag{4.1}$$

where the normalization term ensures that the MTT loss remains effective even when the model's changes are small in the actual training trajectory. We can optimize $\mathcal{L}_{\text{MTT}}(\widetilde{\mathcal{D}})$ over $\widetilde{\mathcal{D}}$ via gradient descent to obtain the auxiliary synthetic dataset.

**Algorithm 1** Federated Learning with Projected Trajectory Regularization (FedPTR)

**Input:** $T$, $m$, $\eta$, $\lambda$.
Initialize $\widetilde{\mathcal{D}}_i$ for all clients
**for** $t = 0$ **to** $T - 1$ **do**
  **for** each client $i \in [N]$ **do**
    Client $i$ receives $\mathbf{w}^t$ from the server
    $\widetilde{\mathbf{w}}_i^{t+1} = \mathbf{w}^t$
    **if** $t > m$ **then**
      $\widetilde{\mathcal{D}}_i \leftarrow \text{MTT}(\mathbf{w}^{t-m}, \mathbf{w}^t, \widetilde{\mathcal{D}}_i)$
      **for** $k = 0$ **to** $K - 1$ **do**
        $\widetilde{\mathbf{w}}_i^{t+1} \leftarrow \widetilde{\mathbf{w}}_i^{t+1} - \eta \nabla \mathcal{L}(\widetilde{\mathbf{w}}_i^{t+1}; \widetilde{\mathcal{D}}_i)$
      **end for**
      Set $\widetilde{\lambda} = \lambda$
    **else**
      Set $\widetilde{\lambda} = 0$
    **end if**
    Solve $\mathbf{w}_i^{t+1}$ as a $\gamma$-inexactness minimizer of: $\mathbf{w}_i^{t+1} \approx \underset{\mathbf{w}}{\operatorname{argmin}}\, \mathcal{L}(\mathbf{w}; \mathcal{D}_i) + \frac{\widetilde{\lambda}}{2} \cdot \|\mathbf{w} - \widetilde{\mathbf{w}}_i^{t+1}\|_2^2$
    Client sends $\mathbf{w}_i^{t+1}$ to server
  **end for**
  Server aggregates $\{\mathbf{w}_i^{t+1}\}_{i=1}^N$ to get $\mathbf{w}^{t+1}$
**end for**

**Projected Trajectory Regularization.** Once we obtain the auxiliary dataset via MTT, the following step is to project the next-step training trajectory and design our federated learning framework. We summarize our design at Algorithm 1 and give more details in the following [1].

Specifically, for each client (after certain training rounds $m$), we utilize MTT for extracting the auxiliary dataset $\widetilde{\mathcal{D}}_i$. Since the auxiliary dataset is designed to mimic the training trajectory from the $(t-m)$-th round global model to the $t$-th round global model, we can obtain our projected next-step model training trajectory $\widetilde{\mathbf{w}}_i^{t+1}$ by further performing several steps of gradient descent training on $\widetilde{\mathcal{D}}_i$ with learning rate $\eta$.

Inspired by FedProx, we also design a proximal term for guiding the local training. Different from FedProx, we don't use the last-step global model $\mathbf{w}^t$, instead, we hope the local update to stay close to our projected training trajectory $\widetilde{\mathbf{w}}_i^{t+1}$:

$$\min_{\mathbf{w}} \mathcal{L}(\mathbf{w}; \mathcal{D}_i) + \frac{\lambda}{2}\|\mathbf{w} - \widetilde{\mathbf{w}}_i^{t+1}\|_2^2, \tag{4.2}$$

where $\lambda$ controls the strength of the proximal term. Through such a design, we utilized the extracted global information from past training trajectories to help guide the local model update.

To make our algorithm more practical and provide more flexibility for local clients, we also follow FedProx and only require a $\gamma$-inexactness solution during local training. After finishing the local computation, clients will send their updated model to the server and the rest steps follow the same procedure as vanilla FedAvg.

**Initialization of the Auxiliary Dataset.** Previous studies (Cazenavette et al., 2022) have shown that using real data samples from true classes can improve the performance of MTT. However, it is not very practical in federated learning with heterogeneous data distribution, as it is nearly impossible for individual clients to possess real data samples from each class. Hence, we use an alternative strategy to initialize the auxiliary dataset $\widetilde{\mathcal{D}}_i$. Specifically, for those classes that have corresponding samples in the local's training data, we directly random sample (with replacement) the desired amount of real data from the local training data. For those classes that do not have samples in the local training data, we adopt the random initialization strategy.

**Algorithm 2** Matching Training Trajectories (MTT)

**Input:** $\mathbf{w}^{\text{start}}$, $\mathbf{w}^{\text{end}}$, $\widetilde{\mathcal{D}}_i$.
**for** $h = 0$ **to** $H-1$ **do**
  $\widehat{\mathbf{w}} = \mathbf{w}^{\text{start}}$
  **for** $r = 0$ **to** $R-1$ **do**
    $\widehat{\mathbf{w}} \leftarrow \widehat{\mathbf{w}} - \beta \nabla \mathcal{L}(\widehat{\mathbf{w}}; \widetilde{\mathcal{D}}_i)$
  **end for**
  Computes $\mathcal{L}_{\text{MTT}}(\widetilde{\mathcal{D}}_i) = \|\widehat{\mathbf{w}} - \mathbf{w}^{\text{end}}\|_2^2 / \|\mathbf{w}^{\text{end}} - \mathbf{w}^{\text{start}}\|_2^2$
  Updates $\widetilde{\mathcal{D}}_i$ with respect to $\mathcal{L}_{\text{MTT}}(\widetilde{\mathcal{D}}_i)$
**end for**
**Output:** $\widetilde{\mathcal{D}}_i$

**Adaptive Regularization Parameter.** In Eq (3.2), we use a standard squared $L_2$ norm regularization as the proximal term. Following common machine learning practices, we usually adopt a constant $\lambda$ as the regularization parameter. However, one thing we notice is that in practice, when applying this $L_2$ norm regularization in neural network models, the model difference $\mathbf{w} - \widetilde{\mathbf{w}}_i^{t+1}$ can vary drastically among different convolutional layers. In Figure 1, we plot the $L_2$ norms of the four different convolutional layers in ConvNet(Gidaris & Komodakis, 2018), with the model trained under a 10-clients federated learning setup, and each client's training data sampled from the original CIFAR-10 dataset following Dirichlet distribution ($\alpha = 0.01$).

From Figure 1, we observe that the $L_2$ norm of $\mathbf{w} - \widetilde{\mathbf{w}}_i^{t+1}$ between layers varies and it also changes along with different phases of training. For example, at the beginning of training, the fourth layer has the largest $L_2$ norm while the second layer has the largest $L_2$ norm when the training is about to end. Such a phenomenon indicates that it might be helpful to use a layer-adaptive $\lambda$ rather than a constant.

Specifically, let $\mathbf{w}_{[j]}$ denotes parameters for the $j$-th layer in $\mathbf{w}$ and $\widetilde{\mathbf{w}}_{i,[j]}^{t+1}$ denotes parameters for the $j$-th layer in $\widetilde{\mathbf{w}}$, we can define the adaptive $\lambda_j$ as follows:

$$\lambda_j = \frac{1}{\left\|\mathbf{w}_{[j]} - \widetilde{\mathbf{w}}_{i,[j]}^{t+1}\right\|_2} \cdot \lambda. \tag{4.3}$$

[1] Algorithm 1 focused on the full participation setting where all clients participate in each round of federated learning. For partial participation cases, we can simply allow selected clients to perform MTT from the past trajectories it observed.

Table 1: Comparisons of layer-adaptive $\lambda$ with fixed $\lambda$ in FedPTR on CIFAR-10 and CIFAR-100 dataset.

| Reg. param. | Dirichlet $\alpha = 0.01$ | | Dirichlet $\alpha = 0.04$ | |
|---|---|---|---|---|
| | CIFAR 10 | CIFAR 100 | CIFAR 10 | CIFAR 100 |
| Fixed $\lambda$ | 62.10 | 41.26 | 67.68 | 42.41 |
| Layer-adaptive $\lambda$ | 69.04 | 44.98 | 72.78 | 45.87 |

Here, we compare our layer adaptive $\lambda$ with the standard fixed $\lambda$. We report results on CIFAR-10 and CIFAR-100 with the parameter of Dirichlet distribution $\alpha = 0.01$ and 0.04 as shown in Table 1. We can observe that our FedPTR adopting layer adaptive $\lambda$ achieves better performances compared with the fixed $\lambda$ setting. By default, we report all our experimental results in Section 6 with the layer adaptive $\lambda$.

**Server-Side FedPTR.** One problem of FedPTR presented in Algorithm 1 is that each client needs to perform Matching Training Trajectories and update their own auxiliary dataset in each communication round, which would introduce extra computation costs. To make our method smoothly accommodate resource-constrained devices (e.g., mobile phones, robots, drone swarms), we also provide a server-side FedPTR algorithm (denoted by FedPTR-S), which only optimizes the auxiliary dataset through MTT in the global server. We summarize the detailed algorithm in Algorithm 3. Specifically, in each communication round, the global server first starts to match training trajectories and optimize a unified $\widetilde{\mathbf{w}}^{t+1}$. Then, each client will receive both $\mathbf{w}^t$ and $\widetilde{\mathbf{w}}^{t+1}$ for local training. Note that different from client-side FedPTR as shown in Algorithm 1, the server-side version initializes the auxiliary dataset using random noise without any real data from local clients (see a detailed comparison of different initialization strategies in B.1). Therefore, our FedPTR-S can preserve data privacy while also making participating devices free from extra computation burdens.

## 5 Theoretical Analysis

In this section, we provide the theoretical convergence analysis for the proposed FedPTR framework. We first state several assumptions needed for the convergence analysis.

**Assumption 5.1.** The loss function on the $i$-th client $\mathcal{L}(\mathbf{w};\mathcal{D}_i)$ is $L$-smooth, i.e., $\forall \mathbf{w}, \mathbf{v} \in \mathbb{R}^d$,

$$\left|\mathcal{L}(\mathbf{w};\mathcal{D}_i) - \mathcal{L}(\mathbf{v};\mathcal{D}_i) - \langle \nabla\mathcal{L}(\mathbf{v};\mathcal{D}_i), \mathbf{w}-\mathbf{v}\rangle\right| \leq \frac{L}{2}\|\mathbf{w}-\mathbf{v}\|_2^2.$$

**Assumption 5.2.** The gradient of the local loss function on the $i$-th client $\nabla\mathcal{L}(\mathbf{w};\mathcal{D}_i)$ is $B$-locally dissimilar, i.e.,

$$\frac{1}{N}\sum_{i\in[N]} \|\nabla\mathcal{L}(\mathbf{w};\mathcal{D}_i)\|_2^2 \leq B^2\|\nabla\mathcal{L}(\mathbf{w})\|_2^2.$$

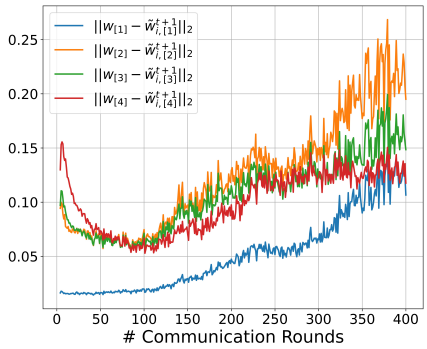



Figure 1: Comparisons of layer-wise $L_2$ norm $\left\|\mathbf{w}_{[j]} - \widetilde{\mathbf{w}}_{i,[j]}^{t+1}\right\|_2$ against the communication rounds, where $j$ denotes $j$-th layer. The layer-wise $L_2$ norm values vary significantly with different layers and communication rounds.

