HARNESSING CLIENT DRIFT WITH DECOUPLED GRADIENT DISSIMILARITY

Anonymous authors
Paper under double-blind review

ABSTRACT

The performance of Federated Learning (FL) typically suffers from client drift caused by heterogeneous data, where data distributions vary with clients. Recent studies show that the gradient dissimilarity between clients induced by the data distribution discrepancy causes the client drift. Thus, existing methods mainly focus on correcting the gradients. However, it is challenging to identify which client should (or not) be corrected. This challenge raises a series of questions: will the local training, without gradient correction, contribute to the server model's generalization on other clients' distributions? when does the generalization contribution hold? how to address the challenge when it fails? To answer these questions, we analyze the generalization contribution of local training and conclude that the generalization contribution of local training is bounded by the conditional Wasserstein distance between clients' distributions. Thus, the key to promote generalization contribution is to leverage similar conditional distributions for local training. As collecting data distribution can cause privacy leakage, we propose decoupling the deep models, i.e., splitting the model into a high-level model and a low-level one, for harnessing client drift. High-level models are trained on shared feature distributions, causing promoted generalization contribution and alleviated gradient dissimilarity. Experimental results demonstrate that FL with decoupled gradient dissimilarity is robust to data heterogeneity.

1 INTRODUCTION

To protect data privacy while cooperatively training machine learning models between personal users and organizations, Federated Learning (FL) (Brendan McMahan et al., 2016) is widely exploited as a powerful framework in recent years. In the FL framework, many clients train models without communicating private data. Federated Average (FedAvg) is proposed to make FL practical in low-bandwidth and low-computing resources environments. However, when data distributions between clients are severely heterogeneous (Non-Independent and Identically Distributed, Non-IID), the convergence rate and the generalization performance of FL are much worse than centralized training which collects all the data (Li et al., 2020a; Karimireddy et al., 2019; Kairouz et al., 2019).

The FL community theoretically and empirically found that the "client drift" caused by the heterogeneous data is the main bottleneck of FedAvg (Li et al., 2020a; Karimireddy et al., 2019; Kairouz et al., 2019; Wang et al., 2020a). It means that, after several or more training epochs on private datasets, local models on clients become extremely far away from each other. Recent convergence analysis (Li et al., 2020a; Reddi et al., 2021; Woodworth et al., 2020) of FedAvg shows that the degree of client drift is linearly upper bounded by gradient dissimilarity. Therefore, most existing works (Karimireddy et al., 2019; Wang et al., 2020a) focus on gradient correction techniques to accelerate the convergence rate of local training.

However, how to correct the gradients during the local training is still an open problem (Kairouz et al., 2019; Woodworth et al., 2020; Karimireddy et al., 2019), especially for achieving better generalization ability. The challenge lies in the lack of criterion for identifying which client should (or not) be corrected. This challenge raises a fundamental question in FL systems: *Can the local training on a specific client m contribute to the generalization performance of the server model when evaluted on other clients' distributions?* Moreover, it is also unclear under which conditions the local training can

lead to generalization contribution. The in-depth question is how to deal with the conditions where local training cannot contribute to the server models' generalizability to other clients.

To answer these questions, we formulate the objective of local training in FL systems as a generalization contribution problem. The generalization contribution means how much local training on one client can improve the generalization performance on other clients' distributions for server models. Specifically, we evaluate the generalization performance of a server model locally trained on one client using other clients' data distributions. Our theoretical analysis shows that the generalization contribution of local training is bounded by the conditional Wasserstein distance between clients' distributions. This implies that even if the marginal distributions on different clients are the same, it is insufficient to achieve a guaranteed generalization performance of local training. Therefore, the key to promoting generalization contribution is to leverage the same or similar conditional distributions for local training.