Assumption 5.1 and 5.2 are standard assumptions in federated learning optimization problems (Acar et al., 2021; Li et al., 2020; Karimireddy et al., 2020b; Xu et al., 2021; Gao et al., 2022). Assumption 5.2 describes the dissimilarity between the gradient of local objectives and the global objective. Note that the commonly used (Li et al., 2020; Reddi et al., 2020; Wang et al., 2022b) bounded variance assumption, $\frac{1}{N}\sum_{i\in[N]}\|\nabla\mathcal{L}(\mathbf{w};\mathcal{D}_i) - \nabla\mathcal{L}(\mathbf{w})\|^2 \leq \sigma^2$, also satisfies the condition in Assumption 5.2.

**Assumption 5.3.** The auxiliary loss function on the $i$-th client $\widetilde{\mathcal{L}}(\mathbf{w};\widetilde{\mathcal{D}}_i)$ has $\ell_2$-bounded gradient dissimilarity with global objective function $\mathcal{L}(\mathbf{w})$, i.e., for all $\mathbf{w} \in \mathbb{R}^d$, we have

$$\|\nabla\mathcal{L}(\mathbf{w}) - \nabla\widetilde{\mathcal{L}}(\mathbf{w};\widetilde{\mathcal{D}}_i)\|_2^2 \leq \sigma_d^2.$$

Assumption 5.3 measures the gradient dissimilarity between the global loss function and the training loss on the auxiliary dataset $\widetilde{\mathcal{D}}_i$, which implies that the MTT algorithm is actually well functioning: the extracted auxiliary dataset indeed mimics the actual global model training trajectories.

In the following, we will state the theoretical convergence analysis for our proposed FedPTR. For expository purposes, we assume $K = 1$, i.e., performing 1 step of auxiliary model update in each round, though our convergence analysis can also be extended to cases when $K > 1$.

**Theorem 5.4** (Non-convex FedPTR convergence)**.** Under Assumptions 5.1-5.3, assume the local objective functions $\mathcal{L}(\mathbf{w}; \mathcal{D}_i)$ are non-convex, and there exists $L_- > 0$, such that $\nabla^2 \mathcal{L}(\mathbf{w}; \mathcal{D}_i) \succeq -L_- \mathbf{I}$ with $\mu = \lambda - L_- > 0$. If $\eta \leq \frac{1}{2\lambda}$, $\gamma$ is selected to be sufficiently small, $\mu$ and $\lambda$ satisfies

$$\frac{1}{6\lambda} - \frac{4L^2}{\lambda\mu^2} - \frac{2L^2}{\lambda\mu^2} - \frac{1}{\lambda N} - \frac{2L}{\mu^2} - \frac{4L}{\mu^2} \geq 0, \tag{5.1}$$

then the iterations of FedPTR satisfy

$$\min_{t \in [T]} \mathbb{E}[\|\nabla \mathcal{L}(\mathbf{w}_t)\|_2^2] \leq \mathcal{O}\left(\frac{\mathbb{E}[\mathcal{L}(\mathbf{w}_0)] - \mathbb{E}[\mathcal{L}(\mathbf{w}_T)]}{6\lambda T}\right) + \mathcal{O}(\eta^2 \sigma_d^2). \tag{5.2}$$

Theorem 5.4 suggests that the gradient norm of FedPTR has an upper bound that contains two terms. The first term vanishes as the communication round $T$ increases and the second term is related to $\sigma_d^2$, i.e., the gradient dissimilarity between global training loss and the training loss on the auxiliary dataset. This implies that if the auxiliary gradient differs significantly from the global gradient, it will hurt FedPTR's performance as larger $\sigma_d$ leads to worse convergence. In such a case, the auxiliary gradient cannot well-represent the global gradient direction, hence to obtain a desired convergence performance, it requires a smaller $\eta$ to control the auxiliary model $\widetilde{\mathbf{w}}_{t+1}$ to stay close to the global initial model $\mathbf{w}_t$ and obtain smaller second term in Eq (5.2).

**Remark 5.5.** Theorem 5.4 can be extended to our proposed server-side FedPTR (Algorithm 3) with slightly modifying Assumption 5.3 to $\|\nabla \mathcal{L}(\mathbf{w}) - \nabla \widetilde{\mathcal{L}}(\mathbf{w}; \widetilde{\mathcal{D}})\|_2^2 \leq \sigma_d^2$, which describes the dissimilarity between the server-side auxiliary loss and the global objective function.

**Corollary 5.6.** If choosing the auxiliary learning rate as $\eta = \Theta\big(\frac{1}{\sqrt{T}}\big)$, then the convergence rate for Algorithm 1 satisfies

$$\min_{t \in [T]} \mathbb{E}[\|\nabla \mathcal{L}(\mathbf{w}^t)\|_2^2] = \mathcal{O}\left(\frac{1}{T}\right). \tag{5.3}$$

Corollary 5.6 suggests that with an appropriately chosen learning rate, FedPTR achieves a convergence rate of $\mathcal{O}(1/T)$ in stochastic non-convex optimization settings, which matches the convergence rate of federated learning with proximal term such as FedProx (Li et al., 2020) and FedDyn (Acar et al., 2021).

## 6 Experiments

In this section, we present comprehensive experimental results to show the effectiveness of FedPTR when competing with other state-of-the-art federated learning methods under various heterogeneous data partitions. We first introduce our experimental settings in 6.1 and give a detailed analysis of our experimental results and ablation study in the following sections. We report the average accuracy of the last five global rounds on the test set.

### 6.1 Experimental Settings

**Datasets.** We conduct experiments on four benchmark datasets: FashionMNIST (Xiao et al., 2017), CIFAR-10 (Krizhevsky et al., 2009) CIFAR-100 (Krizhevsky et al., 2009), and TinyImageNet (Le & Yang, 2015). We follow the commonly used mechanism to perform data partition through Dirichlet distribution (Wang et al., 2020a). The parameter $\alpha$ used in Dirichlet sampling determines the non-i.i.d degree. with smaller

Table 2: Comparisons of test accuracy for FedPTR (FedPTR-S) and other baselines in training different datasets and under different participation ratios with 10 clients. **Bold** and underline represent the best and second best results.

| Participation Ratio | Dataset | FedAvg | FedProx | Scaffold | FedDyn | FedDC | FedPTR | FedPTR-S |
|---|---|---|---|---|---|---|---|---|
| **100%** | **Dirichlet(0.01)** | | | | | | | |
| | FMNIST | 83.69 | 83.75 | 82.21 | 84.36 | 78.23 | 85.11 | **85.48** |
| | CIFAR-10 | 55.43 | 55.94 | 56.58 | 62.29 | 52.55 | **69.04** | 68.16 |
| | CIFAR-100 | 37.32 | 37.33 | 42.02 | 41.73 | 39.52 | **44.98** | 44.82 |
| | **Dirichlet(0.04)** | | | | | | | |
| | FMNIST | 85.05 | 84.94 | 84.59 | 85.26 | 82.17 | **86.80** | 85.90 |
| | CIFAR-10 | 65.44 | 65.46 | 68.98 | 70.26 | 68.93 | 72.78 | **73.07** |
| | CIFAR-100 | 39.65 | 39.63 | 44.66 | 45.57 | 44.23 | 45.87 | **46.10** |
| **50%** | **Dirichlet(0.01)** | | | | | | | |
| | FMNIST | 81.02 | 81.05 | 75.57 | 79.39 | 73.18 | **81.76** | 81.71 |
| | CIFAR-10 | 49.90 | 50.28 | 54.91 | 56.81 | 50.65 | **56.95** | 54.24 |
| | CIFAR-100 | 37.07 | 37.04 | 37.35 | 37.61 | 37.98 | **40.11** | 39.49 |
| | **Dirichlet(0.04)** | | | | | | | |
| | FMNIST | 81.75 | 81.73 | 77.42 | 80.71 | 77.44 | 82.85 | **83.05** |
| | CIFAR-10 | 56.84 | 56.80 | 57.58 | 59.45 | 61.31 | **65.60** | 65.06 |
| | CIFAR-100 | 38.22 | 38.24 | 38.33 | 40.68 | 40.50 | 43.63 | **44.99** |
| **30%** | **Dirichlet(0.01)** | | | | | | | |
| | FMNIST | 75.89 | 76.40 | 73.05 | 75.09 | 73.09 | **76.79** | 76.78 |
| | CIFAR-10 | 41.82 | 43.06 | 47.92 | 45.20 | 44.01 | **48.94** | 47.41 |
| | CIFAR-100 | 29.68 | 29.65 | 29.18 | 30.80 | 32.67 | **35.41** | 33.64 |
| | **Dirichlet(0.04)** | | | | | | | |
| | FMNIST | 77.79 | 78.75 | 77.11 | 78.89 | 76.99 | 79.03 | **79.48** |
| | CIFAR-10 | 48.50 | 48.52 | 53.14 | 44.21 | 55.72 | 56.67 | **57.75** |
| | CIFAR-100 | 32.95 | 32.87 | 37.46 | 36.82 | 37.57 | 37.49 | **38.21** |

Table 3: Comparisons of FedPTR (FedPTR-S) with other baselines for partial participation in training different datasets with 40 clients. **Bold** and underline represent the best and second best results.

| Setting | Dataset | FedAvg | FedProx | Scaffold | FedDyn | FedDC | FedPTR | FedPTR-S |
|---|---|---|---|---|---|---|---|---|
| 40 clients with **25%** participation ratio | **Dirichlet(0.01)** | | | | | | | |
| | CIFAR-100 | 32.68 | 32.71 | 36.07 | 29.91 | 35.66 | **37.50** | 36.40 |
| | **Dirichlet(0.04)** | | | | | | | |
| | FMNIST | 83.73 | 83.77 | 79.76 | 78.78 | 82.71 | **85.92** | 84.68 |
| | CIFAR-10 | 58.00 | 58.06 | 58.32 | 54.11 | 55.13 | **60.30** | 59.73 |
| | CIFAR-100 | 38.35 | 38.32 | 40.34 | 38.67 | **42.74** | 41.90 | 41.11 |

$\alpha$, non-i.i.d degree becomes higher. In our experiments, we focus on the high-level data heterogeneity with $\alpha = 0.01$ and $\alpha = 0.04$.

**Baselines.** We compare our method FedPTR (and FedPTR-S) with five state-of-the-art federated learning methods: FedAvg (McMahan et al., 2017), FedProx (Li et al., 2020), SCAFFOLD (Karimireddy et al., 2020b), FedDyn (Acar et al., 2021), and FedDC (Gao et al., 2022). All the baseline methods except FedAvg contain utilization of the global information especially on the consistency of the global model and local model while FedPTR integrates the global information by training an auxiliary dataset based on the global parameters.

**Hyper-parameters and Model Architectures.** In our experiments, we set the batch size as 500 and the local training epoch as 1. By default, we use SGD with a learning rate $\eta = 0.01$ and momentum of 0.5 as the optimizer. We use 10 images per class for FashionMNIST/CIFAR-10 and 2 images per class for CIFAR-100. We set trajectory projection steps $K = 5$ and the base $\lambda$ for layer-adaptive regularization parameter as $\lambda = 0.05$. We also tune the hyperparameters in all other baselines for their best performances. In matching training trajectories part (Algorithm 2), following (Cazenavette et al., 2022), we utilize a learnable $\beta$ with

initialized $\beta = 0.01$, trajectory matching outer steps $H = 20$, inner steps $R = 10$, while the rest of the MTT hyper-parameters are set as the default value in (Cazenavette et al., 2022). In terms of the model architecture, we follow (Cazenavette et al., 2022) and adopt ConvNet(Gidaris & Komodakis, 2018) as the default CNN architecture.

### 6.2 Comparison of Different Federated Learning Methods on Heterogeneous Data

We compare our method FedPTR (and FedPTR-S) with five state-of-the-art federated learning optimization methods: FedAvg (McMahan et al., 2017), FedProx (Li et al., 2020), SCAFFOLD (Karimireddy et al., 2020b), FedDyn (Acar et al., 2021), and FedDC (Gao et al., 2022), on varying data heterogeneity, the number of clients, and participation rate. Table 2 shows the comparison results of our method FedPTR with baselines for the scenario of full participation and partial participation with 10 clients. We perform evaluations on two different heterogeneous data partitions, Dirichlet $\alpha = 0.04$ and Dirichlet $\alpha = 0.01$. In the full participation setting (participation ratio 100%), we observe that our method outperforms previous methods regarding test accuracy on all datasets. In particular, we achieve the absolute improvement of 6.75% compared with the best-performing baseline FedDyn and over 12% improvement over other baselines on CIFAR-10 dataset with Dirichlet $\alpha = 0.01$. Moreover, it can be seen that previous methods suffer from performance decreases of up to 16.38% when the data partition becomes more heterogeneous (i.e., from Dirichlet $\alpha = 0.04$ to Dirichlet $\alpha = 0.01$), while our method only reduces the test accuracy by up to 4.91%. Such results demonstrate that our method is consistently robust against extremely heterogeneous training data. In partial participation, we evaluate our method and baselines with 50% and 30% participation ratios. We can see that our method obtains relative improvements of $0.9\% \sim 17.7\%$ and $1.2\% \sim 19.3\%$ on 50% participation and 30% participation scenarios compared with FedAvg. These results clearly show the effectiveness of our method for the scenario of partial participation. To further verify our method in the scenario of more clients, we additionally compare FedPTR with baselines in 40 clients and 25% participation ratio as shown in Table 3. We observe that our method FedPTR (and FedPTR-S) still maintains excellent performance compared with the previous methods. Note that in the setting of Dirichlet $\alpha = 0.01$, we only evaluate our method on CIFAR-100 since FashionMNIST and CIFAR-10 with fewer categories cannot make each client have training data.

### 6.3 Reducing the Updates of Auxiliary Dataset

The original design of FedPTR (FedPTR-S) requires the clients/server to update the auxiliary datasets in each communication round. In this section, we explore how performance scales if we reduce the frequency of the auxiliary dataset updates. In Figure 2, we compare the test accuracy against the communication rounds on CIFAR-10 Dirichlet $\alpha = 0.01$ by performing one MTT update per 1, 4, 10, and 40 rounds. We additionally try a one-shot MTT strategy by only updating the auxiliary dataset once. Note that when there is no MTT update over the auxiliary dataset, we still need to perform the trajectory projection step (with the "old" auxiliary dataset) for local update regularization. We find that our proposed FedPTR method does not suffer much in performance even when the number of updates of the auxiliary dataset is reduced by 75%. Moreover, even the most efficient FedPTR with one-shot MTT can still outperform the other baseline methods such as FedDyn (Acar et al., 2021) on CIFAR-10.

### 6.4 Changing the Size of Auxiliary Dataset

We study how the size of the auxiliary dataset affects our final performances. For this, we conduct experiments comparing FedPTR using different sizes of the auxiliary dataset on CIFAR-10 and CIFAR-100 dataset and show the results in Table 4. For CIFAR-10, we explore 5 different sizes, and the results suggest that a moderate size of the auxiliary dataset leads to the best model training performances. When the size is below 100, the test accuracy is largely affected due to less representation power limited by the auxiliary dataset size. On the other hand, when the size grows beyond 100, there is a clear marginal effect and the influence of the size is not as obvious as before (may even lose performances due to insufficient optimization of the large auxiliary data). For CIFAR-100, a similar trend can also be observed. Note that using a larger auxiliary dataset leads to a larger computational cost when performing MTT updates in each communication

round. In practice, we would be picking a moderate auxiliary dataset size (such as 100) for the best of both performance and computational cost.

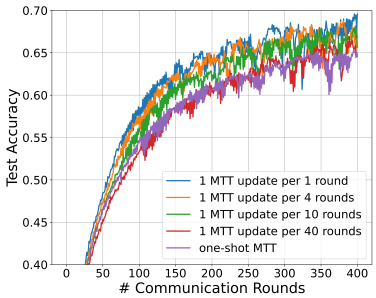



Figure 2: Comparisons of FedPTR test accuracy against communication rounds with the different frequency of MTT updates on CIFAR-10 dataset with Dirichlet $\alpha = 0.01$.

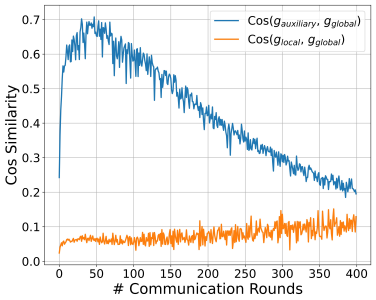



Figure 3: Comparisons of the cosine similarity between the auxiliary gradient and the global gradient, as well the cosine similarity between the local gradient and the global gradient.

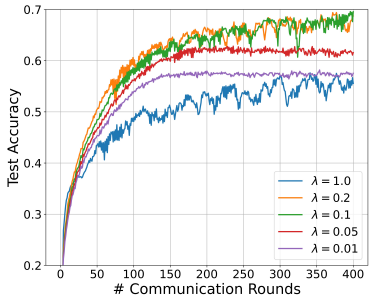



Figure 4: $\lambda$ sensitivity test of FedPTR on CIFAR-10 with Dirichlet $\alpha = 0.01$.

### 6.5 Quality of Extracted Global Information

In this subsection, we give a brief analysis on the quality of the extracted global information by the auxiliary dataset. To do so, we measure the similarity between the gradient of the training loss on the auxiliary dataset (denoted by the auxiliary gradient) and the global gradient. A high similarity between the two gradients could indicate that extracted auxiliary datasets well-approximate the recent learning dynamics and can well project the next-step training trajectory. For computational simplicity, here we directly use the global update direction on the server to approximate the global gradient direction, i.e., $\mathbf{w}^t - \mathbf{w}^{t+1}$. And we use $\mathbf{w}^t - \widetilde{\mathbf{w}}_i^{t+1}$ to approximate the auxiliary gradient. To further demonstrate the quality of extracted global information, we also compute the cosine similarity of local gradient $\mathbf{w}^t - \mathbf{w}_i^{t+1}$ and global gradient. Note that we compute two cosine similarities on vanilla FedAvg, i.e., clients keep the update of the auxiliary dataset and projected training trajectory while not using any regularization term to avoid the auxiliary dataset influencing the training gradient. We present our results for one randomly chosen client in training a CIFAR-100 model with Dirichlet $\alpha = 0.01$ in Figure 3. Compared with the local gradient, the auxiliary gradient has a much higher similarity to the global gradient, which suggests that FedPTR can indeed excavate global information from the model training trajectory. Note that the cosine similarity of the auxiliary gradient and global gradient decreases with the increase of the communication rounds. This suggests that it is harder to extract global information at the later training phase. Nevertheless, we can still find that the auxiliary gradient shares greater similarity to the global gradient direction compared with the local gradient along the entire training trajectory.

Table 4: Comparisons of FedPTR with different sizes of the auxiliary dataset with Dirichlet $\alpha = 0.01$.

| Dataset | Size of $\widetilde{\mathcal{D}}$ | Test Acc |
|---|---|---|
| CIFAR-10 | 10 | 65.83 |
| | 50 | 67.02 |
| | 100 | 69.04 |
| | 150 | 68.46 |
| | 200 | 69.00 |
| CIFAR-100 | 100 | 44.57 |
| | 200 | 44.98 |
| | 300 | 44.67 |

### 6.6 Ablation Study

**Sensitivity Analysis of Regularization weight.** One of the important hyperparameters of FedPTR is the weight of the regularization term $\lambda$. To test its sensitivity, we consider the non-i.i.d. setting of full participation on CIFAR-10 with Dirichlet $\alpha = 0.01$. We conduct the sensitivity test from the candidate set $\{0.01, 0.05, 0.1, 0.2, 1.0\}$. Figure 4 shows the convergence plot at varying $\lambda$, while we keep all other

hyperparameters of FedPTR consistent. We can observe that the best test accuracy is obtained when $\lambda = 0.1$. The small $\lambda$ (e.g., 0.01) may not have any effect, and the large $\lambda$ (e.g., 1.0) possibly force $\mathbf{w}$ to be close to $\widetilde{\mathbf{w}}_i^K$ thus causing slow convergence.

Table 5: Comparisons of FedPTR-S with different trajectory matching steps with Dirichlet $\alpha = 0.01$ and 0.04.

| | Dirichlet $\alpha$ | Trajectory matching steps $H$ | | | |
|---|---|---|---|---|---|
| | | 5 | 10 | 20 | 30 |
| CIFAR-10 | $\alpha = 0.01$ | 67.00 | 68.12 | 68.16 | 67.04 |
| | $\alpha = 0.04$ | 72.85 | 71.44 | 73.07 | 73.20 |
| CIFAR-100 | $\alpha = 0.01$ | 44.56 | 44.53 | 44.82 | 44.89 |
| | $\alpha = 0.04$ | 45.60 | 45.89 | 46.10 | 45.85 |

**Impact of trajectory matching steps.** In Table 5, we present the performance of our method with varying total trajectory matching steps $H$ used in Algorithm 2. Specifically, we consider 10 clients, full participation settings with Dirichlet $\alpha = 0.01$ and Dirichlet $\alpha = 0.04$. We can see that our method is stable with respect to different trajectory matching steps. Even if we reduce the steps from 20 to 5, the test accuracy obtained on CIFAR-10 and CIFAR-100 only decreases by 1.16% and 0.5%, respectively.