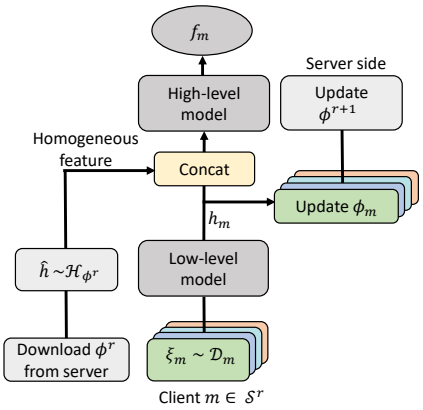


Figure 1: Training process of our framework. The low-level model uses the raw data ξ_m as inputs, and outputs h_m. The high-level model uses h_m and samples $\hat{h}$ from a shared distribution $\mathcal{H}_{\phi^r}$ as inputs for forward and backward propagation.

However, collecting data to construct identical distributions shared across clients is forbidden due to privacy concerns. To avoid privacy leakage, we propose decoupling a deep neural network into a low-level model and a high-level one, i.e., a feature extractor network and a classifier network. Consequently, we can construct a shared identical distribution in the feature space. Namely, on each client, we estimate the feature distribution obtained by the low-level network and send the estimated distribution to the server model. After aggregating the received distributions, the server sends the aggregated distribution and the server model to clients simultaneously. Theoretically, we show that introducing such a simple decoupling strategy promotes the generalization contribution and alleviates gradient dissimilarity. Our extensive experimental results demonstrate the effectiveness of our method, where we consider the global test accuracy of four datasets under various FL settings following previous works (He et al., 2020b; Li et al., 2020a; Wang et al., 2020a).

Our main contributions include: (1) We theoretically show that the generalization contribution from clients during training is bounded by the conditional Wasserstein distance between clients' distributions, answering the question that when the local training on one client can contribute to the generalization performance of server models on other clients' distributions. (2) We are the first to theoretically propose that sharing similar features between clients can improve the generalization contribution from local training, and significantly reduce the gradient dissimilarity. (3) We experimentally validate the gradient dissimilarity reduction and benefits of our method on generalization performance.

2 Related Works

We review FL algorithms aiming to address the Non-IID problem and introduce other works related to measuring client contribution and decoupled training. Due to limited space, we leave a more detailed discussion of the literature review in Appendix C.

2.1 Addressing Non-IID problem in FL

Model Regularization focuses on calibrating the local models to restrict them not to be excessively far away from the server model. A number of works like FedProx (Li et al., 2020a), FedDyn (Acar et al., 2021), SCAFFOLD (Karimireddy et al., 2019) and FedIR (Hsu et al., 2020) add a regularizer of local-global model difference. MOON (Li et al., 2021b) adds the local-global contrastive loss to learn a similar representation between clients.

Reducing Gradient Variance tries to correct the directions of local updates at clients via other gradient information. This kind of method aims to accelerate and stabilize the convergence, like FedNova (Wang et al., 2020a), FedAvgM (Hsu et al., 2019), FedAdaGrad, FedYogi, and FedAdam (Reddi et al., 2021). Our theorem 4.2 provides a new angle to reduce gradient variance.

Personalized Federated Learning aims to make clients optimize different personal models to learn knowledge from other clients and adapt to their own datasets (Tan et al., 2022). The knowledge transfer of personalization is mainly implemented by introducing personalized parameters (Liang et al., 2020; Thapa et al., 2020; Li et al., 2021a), or knowledge distillation on shared local features or extra datasets (He et al., 2020a; Lin et al., 2020; Li & Wang, 2019). Due to the preference for optimizing local objective functions, however, personalized federated models do not have a comparable generic performance (evaluated on global test dataset) to normal FL (Chen & Chao, 2021). Some works (Collins et al., 2021; Arivazhagan et al., 2019) also propose to share feature representations for personalized FL.

2.2 Measuring Contribution from Clients

Clients are only willing to participate in an FL training when given enough rewards. Thus, it is important to measure their contributions to the model performance (Yu et al., 2020; Ng et al., 2020; Liu et al., 2022; Sim et al., 2020). Some works (Yuan et al., 2022) propose to measure the performance gaps from the unseen client distributions experimentally. Data shapley (Ghorbani & Zou, 2019; Sim et al., 2020; Liu et al., 2022) is proposed to measure the generalization performance gain of client participation. Precisely, these works measure the generalization performance gap with or without some clients that never join the whole process of FL. However, we hope to understand the contribution of clients at each communication round. Consequently, our theoretical conclusion guides a modification on data distributions that cannot provide generalization contribution, so that they can improve the generalization performance of the trained model.

2.3 Split Training

Some works propose Split FL (SFL) to utilize split training to accelerate federated learning (Oh et al., 2022; Thapa et al., 2020). In SFL, the model is split into client-side and server-side parts. At each communication round, the client only downloads the client-side model from the server, and conducts forward propagation, and sends the hidden features to the server to compute the loss and conduct backward propagation. These methods aim to accelerate the training speed of FL on the client side and cannot support local updates. In addition, sending all raw features could introduce a high risk of data leakage. Thus, we omit the comparisons to these methods.

2.4 Privacy Concerns

There are many other works (Luo et al., 2021; Chang et al., 2019; Li & Wang, 2019; Bistritz et al., 2020; He et al., 2020a; Liang et al., 2020; Thapa et al., 2020; Oh et al., 2022) that propose to share the hidden features to the server or other clients. Different from them, our decoupling strategy shares the parameters of the estimated feature distributions instead of the raw features, avoiding privacy leakage. We demystify the differences between our method and others in Appendix C.

3 Preliminaries

3.1 Problem Definition

Suppose we have a set of clients $\mathcal{M} = \{1, 2, \cdots, M\}$ with M being the total number of participating clients. FL aims to make these clients with their own distribution $\mathcal{D}_m$ cooperatively learn a machine learning model parameterized as $\theta \in \mathbb{R}^d$. Formally, the global optimization problem of FL can be formulated as (McMahan et al., 2017; Li et al., 2020a):

$$\min_{\theta \in \mathbb{R}^d} F(\theta) := \sum_{m=1}^{M} p_m F_m(\theta) = \sum_{m=1}^{M} p_m \mathbb{E}_{\xi_m \sim \mathcal{D}_m} f(\theta; \xi_m), \tag{1}$$

where $F_m(\theta) = \mathbb{E}_{\xi_m \sim \mathcal{D}_m} f(\theta; \xi_m)$ is the local objective function of client m with $f(\theta; \xi_m) = CE(\rho(\theta; \xi_m), y_m)$, CE denotes the cross-entropy loss, $\rho(\theta; \xi_m)$ is the model output, y_m is the label of ξ_m, $p_m > 0$ and $\sum_{m=1}^{M} p_m = 1$. Usually, p_m is set as $\frac{n_m}{N}$, where n_m denotes the number of samples on client m and $N = \sum_{m=1}^{M} n_m$.

The clients usually have a low communication bandwidth, causing extremely long training time. To address this issue, the classical FL algorithm FedAvg (McMahan et al., 2017) proposes to utilize local updates. Specifically, at each round r, the server sends the global model θ^{r-1} to a subset of clients $\mathcal{S}^r \subseteq \mathcal{M}$ which are randomly chosen. Then, all selected clients conduct some iterations of updates to obtain new client models $\{\theta_m^r\}$, which are sent back to the server. Finally, the server averages local models according to the dataset size of clients to obtain a new global model θ^r.