**Impact of trajectory projecting steps.** To test the sensitivity of the trajectory projection steps $K$, we consider CIFAR-10 with Dirichlet 0.01. Figure 5 shows convergence plots for different $K$ configurations. We can observe that our method is relatively stable when $k = 3$ and $k = 5$. However, setting $k = 1$ would lead to slower convergence and a slight performance decrease, and setting $k = 10$ could accelerate the training in the early stage but result in instability and make it difficult to converge.

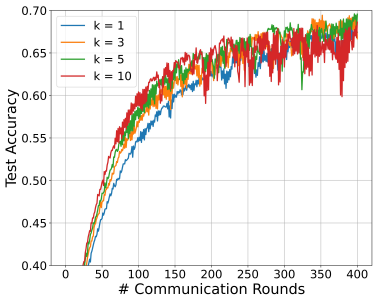



Figure 5: Comparisons of FedPTR with different trajectory projecting steps $K$ on CIFAR-10 with Dirichlet $\alpha = 0.01$.

## 7 Privacy leakage of FedPTR

For privacy leakage analysis of our proposed method, we argue that although FedPTR does not incorporate explicit privacy protection mechanisms, it does not introduce additional privacy vulnerabilities compared to existing federated learning approaches such as FedAvg (McMahan et al., 2017) and FedDyn (Acar et al., 2021). This is because all client-server communications in FedPTR consist solely of model weights and updates, identical to the communication patterns in FedAvg (McMahan et al., 2017) and FedDyn (Acar et al., 2021). To empirically validate that FedPTR does not incur additional privacy leakage, we conduct membership inference attacks (MIA) on the final models trained using different methods. We present MIA results for 10-client federated learning with $\alpha = 0.04$ on the CIFAR-100 dataset in Table 6, employing two well-established MIA techniques: (1) the loss-based method and (2) the Likelihood Ratio Attack (LiRA). The reported accuracy represents the success rate of the MIA method in correctly predicting whether a sample belongs to the original training set. Notably, MIA test accuracy closer to 50% (equivalent to random guessing) indicates minimal privacy leakage. Our results demonstrate that FedPTR achieves comparable privacy preservation levels to other baseline methods, confirming our theoretical analysis.

### 7.1 Inadequacy of Trivial Synthetic Data Integration

In this section, we present a naive approach that directly combines distilled datasets with local data and analyze its limitations. Since MTT (Cazenavette et al., 2022) was originally designed for distilling small synthetic datasets, a straightforward solution for addressing data heterogeneity would be to apply MTT (Cazenavette et al., 2022) to generate synthetic data and combine it with available local data for FedProx (Li et al., 2020) training. However, this trivial approach proves inadequate for addressing data heterogeneity

issues, particularly under highly non-i.i.d. data distributions across clients, as demonstrated in Table 7. The results show that simply augmenting FedProx training with synthetic data from MTT (Cazenavette et al., 2022) yields even worse performance than FedAvg. This degradation occurs because the synthetic data generated by MTT (Cazenavette et al., 2022) in federated settings may inadequately represent the original training data information. Unlike the original MTT (Cazenavette et al., 2022), which can leverage multiple training trajectories and restart strategies, federated settings impose realistic constraints that limit these capabilities. Consequently, direct training on such synthetic data can actually harm performance. In contrast, our approach fundamentally differs from this naive combination. We utilize the auxiliary data produced by MTT (Cazenavette et al., 2022) specifically to guide trajectory prediction for subsequent training steps—precisely the task MTT (Cazenavette et al., 2022) was designed for (i.e., matching recent training trajectories). This targeted usage represents a fundamental departure from simply combining existing methods like FedProx (Li et al., 2020) and MTT (Cazenavette et al., 2022).

Table 6: Membership inference attack for global model with $\alpha = 0.04$ on CIFAR-100 dataset.

| | FedAvg | FedDyn | FedPTR |
|---|---|---|---|
| loss-based | 56.37 | 61.67 | 57.46 |
| LIRA | 61.06 | 68.29 | 65.59 |

Table 7: Comparisons of directly using auxiliary dataset and FedPTR on three datasets.

| | FMNIST | CIFAR-10 | CIFAR-100 |
|---|---|---|---|
| Extra synthetic data | 76.23 | 50.08 | 27.78 |
| FedAvg | 83.69 | 55.45 | 37.32 |
| FedPTR | 85.11 | 69.04 | 44.98 |
| FedPTR-S | 85.48 | 68.16 | 44.82 |

## 8 Conclusion

Data heterogeneity issue is a critical factor for the practical deployment of Federated Learning algorithms. In this work, we propose a novel federated learning framework FedPTR that handles extreme data heterogeneity across local clients. The proposed method allows each client or the server to extract essential global information from the observed global model training trajectory and utilize it to project the next-step model trajectory for local training regularization. The exhaustive experimental results on various benchmark datasets and different non-i.i.d settings clearly demonstrate the effectiveness of our proposed FedPTR in training real-world machine learning models. One potential limitation of FedPTR is that it cannot be directly applied to language tasks since the application of MTT on language tasks is still an open question. We will leave it as our future work.

## References

Durmus Alp Emre Acar, Yue Zhao, Ramon Matas Navarro, Matthew Mattina, Paul N Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization. *arXiv preprint arXiv:2111.04263*, 2021.

Ondrej Bohdal, Yongxin Yang, and Timothy Hospedales. Flexible dataset distillation: Learn labels instead of images. *arXiv preprint arXiv:2006.08572*, 2020.

George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4750–4759, 2022.

Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive personalized federated learning. *Unpublished*, 2020.

Jiawei Du, Yidi Jiang, Vincent TF Tan, Joey Tianyi Zhou, and Haizhou Li. Minimizing the accumulated trajectory error to improve dataset distillation. *arXiv preprint arXiv:2211.11004*, 2022.

Jiawei Du, Qin Shi, and Joey Tianyi Zhou. Sequential subset matching for dataset distillation. *Advances in Neural Information Processing Systems*, 36:67487–67504, 2023.

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning: A meta-learning approach. *Advances in neural information processing systems*, 34, 2020.

Ziqing Fan, Shengchao Hu, Jiangchao Yao, Gang Niu, Ya Zhang, Masashi Sugiyama, and Yanfeng Wang. Locally estimated global perturbations are better than local perturbations for federated sharpness-aware minimization. *arXiv preprint arXiv:2405.18890*, 2024.

Liang Gao, Huazhu Fu, Li Li, Yingwen Chen, Ming Xu, and Cheng-Zhong Xu. Feddc: Federated learning with non-iid data via local drift decoupling and correction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10112–10121, 2022.

Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4367–4375, 2018.

Ziyao Guo, Kai Wang, George Cazenavette, HUI LI, Kaipeng Zhang, and Yang You. Towards lossless dataset distillation via difficulty-aligned trajectory matching. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=rTBL8OhdhH`.

Binxuan Hu, Yujia Gao, Liang Liu, and Huadong Ma. Federated region-learning: An edge computing based framework for urban environment sensing. In *2018 ieee global communications conference (globecom)*, pp. 1–7. IEEE, 2018.

Sai Praneeth Karimireddy, Martin Jaggi, Satyen Kale, Mehryar Mohri, Sashank J Reddi, Sebastian U Stich, and Ananda Theertha Suresh. Mime: Mimicking centralized stochastic algorithms in federated learning. *arXiv preprint arXiv:2008.03606*, 2020a.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pp. 5132–5143. PMLR, 2020b.

Ahmed Khaled, Konstantin Mishchenko, and Peter Richtárik. Tighter theory for local sgd on identical and heterogeneous data. In *International Conference on Artificial Intelligence and Statistics*, pp. 4519–4529. PMLR, 2020.

Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

Junghye Lee, Jimeng Sun, Fei Wang, Shuang Wang, Chi-Hyuck Jun, Xiaoqian Jiang, et al. Privacy-preserving patient similarity learning in a federated environment: development and analysis. *JMIR medical informatics*, 6(2):e7744, 2018.

Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10713–10722, 2021a.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450, 2020.

Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning*, pp. 6357–6368. PMLR, 2021b.

Xiaoxiao Li, Meirui Jiang, Xiaofei Zhang, Michael Kamp, and Qi Dou. Fedbn: Federated learning on non-iid features via local batch normalization. *arXiv preprint arXiv:2102.07623*, 2021c.

Yuhang Li, Tong Liu, Yangguang Cui, Ming Hu, and Xiaoqiang Li. One arrow, two hawks: Sharpness-aware minimization for federated learning via global model trajectory. In *Forty-second International Conference on Machine Learning*, 2025. URL `https://openreview.net/forum?id=80mK2Mqaph`.

Ping Liu, Xin Yu, and Joey Tianyi Zhou. Meta knowledge condensation for federated learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=TDf-XFAwc79`.

Songhua Liu, Kai Wang, Xingyi Yang, Jingwen Ye, and Xinchao Wang. Dataset distillation via factorization. *arXiv preprint arXiv:2210.16774*, 2022.

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.

Renjie Pi, Weizhong Zhang, Yueqi Xie, Jiahui Gao, Xiaoyu Wang, Sunghun Kim, and Qifeng Chen. Dynafed: Tackling client data heterogeneity with global dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12177–12186, 2023.

Zhe Qu, Xingyu Li, Rui Duan, Yao Liu, Bo Tang, and Zhuo Lu. Generalized federated learning via sharpness aware minimization. In *International Conference on Machine Learning*, pp. 18250–18280. PMLR, 2022.

Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and H Brendan McMahan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. *Advances in neural information processing systems*, 30, 2017.

Rui Song, Dai Liu, Dave Zhenyu Chen, Andreas Festag, Carsten Trinitis, Martin Schulz, and Alois Knoll. Federated learning via decentralized dataset distillation in resource-constrained edge environments. *arXiv preprint arXiv:2208.11311*, 2022.

Yan Sun, Li Shen, Shixiang Chen, Liang Ding, and Dacheng Tao. Dynamic regularized sharpness aware minimization in federated learning: Approaching global consistency and smooth landscape. In *International conference on machine learning*, pp. 32991–33013. PMLR, 2023a.

Yan Sun, Li Shen, Tiansheng Huang, Liang Ding, and Dacheng Tao. Fedspeed: Larger local interval, less communication round, and higher generalization accuracy. *arXiv preprint arXiv:2302.10429*, 2023b.

Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. *arXiv preprint arXiv:2002.06440*, 2020a.

Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in neural information processing systems*, 33: 7611–7623, 2020b.

Kai Wang, Bo Zhao, Xiangyu Peng, Zheng Zhu, Shuo Yang, Shuo Wang, Guan Huang, Hakan Bilen, Xinchao Wang, and Yang You. Cafe: Learning to condense dataset by aligning features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12196–12205, 2022a.

Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018.

Xiaofei Wang, Yiwen Han, Chenyang Wang, Qiyang Zhao, Xu Chen, and Min Chen. In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning. *Ieee Network*, 33(5): 156–165, 2019.

Yujia Wang, Lu Lin, and Jinghui Chen. Communication-efficient adaptive federated learning. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 22802–22838. PMLR, 2022b.