3.2 GENERALIZATION QUANTIFICATION

Besides defining the metric for the training procedure, we also introduce a metric for the testing phase. Specifically, we define criteria for measuring the generalization performance for a given deep model. Built upon the margin theory (Koltchinskii & Panchenko, 2002), for a given model $\rho(\theta;\cdot)$ parameterized with θ, we use the worst-case margin [1] to measure the generalizability on the distribution $\mathcal{D}$:

Definition 1. (Worst-case margin.) Given a distribution $\mathcal{D}$, the worst-case margin of model $\rho(\theta;\cdot)$ is defined as $W_d(\rho(\theta),\mathcal{D}) = \mathbb{E}_{(\xi,y)\sim\mathcal{D}} \inf_{\rho(\theta;\xi')\neq y} d(\xi',\xi)$ with d being a specific distance.

According to the definition, we can leverage the defined worst-case margin to quantify the generalization performance for a given model ρ and a data distribution $\mathcal{D}$ under a specific distance. Moreover, the defined margin is always not less than zero. It is clear that if the margin is equal to zero, the model mis-classifies almost all samples of the given distribution.

4 DECOUPLED TRAINING AGAINST DATA HETEROGENEITY

This section formulates the generalization contribution in FL systems and decoupling gradient dissimilarity.

4.1 GENERALIZATION CONTRIBUTION

Although Eq. 1 quantifies the performance of model ρ with parameter θ, it focuses more on the training distribution. In FL, we cooperatively train machine learning models because of a belief that introducing more clients *seems* to contribute to the performance of the server models. Given client m, we quantify the "belief", i.e., the generalization contribution, in FL systems as follows:

$$\mathbb{E}_{\Delta:\mathbf{L}(\mathcal{D}_m)} W_d(\rho(\theta+\Delta),\mathcal{D}\backslash\mathcal{D}_m)), \tag{2}$$

where Δ is a pseudo gradient [2] obtained by applying a learning algorithm $\mathbf{L}(\cdot)$ to a distribution $\mathcal{D}_m$, W_d is the quantification of generalization, and $\mathcal{D}\backslash\mathcal{D}_m$ means the distribution of all clients except for client m. Eq. 2 depicts the contribution of client m to generalization ability. Intuitively, we prefer the client where the generalization contribution can be lower bounded.

Definition 2. The Conditional Wasserstein distance $C_d(\mathcal{D},\mathcal{D}')$ between the distribution $\mathcal{D}$ and $\mathcal{D}'$:

$$C_d(\mathcal{D},\mathcal{D}') = \frac{1}{2}\mathbb{E}_{(\cdot,y)\sim\mathcal{D}} \inf_{J\in\mathcal{J}(\mathcal{D}|y,\mathcal{D}'|y)} \mathbb{E}_{(\xi,\xi')\sim J} d(\xi,\xi') + \frac{1}{2}\mathbb{E}_{(\cdot,y)\sim\mathcal{D}'} \inf_{J\in\mathcal{J}(\mathcal{D}|y,\mathcal{D}'|y)} \mathbb{E}_{(\xi,\xi')\sim J} d(\xi,\xi'). \tag{3}$$

Built upon Definition 1, 2, and Eq. 2, we are ready to state the following theorem (proof in Appendix B.1).

Theorem 4.1. *With the pseudo gradient Δ obtained by $\mathbf{L}(\mathcal{D}_m)$, the generalization contribution is lower bounded:*

$$\begin{aligned}\mathbb{E}_{\Delta:\mathbf{L}(\mathcal{D}_m)} W_d(\rho,\mathcal{D}\backslash\mathcal{D}_m)) \geq\ & \mathbb{E}_{\Delta:\mathbf{L}(\mathcal{D}_m)} W_d(\rho,\tilde{\mathcal{D}}_m) - |\mathbb{E}_{\Delta:\mathbf{L}(\mathcal{D}_m)} W_d(\rho,\mathcal{D}_m)) - W_d(\rho,\tilde{\mathcal{D}}_m))| \\ & - 2C_d(\mathcal{D}_m,\mathcal{D}\backslash\mathcal{D}_m),\end{aligned}$$

where $\tilde{\mathcal{D}}_m$ represents the dataset sampled from $\mathcal{D}_m$.

Remark 1. Theorem 4.1 implies that: a) the generalization contribution of the training distribution is expected to be large on the training set; b) the generalization contribution on the training distribution should be similar to that on the training dataset; c) promoting the generalization performance requires constructing similar conditional distributions of clients, i.e., the distribution distance $C_d(\mathcal{D}_m,\mathcal{D}\backslash\mathcal{D}_m)$ is expected to be small. Both the Definition 2 and Theorem 4.1 use distributions conditioned on the label y, so we write the feature distribution $\mathcal{H}|y$ as $\mathcal{H}$ for brevity in rest of the paper.

Built upon the theoretical analysis, it is straightforward to make all client models trained on similar distributions to obtain higher generalization performance. However, collecting data to construct such a distribution is forbidden in FL due to privacy concerns. To address this challenge, we propose decoupling a deep neural network into a feature extractor network $\varphi_{\theta_{low}}$ parameterized by $\theta_{low}\in\mathbb{R}^{d_l}$ and a classifier network parameterized by $\theta_{high}\in\mathbb{R}^{d_h}$, and making the classifier network trained on

[1] The same definition is used in the literature (Franceschi et al., 2018).

[2] The pseudo gradient at round r is calculated as: $\Delta^r = \theta_T^{r-1} - \theta_0^{r-1}$ with the maximum local iterations T.

the similar conditional distributions, as shown in Figure 1. Here, d_l and d_h represent the dimensions of parameters θ_{low} and θ_{high}, respectively.

Specifically, client m can calculate the hidden feature distribution using $\varphi_{\theta_{low}}(\xi_m), \xi_m \sim \mathcal{D}_m$ and send it to the server for the distribution approximation. Then, the server aggregates the received distributions to obtain the global feature distribution $\mathcal{H}$ and broadcasts it, being similar to the widely used FedAvg strategy. Finally, the classifier network thus can receive a distribution shared across clients and perform local training on both the local and the received feature distribution during the local training. In what follows, we show that such a decoupling strategy can reduce the gradient dissimilarity, besides the promoted generalization performance.

4.2 Decoupled Gradient Dissimilarity

The gradient dissimilarity in FL resulted from heterogeneous data, i.e., the data distribution on client m, $\mathcal{D}_m$, is different from that on client k, $\mathcal{D}_k$ (Karimireddy et al., 2019; Li et al., 2020a). The commonly used quantitative measure of gradient dissimilarity is defined as inter-client gradient variance (CGV).

Definition 3. Inter-client Gradient Variance (CGV): (Kairouz et al., 2019; Karimireddy et al., 2019; Woodworth et al., 2020; Koloskova et al., 2020) $\text{CGV}(F, \theta) = \mathbb{E}_{\xi_m}||\nabla f_m(\theta; \xi_m) - \nabla F(\theta)||^2$. CGV is usually assumed to be upper bounded (Kairouz et al., 2019; Woodworth et al., 2020; Lian et al., 2017), i.e., $\text{CGV}(F, \theta) = \mathbb{E}_{\xi_m}||\nabla f_m(\theta; \xi_m) - \nabla F(\theta)||^2 \leq \sigma^2$ with a constant σ.

Lower bounded gradient dissimilarity benefits the theoretical convergence rate (Woodworth et al., 2020). Specifically, lower gradient dissimilarity directly causes higher convergence rate (Karimireddy et al., 2019; Li et al., 2020a; Woodworth et al., 2020). This means that the decoupling strategy can also benefit the convergence rate if the gradient dissimilarity can be reduced. Now, we are ready to demonstrate how to reduce the gradient dissimilarity CGV with our decoupling strategy. With representing $\nabla f_m(\theta; \xi_m)$ as $\left\{\nabla_{\theta_{low}} f_m(\theta; \xi_m), \nabla_{\theta_{high}} f_m(\theta; \xi_m)\right\}$, we propose that the CGV can be divided into two terms of the different parts of θ (see Appendix B.2 for details):

$$\begin{aligned}\text{CGV}(F, \theta) &= \mathbb{E}_{\xi_m}||\nabla f_m(\theta; \xi_m) - \nabla F(\theta)||^2 \\ &= \mathbb{E}_{\xi_m}\left[||\nabla_{\theta_{low}} f_m(\theta; \xi_m) - \nabla_{\theta_{low}} F(\theta)||^2 + ||\nabla_{\theta_{high}} f_m(\theta; \xi_m) - \nabla_{\theta_{high}} F(\theta)||^2\right].\end{aligned} \tag{4}$$