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

Yuanhao Xiong, Ruochen Wang, Minhao Cheng, Felix Yu, and Cho-Jui Hsieh. Feddm: Iterative distribution matching for communication-efficient federated learning. *arXiv preprint arXiv:2207.09653*, 2022.

Jing Xu, Sen Wang, Liwei Wang, and Andrew Chi-Chih Yao. Fedcm: Federated learning with client-level momentum. *arXiv preprint arXiv:2106.10874*, 2021.

Xinwei Zhang, Mingyi Hong, Sairaj Dhople, Wotao Yin, and Yang Liu. Fedpd: A federated learning framework with optimal rates and adaptivity to non-iid data. *arXiv preprint arXiv:2005.11418*, 2020.

Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. *arXiv preprint arXiv:2110.04181*, 2021a.

Bo Zhao and Hakan Bilen. Dataset condensation with differentiable siamese augmentation. In *International Conference on Machine Learning*, pp. 12674–12685. PMLR, 2021b.

Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. *ICLR*, 1 (2):3, 2021.

Yanlin Zhou, George Pu, Xiyao Ma, Xiaolin Li, and Dapeng Wu. Distilled one-shot federated learning. *arXiv preprint arXiv:2009.07999*, 2020.

Wei Zhu, Dongjin Song, Yuncong Chen, Wei Cheng, Bo Zong, Takehiko Mizoguchi, Cristian Lumezanu, Haifeng Chen, and Jiebo Luo. Deep federated anomaly detection for multivariate time series data. *arXiv preprint arXiv:2205.04041*, 2022.

# A Server-Side FedPTR

We present our Server-Side FedPTR in Algorithm 3, which only updates the auxiliary dataset through MTT in the global server. At the start of each communication round, the global server first matches training trajectories and optimizes a unified $\widetilde{\mathbf{w}}^{t+1}$. Then, each client will receive both $\mathbf{w}^t$ and $\widetilde{\mathbf{w}}^{t+1}$ for local training. Different from client-side FedPTR, server-side FedPTR only uses random noise to initialize the auxiliary dataset. Therefore, our FedPTR-S maintains privacy-preserving while also making participating devices free from extra computation burdens.

# B Additional Experiments

## B.1 Comparison of Different Initialization Strategies.

As presented in Table 8, we compare our strategy that initializes the auxiliary dataset using (partial) local client data with the random initialization strategy. It can be seen that FedPTR using our proposed initialization strategy obtains better performance on CIFAR-10 and CIFAR-100 with the improvement of 1.54% and 0.47%, respectively. On the other hand, our FedPTR still maintains excellent performance even after adopting the random initialization strategy, which supports the design of server-side FedPTR (FedPTR-S) that performs MTT and optimizes the auxiliary dataset with random initialization in the global server.

Table 8: Comparisons of different strategies of auxiliary dataset initialization on CIFAR-10 with Dirichlet $\alpha = 0.01$.

| Dataset | FMNIST | CIFAR-10 | CIFAR-100 |
|---|---|---|---|
| FedPTR - random init | 85.13 | 67.50 | 44.51 |
| FedPTR - our init | 85.11 | 69.04 | 44.98 |

**Algorithm 3** Server-Side FedPTR (FedPTR-S)

**Input:** $T$, $m$, $\eta$, $\lambda$.
Initialize $\widetilde{\mathcal{D}}$ for server
**for** $t = 0$ **to** $T - 1$ **do**
  Server starts to match training trajectories:
  **if** $t > m$ **then**
    $\widetilde{\mathcal{D}} \leftarrow \mathrm{MTT}(\mathbf{w}^{t-m}, \mathbf{w}^{t}, \widetilde{\mathcal{D}})$
    **for** $k = 0$ **to** $K - 1$ **do**
      $\widetilde{\mathbf{w}}^{t+1} \leftarrow \widetilde{\mathbf{w}}^{t+1} - \eta \nabla \mathcal{L}(\widetilde{\mathbf{w}}^{t+1}; \widetilde{\mathcal{D}})$
    **end for**
    Set $\widetilde{\lambda} = \lambda$
  **else**
    $\widetilde{\mathbf{w}}^{t+1} = \mathbf{w}^{t}$
    Set $\widetilde{\lambda} = 0$
  **end if**
  **for** each client $i \in [N]$ **do**
    Client $i$ receives $\mathbf{w}^{t}$ and $\widetilde{\mathbf{w}}^{t+1}$ from the server
    Solve $\mathbf{w}_i^{t+1}$ as a $\gamma$-inexactness minimizer of:
    $\mathbf{w}_i^{t+1} \approx \underset{\mathbf{w}}{\operatorname{argmin}}\, \mathcal{L}(\mathbf{w}; \mathcal{D}_i) + \frac{\widetilde{\lambda}}{2} \cdot \|\mathbf{w} - \widetilde{\mathbf{w}}^{t+1}\|_2^2$
    Client sends $\mathbf{w}_i^{t+1}$ to server
  **end for**
  Server aggregates $\{\mathbf{w}_i^{t+1}\}_{i=1}^{N}$ to get $\mathbf{w}^{t+1}$
**end for**

### B.2 More Results of Reducing the Updates of Auxiliary Dataset

For the update frequency, we have conducted ablation studies on client's side as shown in Figure 2. Following the same setting, here we present our experiments on server's side in Table 9, which shows that the server-side method FedPTR-S can still maintain a relatively stable performance with reduced updates of auxiliary datasets.

Table 9: Comparisons of FedPTR-S with different update frequencies of the auxiliary dataset on CIFAR-10 with Dirichlet $\alpha = 0.01$.

| Update Frequency | Test Acc |
|---|---|
| per 1 round | 68.16 |
| per 4 round | 65.42 |
| per 10 round | 65.97 |
| per 40 round | 65.82 |
| one-shot | 65.34 |

### B.3 More Results of Changing the Size of Auxiliary Dataset.

We have conducted comparisons of the client's side method FedPTR with different sizes of the auxiliary dataset on CIFAR-10 with Dirichlet $\alpha = 0.01$ as shown in Table 4. Here we present the results on server's side method FedPTR-S with the same setting as shown in Table 10. We can observe FedPTR-S could obtain better performance with more auxiliary data, similar to FedPTR.

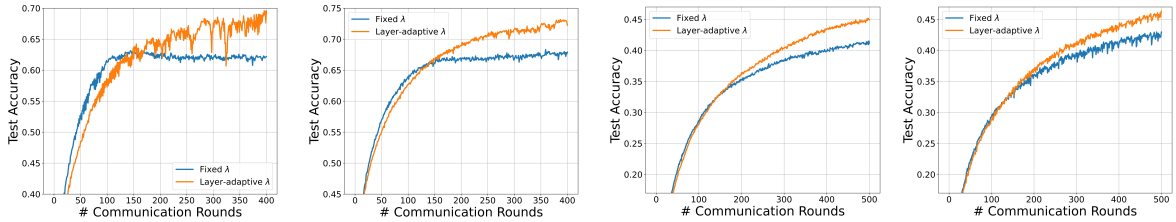
### B.4 More Results of Layer-adaptive $\lambda$ V.S. Fixed $\lambda$.

**Layer-adaptive $\lambda$ V.S. Fixed $\lambda$.** In Table 1, we have already shown the effectiveness of layer-adaptive $\lambda$, and here we give a further analysis. As shown in Figure 6, we present the complete convergence plots

Table 10: Comparisons of FedPTR-S with different sizes of the auxiliary dataset on CIFAR-10 with Dirichlet $\alpha = 0.01$.

| Size of $\widetilde{\mathcal{D}}$ | Test Acc |
|---|---|
| 10 | 67.47 |
| 50 | 67.64 |
| 100 | 68.16 |
| 150 | 69.11 |
| 200 | 69.75 |

of FedPTR with layer-adaptive $\lambda$ and fixed $\lambda$ on CIFAR-10 and CIFAR-100 with Dirichlet $\alpha = 0.01$ and Dirichlet $\alpha = 0.04$. It can be seen that adopting layer-adaptive $\lambda$ slightly falls behind at the beginning of the training but obtains better performance in the late phase of the training, indicating that layer-adaptive $\lambda$ can help converge to a better stationary point.




(a) CIFAR-10, Dir(0.01) (b) CIFAR-10, Dir(0.04) (c) CIFAR-100, Dir(0.01) (d) CIFAR-100, Dir(0.04)

Figure 6: Test accuracy on CIFAR-10 and CIFAR-100 against the communication rounds, obtained by FedPTR with different regularizations.

### B.5 Experiments on TinyImagenet

We additionally conduct experiments to test our method on larger TinyImageNet data to demonstrate that our method can indeed extend to larger-scale experiments. We summarize the experimental results with 10 clients and Dir $\alpha = 0.01$ in Table 11. The results suggest that our method still outperforms other baselines.

Table 11: Comparison of test accuracy for FedPTR (FedPTR-S) and other baselines on TinyImageNet.

| FedAvg | FedProx | Scaffold | FedDyn | FedDC | FedPTR | FedPTR-S |
|---|---|---|---|---|---|---|
| 27.68 | 27.11 | 21.55 | 27.49 | 23.81 | 28.6 | **29.6** |

## C Detailed Description of Data Sampling

As we have mentioned in experimental settings, for all experiments, we adopt Dirichlet data sampling methods, which are commonly used in federated learning papers(Acar et al., 2021; Wang et al., 2020a). For Dirichlet sampling methods, each client $j$ is allocated a proportion of the samples from each label $k$ according to Dirichlet distribution. Specifically, we sample $p_k \sim Dir_N(\alpha)$ and allocate a $p_{kj}$ proportion of data samples of class $k$ to client $j$, where $Dir_N(\alpha)$ denotes the Dirichlet distribution and $\alpha > 0$ is a concentration parameter. A smaller $\alpha$ indicates a higher non-i.i.d. degree when sampling data, and we focused on the high-level data heterogeneity with $\alpha = 0.01$ and $\alpha = 0.04$. Table 12 shows a typical Dirichlet sampling result for different settings. We can observe that when choosing 40 clients on CIFAR-10, it can easily assign 0 data samples to certain clients due to low value and a small number of classes. That is the reason why we do not conclude experiments with more clients on CIFAR-10 and FashionMNIST.