According to the chain rule of the gradients of a deep model, we can derive that the high-level part of gradients that are calculated with the raw data ξ_m is equal to gradients with the hidden features $h_m = \varphi_{\theta_{low}}(\xi_m)$ (proof in Appendix B.2):

$$\nabla_{\theta_{high}} f_m(\theta; \xi_m) = \nabla_{\theta_{high}} f_m(\theta; h_m), \nabla_{\theta_{high}} F(\theta) = \sum_{m=1}^{M} p_m \mathbb{E}_{\xi_m \sim \mathcal{D}_m} \nabla_{\theta_{high}} f(\theta; h_m), \tag{5}$$

in which $f_m(\theta; h_m)$ is computed by forwarding the h_m through the high-level model without the low-level part.

We propose to let all clients share a global feature distribution $\mathcal{H}$ which approximates all features of clients. Client m will sample $\hat{h} \sim \mathcal{H}$ and $h_m = \varphi_{\theta_{low}}(\xi_m)|_{\xi_m \sim \mathcal{D}_m}$ to train their classifier network, then the objective function becomes as [3]:

$$\min_{\theta \in \mathbb{R}^d} \hat{F}(\theta) := \sum_{m=1}^{M} \hat{p}_m \mathbb{E}_{\substack{\xi_m \sim \mathcal{D}_m \\ \hat{h} \sim \mathcal{H}}} \hat{f}(\theta; \xi_m, \hat{h}) = \sum_{m=1}^{M} \hat{p}_m \mathbb{E}_{\substack{\xi_m \sim \mathcal{D}_m \\ \hat{h} \sim \mathcal{H}}} \left[f(\theta; \varphi_{\theta_{low}}(\xi_m)) + f(\theta; \hat{h})\right]. \tag{6}$$

Here, $\hat{p}_m = \frac{n_m + \hat{n}_m}{N + \hat{N}}$ with n_m and $\hat{n}_m$ being the sampling size of ξ_m and $\hat{h}$ respectively, and $\hat{N} = \sum_{m=1}^{M} \hat{n}_m$. Now, we are ready to state the following theorem of reducing gradient dissimilarity by sampling features from the same distribution (proof in Appendix B.3).

Theorem 4.2. *Under the gradient variance measure CGV (Definition 3), with $\hat{n}_m$ satisfying $\frac{\hat{n}_m}{n_m + \hat{n}_m} = \frac{\hat{N}}{N + \hat{N}}$, the objective function $\hat{F}(\theta)$ causes a tighter bounded gradient dissimilarity, i.e., the CGV$(\hat{F}, \theta) = \mathbb{E}_{\xi_m}||\nabla_{\theta_{low}} f_m(\theta; \xi_m) - \nabla_{\theta_{low}} F(\theta)||^2 + \frac{N^2}{(N+\hat{N})^2}||\nabla_{\theta_{high}} f_m(\theta; \xi_m) - \nabla_{\theta_{high}} F(\theta)||^2 \leq$ CGV(F, θ).*

[3] We reuse f here for brevity, the input of f can be the input data ξ_m or the hidden feature $\varphi_{\theta_{low}}(\xi_m)$.