Table 12: Dirichlet sampling result for different settings.

| Dataset | $\alpha$ | # clients | # samples in each client |
|---|---|---|---|
| FMNIST | 0.01 | 10 | $12032, 4244, 6003, 6037, 6002, 6165, 5963, 5760, 2602, 5192$ |
| CIFAR-10 | 0.01 | 10 | $2972, 9419, 2936, 5779, 4111, 5003, 4949, 4979, 5615, 4237$ |
| CIFAR-100 | 0.01 | 10 | $2690, 5209, 4764, 6463, 4467, 5679, 3983, 5910, 5497, 5338$ |
| CIFAR-100 | 0.01 | 40 | $735, 1371, 1365, 761, 1123, 2194, 2399, 531, 848, 1549,$ $2378, 569, 2007, 984, 518, 608, 1250, 780, 2601, 2784,$ $1408, 1831, 1108, 2166, 523, 814, 1281, 579, 1067, 1172,$ $803, 1085, 902, 2603, 1237, 940, 588, 683, 1246, 609$ |
| CIFAR-10 | 0.01 | 40 | $0, 3133, 0, 274, 373, 0, 296, 1709, 1114, 4636,$ $5287, 0, 1469, 1684, 0, 415, 3, 3950, 718, 46,$ $3355, 29, 7318, 1033, 0, 0, 918, 0, 1342, 0,$ $1, 0, 4979, 101, 1397, 1615, 5, 2798, 2, 0$ |

## D Proof of Theorem 5.4.

For notation convenience, we conclude $\mathcal{L}(\mathbf{w};\mathcal{D}_i)$ as $\mathcal{L}_i(\mathbf{w})$ and $L(\mathbf{w};\widetilde{\mathcal{D}}_i)$ as $\widetilde{\mathcal{L}}_i(\mathbf{w})$ in the following.

**Lemma D.1.** For $h_i(\mathbf{w}) := \mathcal{L}_i(\mathbf{w}) + \frac{\lambda}{2}\|\mathbf{w} - \mathbf{w}^t + \eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\|^2$, there is $\|\nabla h_i(\mathbf{w}) - \nabla h_i(\mathbf{v})\| \leq (L+\lambda)\|\mathbf{w} - \mathbf{v}\|, \forall \mathbf{w}, \mathbf{v} \in \mathbb{R}^d$.

*Proof of Lemma D.1.*

$$\begin{aligned}
&\|\nabla h_i(\mathbf{w}) - \nabla h_i(\mathbf{y})\| \\
&= \|\nabla\mathcal{L}_i(\mathbf{w}) - \nabla\mathcal{L}_i(\mathbf{y}) + \lambda\mathbf{w} - \lambda\mathbf{y}\| \\
&\leq \|\nabla\mathcal{L}_i(\mathbf{w}) - \nabla\mathcal{L}_i(\mathbf{y})\| + \lambda\|\mathbf{w} - \mathbf{y}\| \\
&\leq (L+\lambda)\|\mathbf{w} - \mathbf{y}\|.
\end{aligned} \tag{D.1}$$

This concludes the proof. □

*Proof of Theorem 5.4.* Recall the notation and the objective function, we have the following for the steps,

$$\begin{aligned}
\mathbf{w}_i^{t,q+1} &= \mathbf{w}_i^{t,q} - \eta(\mathbf{g}_i^{t,q} + \lambda(\mathbf{w}_i^{t,q} - \widetilde{\mathbf{w}}_i^{t+1})) \\
&= \mathbf{w}_i^{t,q} - \eta(\mathbf{g}_i^{t,q} + \lambda(\mathbf{w}_i^{t,q} - \mathbf{w}_t + \eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t))),
\end{aligned} \tag{D.2}$$

where $q = 0, 1, ..., Q-1$ indicates the local step. We also have

$$\begin{aligned}
h_i(\mathbf{w}) &= f_i(\mathbf{w}) + \frac{\lambda}{2}\|\mathbf{w} - \mathbf{w}^t + \eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\|^2, \\
\nabla h_i(\mathbf{w}) &= \nabla\mathcal{L}_i(\mathbf{w}) + \lambda(\mathbf{w} - \mathbf{w}_t + \eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)).
\end{aligned} \tag{D.3}$$

Denote $\mathbf{e}_i^{t,q+1}$ as follows

$$\mathbf{e}_i^{t,q+1} = \nabla\mathcal{L}_i(\mathbf{w}_i^{t,q}) + \lambda(\mathbf{w}_i^{t,q} - \mathbf{w}_t + \eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)) = \nabla h_i(\mathbf{w}_i^{t,q}), \tag{D.4}$$

by the notation of $\gamma$-inexactness, there is

$$\begin{aligned}
\|\mathbf{e}_i^{t,q+1}\|^2 &\leq \gamma^2\|\nabla\mathcal{L}_i(\mathbf{w}^t - \eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t))\|^2 \\
&= \gamma^2\|\nabla\mathcal{L}_i(\widetilde{\mathbf{w}}_i^{t+1})\|^2 \\
&= \gamma^2\|\nabla\mathcal{L}_i(\widetilde{\mathbf{w}}_i^{t+1}) - \nabla\mathcal{L}_i(\mathbf{w}^t) + \nabla\mathcal{L}_i(\mathbf{w}^t)\|^2 \\
&\leq 2\gamma^2\|\nabla\mathcal{L}_i(\mathbf{w}^t)\|^2 + 2\gamma^2L^2\|\widetilde{\mathbf{w}}_i^{t+1} - \mathbf{w}^t\|^2 \\
&= 2\gamma^2\|\nabla\mathcal{L}_i(\mathbf{w}^t)\|^2 + 2\gamma^2L^2\eta^2\|\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\|^2,
\end{aligned} \tag{D.5}$$

where the first equation holds by the definition of distilled model weight $\widetilde{\mathbf{w}}_i^{t+1}$, the second inequality holds by the $L$-smoothness of $f_i$, and the last equation is from the definition of $\widetilde{\mathbf{w}}_i^{t+1}$. Then we have

$$\begin{aligned}\mathbb{E}_t[\mathbf{w}^{t+1} - \mathbf{w}^t] &= \frac{1}{N}\sum_{i\in[N]} \mathbb{E}_t[\mathbf{w}_i^{t,Q} - \mathbf{w}^t] \\ &= -\frac{1}{\lambda}\frac{1}{N}\sum_{i\in[N]} \mathbb{E}_t[\mathbf{g}_i^{t,Q} + \lambda\eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t) - \mathbf{e}_i^{t,Q}]. \end{aligned} \tag{D.6}$$

Define $\mu = \lambda - L_- > 0$ $[L_-]$ and $\mathbf{w}_{i,*}^{t+1} = \operatorname{argmin}_{\mathbf{w}} h_i(\mathbf{w};\mathbf{w}^t)$, then we have

$$\begin{aligned}\|\mathbf{w}_{i,*}^{t+1} - \mathbf{w}_i^{t,Q}\| &\le \frac{1}{\mu}\|\nabla h_i(\mathbf{w}_{i,*}^{t+1}) - \nabla h_i(\mathbf{w}_i^{t,Q})\| \\ &= \frac{1}{\mu}\|\nabla h_i(\mathbf{w}_i^{t,Q})\| \\ &\le \frac{\gamma}{\mu}\|\nabla h_i(\widetilde{\mathbf{w}}_i^{t+1})\| \\ &\le \frac{\gamma}{\mu}\|\nabla h_i(\widetilde{\mathbf{w}}_i^{t+1}) - \nabla h_i(\mathbf{w}^t)\| + \frac{\gamma}{\mu}\|\nabla h_i(\mathbf{w}^t)\| \\ &\le \frac{\gamma}{\mu}(L+\lambda)\|\widetilde{\mathbf{w}}_i^{t+1} - \mathbf{w}^t\| + \frac{\gamma}{\mu}\|\nabla\mathcal{L}_i(\mathbf{w}^t) + \lambda\eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\| \\ &= \frac{\gamma\eta}{\mu}(L+\lambda)\|\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\| + \frac{\gamma}{\mu}\|\nabla\mathcal{L}_i(\mathbf{w}^t) + \lambda\eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\|, \end{aligned} \tag{D.7}$$

where the first inequality holds by the $\mu$-strong convexity of $h_i$, the first equation holds by $\nabla h_i(\mathbf{w}_{i,*}^{t+1}) = 0$ since $\mathbf{w}_{i,*}^{t+1} = \operatorname{argmin}_{\mathbf{w}} h_i(\mathbf{w};\mathbf{w}^t)$, and the second inequality holds by $\gamma$-inexactness. The third inequality follows the triangle inequality, the forth one follows Lemma D.1. Similarly by the $\mu$-strongly convexity, we have

$$\|\mathbf{w}_{i,*}^{t+1} - \mathbf{w}^t\| \le \frac{1}{\mu}\|\nabla h_i(\mathbf{w}^t)\| = \frac{1}{\mu}\|\nabla\mathcal{L}_i(\mathbf{w}^t) + \lambda\eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\|. \tag{D.8}$$

By the triangle inequality, we have

$$\begin{aligned}\|\mathbf{w}_i^{t,Q} - \mathbf{w}^t\| &\le \|\mathbf{w}_{i,*}^{t+1} - \mathbf{w}^t\| + \|\mathbf{w}_{i,*}^{t+1} - \mathbf{w}_i^{t,Q}\| \\ &\le \frac{1+\gamma}{\mu}\|\nabla\mathcal{L}_i(\mathbf{w}^t) + \lambda\eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\| + \frac{\gamma\eta(L+\lambda)}{\mu}\|\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\| \end{aligned} \tag{D.9}$$

and

$$\begin{aligned}\|\mathbf{w}_i^{t,Q} - \mathbf{w}^t\|^2 &\le 2\|\mathbf{w}_{i,*}^{t+1} - \mathbf{w}^t\|^2 + 2\|\mathbf{w}_{i,*}^{t+1} - \mathbf{w}_i^{t,Q}\|^2 \\ &\le \frac{2+4\gamma^2}{\mu^2}\|\nabla\mathcal{L}_i(\mathbf{w}^t) + \lambda\eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\|^2 + \frac{4\gamma^2\eta^2(L+\lambda)^2}{\mu^2}\|\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\|^2. \end{aligned} \tag{D.10}$$

Since the objective function $f$ is $L$ smooth, taking conditional expectation at time $t$, (abbreviate $\mathbb{E}_t$ as $\mathbb{E}$) we have

$$\begin{aligned}\mathbb{E}_t[\mathcal{L}(\mathbf{w}^{t+1})] &\le \mathcal{L}(\mathbf{w}_t) + \mathbb{E}_t\langle\nabla\mathcal{L}(\mathbf{w}_t), \mathbf{w}^{t+1} - \mathbf{w}^t\rangle + \frac{L}{2}\mathbb{E}_t[\|\mathbf{w}^{t+1} - \mathbf{w}^t\|^2] \\ &\le \mathcal{L}(\mathbf{w}_t) + \underbrace{\mathbb{E}\left\langle\nabla\mathcal{L}(\mathbf{w}_t), -\frac{1}{\lambda}\frac{1}{N}\sum_{i\in[N]}[\mathbf{g}_i^{t,Q} + \lambda\eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t) - \mathbf{e}_i^{t,Q}]\right\rangle}_{I_1} + \underbrace{\frac{L}{2}\mathbb{E}[\|\mathbf{w}^{t+1} - \mathbf{w}^t\|^2]}_{I_2}. \end{aligned} \tag{D.11}$$

For term $I_2$, we have

$$
\begin{aligned}
I_2 &= \frac{L}{2}\mathbb{E}_t[\|\mathbf{w}^{t+1}-\mathbf{w}^t\|^2] \\
&= \frac{L}{2}\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i\in[N]}\mathbf{w}_i^{t,Q}-\mathbf{w}^t\right\|^2\right] \\
&\leq \frac{L}{2}\frac{1}{N}\sum_{i\in[N]}\mathbb{E}[\|\mathbf{w}_i^{t,Q}-\mathbf{w}^t\|^2] \\
&\leq \frac{1+2\gamma^2}{\mu^2}\frac{L}{N}\sum_{i\in[N]}\mathbb{E}[\|\nabla\mathcal{L}_i(\mathbf{w}^t)+\lambda\eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\|^2]+\frac{4\gamma^2\eta^2L^2}{\mu^2}\frac{L}{N}\sum_{i\in[N]}\mathbb{E}[\|\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\|^2] \\
&\leq \frac{2+4\gamma^2}{\mu^2}\frac{L}{N}\sum_{i\in[N]}\mathbb{E}[\|\nabla\mathcal{L}_i(\mathbf{w}^t)\|^2]+\left(\frac{4\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(2+4\gamma^2)\lambda^2\eta^2}{\mu^2}\right)\frac{L}{N}\sum_{i\in[N]}\mathbb{E}[\|\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\|^2] \\
&\leq \frac{LB^2(2+4\gamma^2)}{\mu^2}\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}^t)\|^2]+\left(\frac{4\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(2+4\gamma^2)\lambda^2\eta^2}{\mu^2}\right)\frac{L}{N}\sum_{i\in[N]}\mathbb{E}[\|\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\mp\nabla\mathcal{L}(\mathbf{w}^t)\|^2],
\end{aligned}
\tag{D.12}
$$

where the first and third inequalities hold by Cauchy inequality, the forth one holds by Assumption 5.2, and the last one holds by Assumption 5.3. Then we have

$$
\begin{aligned}
&\left(\frac{4\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(2+4\gamma^2)\lambda^2\eta^2}{\mu^2}\right)\frac{L}{N}\sum_{i\in[N]}\mathbb{E}[\|\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\mp\nabla\mathcal{L}(\mathbf{w}^t)\|^2] \\
&\leq 2\left(\frac{4\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(2+4\gamma^2)\lambda^2\eta^2}{\mu^2}\right)\frac{L}{N}\sum_{i\in[N]}\mathbb{E}[\|\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)-\nabla\mathcal{L}(\mathbf{w}^t)\|^2] \\
&\quad+2\left(\frac{4\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(2+4\gamma^2)\lambda^2\eta^2}{\mu^2}\right)\frac{L}{N}\sum_{i\in[N]}\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}^t)\|^2] \\
&\leq \left(\frac{8\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(4+8\gamma^2)\lambda^2\eta^2}{\mu^2}\right)L\sigma_d^2+\left(\frac{8\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(4+8\gamma^2)\lambda^2\eta^2}{\mu^2}\right)L\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}^t)\|^2].
\end{aligned}
\tag{D.13}
$$

Thus

$$
\begin{aligned}
I_2 \leq &\left(\frac{2+4\gamma^2}{\mu^2}+\frac{8\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(4+8\gamma^2)\lambda^2\eta^2}{\mu^2}\right)L\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}^t)\|^2] \\
&+\left(\frac{8\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(4+8\gamma^2)\lambda^2\eta^2}{\mu^2}\right)L\sigma_d^2.
\end{aligned}
\tag{D.14}
$$

Similarly, it also implies

$$
\begin{aligned}
\mathbb{E}[\|\mathbf{w}^{t+1}-\mathbf{w}^t\|^2] &\leq \frac{1}{N}\sum_{i\in[N]}\mathbb{E}[\|\mathbf{w}_i^{t+1}-\mathbf{w}^t\|^2] \\
&\leq 2\left(\frac{2+4\gamma^2}{\mu^2}+\frac{8\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(4+8\gamma^2)\lambda^2\eta^2}{\mu^2}\right)\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}^t)\|^2] \\
&\quad+2\left(\frac{8\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(4+8\gamma^2)\lambda^2\eta^2}{\mu^2}\right)\sigma_d^2.
\end{aligned}
\tag{D.15}
$$

For term $I_1$, we have

$$
\begin{aligned}
I_1 &= \mathbb{E}\Big\langle \nabla\mathcal{L}(\mathbf{w}_t), -\frac{1}{\lambda}\frac{1}{N}\sum_{i\in[N]}[\mathbf{g}_i^{t,Q} + \lambda\eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t) - \mathbf{e}_i^{t,Q}] + \frac{1}{\lambda}\nabla\mathcal{L}(\mathbf{w}^t) - \frac{1}{\lambda}\nabla\mathcal{L}(\mathbf{w}^t)\Big\rangle \\
&= -\frac{1}{\lambda}\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}_t)\|^2] + \mathbb{E}\Big\langle \nabla\mathcal{L}(\mathbf{w}_t), -\frac{1}{\lambda}\frac{1}{N}\sum_{i\in[N]}[\nabla\mathcal{L}_i(\mathbf{w}_i^{t,Q}) + \lambda\eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t) - \mathbf{e}_i^{t,Q}] + \frac{1}{\lambda}\nabla\mathcal{L}(\mathbf{w}^t)\Big\rangle \\
&= -\frac{1}{\lambda}\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}_t)\|^2] + \mathbb{E}\Big\langle \nabla\mathcal{L}(\mathbf{w}_t), -\frac{1}{\lambda}\frac{1}{N}\sum_{i\in[N]}[\nabla\mathcal{L}_i(\mathbf{w}_i^{t,Q}) + \lambda\eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t) - \mathbf{e}_i^{t,Q}] + \frac{1}{\lambda}\frac{1}{N}\sum_{i\in[N]}\nabla\mathcal{L}_i(\mathbf{w}^t)\Big\rangle \\
&= -\frac{1}{\lambda}\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}_t)\|^2] + \frac{1}{2\lambda}\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}_t)\|^2] \\
&\quad + \underbrace{\frac{1}{2\lambda}\mathbb{E}\Big[\Big\|-\frac{1}{N}\sum_{i\in[N]}[\nabla\mathcal{L}_i(\mathbf{w}_i^{t,Q}) + \lambda\eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t) - \mathbf{e}_i^{t,Q}] + \nabla\mathcal{L}(\mathbf{w}^t)\Big\|^2\Big]}_{I_3} \\
&\quad - \frac{1}{2\lambda}\mathbb{E}\Big[\Big\|-\frac{1}{N}\sum_{i\in[N]}[\nabla\mathcal{L}_i(\mathbf{w}_i^{t,Q}) + \lambda\eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t) - \mathbf{e}_i^{t,Q}]\Big\|^2\Big],
\end{aligned}
\tag{D.16}
$$

where the second equation holds due to the unbiased-ness of stochastic gradient $\mathbf{g}_i^{t,Q}$, the third equation holds by the definition of global objective function, the forth one holds by the fact of $\langle \mathbf{a}, \boldsymbol{b}\rangle = \frac{1}{2}[\|\mathbf{a}\|^2 + \|\boldsymbol{b}\|^2 - \|\mathbf{a}-\boldsymbol{b}\|^2]$. For $I_3$ term, we have

$$
\begin{aligned}
I_3 &= \frac{1}{2\lambda}\mathbb{E}\Big[\Big\|-\frac{1}{N}\sum_{i\in[N]}\nabla\mathcal{L}_i(\mathbf{w}_i^{t,Q}) - \frac{1}{N}\sum_{i\in[N]}\lambda\eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t) + \frac{1}{N}\sum_{i\in[N]}\mathbf{e}_i^{t,Q} + \frac{1}{N}\sum_{i\in[N]}\nabla\mathcal{L}_i(\mathbf{w}^t)\Big\|^2\Big] \\
&\le \frac{1}{\lambda}\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{i\in[N]}\nabla\mathcal{L}_i(\mathbf{w}^t) - \frac{1}{N}\sum_{i\in[N]}\nabla\mathcal{L}_i(\mathbf{w}_i^{t,Q})\Big\|^2\Big] + \frac{1}{\lambda}\mathbb{E}\Big[\Big\|-\frac{1}{N}\sum_{i\in[N]}\lambda\eta\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t) + \frac{1}{N}\sum_{i\in[N]}\mathbf{e}_i^{t,Q}\Big\|^2\Big] \\
&\le \frac{L^2}{\lambda}\frac{1}{N}\sum_{i\in[N]}\mathbb{E}[\|\mathbf{w}_i^{t,Q} - \mathbf{w}^t\|^2] + \frac{2}{\lambda}\mathbb{E}\Big[\Big\|\frac{\lambda\eta}{N}\sum_{i\in[N]}\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\Big\|^2\Big] + \frac{2}{\lambda}\frac{1}{N}\sum_{i\in[N]}\mathbb{E}[\|\mathbf{e}_i^{t,Q}\|^2] \\
&\le \frac{L^2}{\lambda}\Big[\Big(\frac{4+8\gamma^2}{\mu^2} + \frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2} + \frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\Big)\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}^t)\|^2] \\
&\quad + \Big(\frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2} + \frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\Big)\sigma_d^2\Big] \\
&\quad + \frac{2\lambda\eta^2}{N}\sum_{i\in[N]}\mathbb{E}[\|\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\|^2] + \frac{2}{\lambda}\frac{1}{N}\sum_{i\in[N]}\big[2\gamma^2\mathbb{E}[\|\nabla\mathcal{L}_i(\mathbf{w}^t)\|^2] + 2\gamma^2L^2\eta^2\mathbb{E}[\|\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\|^2]\big] \\
&\le \frac{L^2}{\lambda}\Big[\Big(\frac{4+8\gamma^2}{\mu^2} + \frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2} + \frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\Big)\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}^t)\|^2] \\
&\quad + \Big(\frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2} + \frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\Big)\sigma_d^2\Big] \\
&\quad + \Big(\frac{2\lambda\eta^2}{N} + \frac{4\gamma^2L^2\eta^2}{\lambda N}\Big)\sum_{i\in[N]}\mathbb{E}[\|\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\|^2] + \frac{4\gamma^2}{\lambda}\frac{1}{N}\sum_{i\in[N]}\mathbb{E}[\|\nabla\mathcal{L}_i(\mathbf{w}^t)\|^2],
\end{aligned}
\tag{D.17}
$$

where the first and last inequalities hold by Cauchy inequality, and the second one holds by $L$-smoothness and Cauchy inequality as well. The third one is due to the bound for $\|\mathbf{w}_i^{t,Q} - \mathbf{w}^t\|^2$ and the $\gamma$-inexactness of

$\mathbf{e}_i^{t,Q}$. Then by the bounded gradient dissimilarity and bounded distilled gradient, organizing pieces, we have

$$
\begin{aligned}
I_3 \leq & \frac{L^2}{\lambda}\Bigg[\left(\frac{4+8\gamma^2}{\mu^2}+\frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\right)\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}^t)\|^2] \\
& +\left(\frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\right)\sigma_d^2\Bigg] \\
& +\left(\frac{2\lambda\eta^2}{N}+\frac{4\gamma^2L^2\eta^2}{\lambda N}\right)\sum_{i\in[N]}\mathbb{E}[\|\nabla\widetilde{\mathcal{L}}_i(\mathbf{w}^t)\|^2]+\frac{4\gamma^2}{\lambda}\frac{1}{N}\sum_{i\in[N]}\mathbb{E}[\|\nabla\mathcal{L}_i(\mathbf{w}^t)\|^2] \\
\leq & \frac{L^2}{\lambda}\Bigg[\left(\frac{4+8\gamma^2}{\mu^2}+\frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\right)\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}^t)\|^2] \\
& +\left(\frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\right)\sigma_d^2\Bigg]+\left(4\lambda\eta^2+\frac{8\gamma^2L^2\eta^2}{\lambda}\right)\sigma_d^2 \\
& +\left(\frac{4\lambda\eta^2}{N}+\frac{8\gamma^2L^2\eta^2}{\lambda N}\right)\sum_{i\in[N]}\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}^t)\|^2]+\frac{4\gamma^2}{\lambda N}\sum_{i\in[N]}\mathbb{E}[\|\nabla\mathcal{L}_i(\mathbf{w}^t)\|^2] \\
\leq & \left[\frac{L^2}{\lambda}\left(\frac{4+8\gamma^2}{\mu^2}+\frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\right)+\left(\frac{4\lambda\eta^2}{N}+\frac{8\gamma^2L^2\eta^2}{\lambda N}\right)+\frac{4\gamma^2}{\lambda N}B^2\right]\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}^t)\|^2] \\
& +\left[\frac{L^2}{\lambda}\left(\frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\right)+\left(4\lambda\eta^2+\frac{8\gamma^2L^2\eta^2}{\lambda}\right)\right]\sigma_d^2. \qquad \text{(D.18)}
\end{aligned}
$$

Therefore, by merging pieces together, we have

$$
\begin{aligned}
& \mathbb{E}[\mathcal{L}(\mathbf{w}_{t+1})]-\mathbb{E}[\mathcal{L}(\mathbf{w}_t)] \\
& \leq \Bigg[-\frac{1}{2\lambda}+\frac{L^2}{\lambda}\left(\frac{4+8\gamma^2}{\mu^2}+\frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\right)+\frac{4\lambda\eta^2}{N}+\frac{8\gamma^2L^2\eta^2}{\lambda N}+\frac{4\gamma^2B^2}{\lambda N} \\
& \quad +\frac{L(2+4\gamma^2)}{\mu^2}+\frac{8L\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{L(4+8\gamma^2)\lambda^2\eta^2}{\mu^2}\Bigg]\mathbb{E}\|\nabla\mathcal{L}(\mathbf{w}^t)\|^2 \\
& \quad +\left[\frac{L^2}{\lambda}\left(\frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\right)+4\lambda\eta^2+\frac{8\gamma^2L^2\eta^2}{\lambda}+L\left(\frac{8\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(4+8\gamma^2)\lambda^2\eta^2}{\mu^2}\right)\right]\sigma_d^2, \qquad \text{(D.19)}
\end{aligned}
$$

summing up from $t=0$ to $T-1$, we have

$$
\begin{aligned}
& \mathbb{E}[\mathcal{L}(\mathbf{w}_T)]-\mathbb{E}[\mathcal{L}(\mathbf{w}_0)] \\
& \leq \Bigg[-\frac{1}{2\lambda}+\frac{L^2}{\lambda}\left(\frac{4+8\gamma^2}{\mu^2}+\frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\right)+\frac{4\lambda\eta^2}{N}+\frac{8\gamma^2L^2\eta^2}{\lambda N}+\frac{4\gamma^2B^2}{\lambda N} \\
& \quad +\frac{L(2+4\gamma^2)}{\mu^2}+\frac{8L\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{L(4+8\gamma^2)\lambda^2\eta^2}{\mu^2}\Bigg]\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}^t)\|^2] \\
& \quad +T\left[\frac{L^2}{\lambda}\left(\frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\right)+4\lambda\eta^2+\frac{8\gamma^2L^2\eta^2}{\lambda}+L\left(\frac{8\gamma^2\eta^2(L+\lambda)^2}{\mu^2}+\frac{(4+8\gamma^2)\lambda^2\eta^2}{\mu^2}\right)\right]\sigma_d^2, \qquad \text{(D.20)}
\end{aligned}
$$

then

$$\begin{aligned}
&\bigg[\frac{1}{2\lambda} - \frac{L^2}{\lambda}\bigg(\frac{4+8\gamma^2}{\mu^2} + \frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2} + \frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\bigg) - \frac{4\lambda\eta^2}{N} - \frac{8\gamma^2L^2\eta^2}{\lambda N} - \frac{4\gamma^2B^2}{\lambda N} \\
&\quad - \frac{L(2+4\gamma^2)}{\mu^2} - \frac{8L\gamma^2\eta^2(L+\lambda)^2}{\mu^2} - \frac{L(4+8\gamma^2)\lambda^2\eta^2}{\mu^2}\bigg]\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}^t)\|^2] \\
&\le \mathbb{E}[\mathcal{L}(\mathbf{w}_0)] - \mathbb{E}[\mathcal{L}(\mathbf{w}_T)] + T\bigg[\frac{L^2}{\lambda}\bigg(\frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2} + \frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\bigg) \\
&\quad + 4\lambda\eta^2 + \frac{8\gamma^2L^2\eta^2}{\lambda} + L\bigg(\frac{8\gamma^2\eta^2(L+\lambda)^2}{\mu^2} + \frac{(4+8\gamma^2)\lambda^2\eta^2}{\mu^2}\bigg)\bigg]\sigma_d^2, \qquad \text{(D.21)}
\end{aligned}$$

hence we have

$$\begin{aligned}
&\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}^t)\|^2] \\
&\le \bigg[\frac{1}{2\lambda} - \frac{L^2}{\lambda}\bigg(\frac{4+8\gamma^2}{\mu^2} + \frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2} + \frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\bigg) - \frac{4\lambda\eta^2}{N} - \frac{8\gamma^2L^2\eta^2}{\lambda N} - \frac{4\gamma^2B^2}{\lambda N} \\
&\quad - \frac{L(2+4\gamma^2)}{\mu^2} - \frac{8L\gamma^2\eta^2(L+\lambda)^2}{\mu^2} - \frac{L(4+8\gamma^2)\lambda^2\eta^2}{\mu^2}\bigg]^{-1}\frac{\mathbb{E}[\mathcal{L}(\mathbf{w}_0)] - \mathbb{E}[\mathcal{L}(\mathbf{w}_T)]}{T} \\
&\quad + \bigg[\frac{L^2}{\lambda}\bigg(\frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2} + \frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\bigg) + 4\lambda\eta^2 + \frac{8\gamma^2L^2\eta^2}{\lambda} + L\bigg(\frac{8\gamma^2\eta^2(L+\lambda)^2}{\mu^2} + \frac{(4+8\gamma^2)\lambda^2\eta^2}{\mu^2}\bigg)\bigg]\sigma_d^2. \qquad \text{(D.22)}
\end{aligned}$$

If we assume that $\eta \le \frac{1}{2\lambda}$ and $\gamma \to 0$, then we have

$$\begin{aligned}
&\bigg[\frac{1}{2\lambda} - \frac{L^2}{\lambda}\bigg(\frac{4+8\gamma^2}{\mu^2} + \frac{16\gamma^2\eta^2(L+\lambda)^2}{\mu^2} + \frac{(8+16\gamma^2)\lambda^2\eta^2}{\mu^2}\bigg) - \frac{4\lambda\eta^2}{N} - \frac{8\gamma^2L^2\eta^2}{\lambda N} - \frac{4\gamma^2B^2}{\lambda N} \\
&\quad - \frac{L(2+4\gamma^2)}{\mu^2} - \frac{8L\gamma^2\eta^2(L+\lambda)^2}{\mu^2} - \frac{L(4+8\gamma^2)\lambda^2\eta^2}{\mu^2}\bigg] \\
&= \frac{1}{2\lambda} - \frac{4L^2}{\lambda\mu^2} - \frac{8L^2\lambda\eta^2}{\mu^2} - \frac{4\lambda\eta^2}{N} - \frac{2L}{\mu^2} - \frac{4L\lambda^2\eta^2}{\mu^2} - \mathcal{O}(\gamma^2) \\
&\ge \frac{1}{2\lambda} - \frac{4L^2}{\lambda\mu^2} - \frac{2L^2}{\lambda\mu^2} - \frac{1}{\lambda N} - \frac{2L}{\mu^2} - \frac{4L}{\mu^2} - \mathcal{O}(\gamma^2) > 0, \qquad \text{(D.23)}
\end{aligned}$$

and

$$\begin{aligned}
&\frac{8L^2\gamma^2}{\lambda\mu^2} + \frac{16\gamma^2L^2\eta^2(L+\lambda)^2}{\lambda\mu^2} + \frac{16\gamma^2L^2\eta^2\lambda^2}{\lambda\mu^2} + \frac{8\gamma^2L^2\eta^2}{\lambda N} + \frac{4\gamma^2B^2}{\lambda N} + \frac{4\gamma^2L}{\mu^2} + \frac{8L\gamma^2\eta^2(L+\lambda)^2}{\mu^2} + \frac{8L\gamma^2\lambda^2\eta^2}{\mu^2} \\
&\le \frac{8L^2\gamma^2}{\lambda\mu^2} + \frac{4\gamma^2L^2(L+\lambda)^2}{\lambda^3\mu^2} + \frac{4\gamma^2L^2}{\lambda\mu^2} + \frac{2\gamma^2L^2}{\lambda^3 n} + \frac{4\gamma^2B^2}{\lambda N} + \frac{4\gamma^2L}{\mu^2} + \frac{2L\gamma^2(L+\lambda)^2}{\lambda^2\mu^2} + \frac{2L\gamma^2}{\mu^2} = \mathcal{O}(\gamma^2), \qquad \text{(D.24)}
\end{aligned}$$

hence we need the condition of

$$\mathcal{O}(\gamma^2) \le \frac{1}{6\lambda} \text{ and } \frac{1}{6\lambda} - \frac{4L^2}{\lambda\mu^2} - \frac{2L^2}{\lambda\mu^2} - \frac{1}{\lambda N} - \frac{2L}{\mu^2} - \frac{4L}{\mu^2} \ge 0 \qquad \text{(D.25)}$$

$$\gamma \le 1/\sqrt{C\lambda}, \qquad \text{(D.26)}$$

where $C$ is a numerical constant irrelevant to $L, B, \gamma, \lambda, \eta$, etc. To obtain the final convergence rate. Reorganize items, we have

$$\min_{t\in[T]}\mathbb{E}[\|\nabla\mathcal{L}(\mathbf{w}_t)\|_2^2] \le \mathcal{O}\bigg(\frac{\mathbb{E}[\mathcal{L}(\mathbf{w}_0)] - \mathbb{E}[\mathcal{L}(\mathbf{w}_T)]}{6\lambda T}\bigg) + \mathcal{O}(\eta^2\sigma_d^2). \qquad \text{(D.27)}$$

This concludes the proof. □