Remark 2. Theorem 4.2 shows that the high-level gradient dissimilarity can be reduced as $\frac{N^2}{(N+\hat{N})^2}$ times by sampling the same features between clients. Hence, estimating and sharing feature distributions is the key to promoting the generalization contribution and the reduction of gradient dissimilarity. Note that choosing $\hat{N} = \infty$ can eliminate high-level dissimilarity. However, two reasons make it impractical to sample infinite features $\hat{h}$. First, the distribution is estimated using limited samples, leading to biased estimations. Second, infinite sampling will dramatically increase the calculating cost. We set $\hat{N} = N$ in our experiments.

4.3 Training Procedure

The training procedure of the proposed decoupling strategy is simple to implement. Specifically, it merely requires two extra steps compared with the vanilla FedAvg method: a) estimating and broadcasting a global distribution $\mathcal{H}$; b) performing local training with both the local data and the $\hat{h} \sim \mathcal{H}$.

Moreover, sampling $\hat{h} \sim \mathcal{H}$ has two additional advantages as follows. First, directly sharing the raw hidden features may incur privacy concerns. The raw data may be reconstructed by feature inversion methods (Zhao et al., 2021). The $\hat{h} \sim \mathcal{H}$ is the approximation of the real h_m. One can use different distribution approximation methods to estimate $\{h_m | m \in \mathcal{M}\}$ to avoid exposing the raw data. Second, the hidden features usually have much higher dimensions than the raw data (Lin et al., 2021). Hence, communicating and saving them between clients and servers may not be practical. We can use different distribution approximation methods to obtain $\mathcal{H}$ with different sizes of parameters ϕ. Transmitting ϕ can consume less communication resource than a set of distributions $\{h_m | m \in \mathcal{M}\}$.

Following previous work (Kendall & Gal, 2017), we simply assume a Gaussian distribution to approximate the feature distributions. Namely, on the client-side, we use a Gaussian Distribution $\mathcal{N}(\sigma_m, \mu_m)$ parameterized with σ_m and μ_m to approximate the feature distribution $\varphi_{\theta_{low}}(\xi_m), \xi_m \sim \mathcal{D}_m$ on client m. As shown in Figure 1 and Algorithm 1, during the local training, clients update σ_m and μ_m using the real feature h_m following a moving average strategy which is widely used in the literature (Ioffe & Szegedy, 2015; Wang et al., 2021):

Algorithm 1 Framework of our method.

server input: initial θ^0, maximum communication round R
client m**'s input:** local iterations T

Initialization: server distributes the initial model θ^0 to all clients, and the initial global parameter ϕ^0.

Server_Executes:
for each round $r = 0, 1, \cdots, R$ **do**
  server samples a set of clients $\mathcal{S}_r \subseteq \{1, ..., M\}$.
  server **communicates** θ_r and ϕ^r to all clients $m \in \mathcal{S}$.
  for each client $m \in \mathcal{S}^r$ **in parallel do do**
    $\theta^{r+1}_{m,E-1}$, $\phi^{r+1}_m \leftarrow$ ClientUpdate(m, θ^r, ϕ^r).
  end for
  $\theta^{r+1} \leftarrow \sum_{m=1}^{M} p_m \theta^{r+1}_{m,E-1}$.
  Update ϕ^{r+1} using ϕ^{r+1}_m.
end for

ClientUpdate(m, θ, ϕ)**:**
for each local iteration t with $t = 0, \cdots, T-1$ **do**
  Sample raw data $\xi_m \sim \mathcal{D}_m$ and $\hat{h} \sim \mathcal{H}_\phi(\hat{h}|y)$.
  $\theta_{m,t+1} \leftarrow \theta_{m,t} - \eta_{m,t} \nabla_\theta \hat{f}(\theta; \xi_m, \hat{h})$, i.e., Eq. 6
  Update ϕ_m using $\hat{h}_m = \varphi_{\theta_{low}}(\xi_m)$.
end for
Return θ and ϕ_m to server.

$$\begin{aligned} \sigma_m^{(t+1)} &= \beta_m \sigma_m^{(t)} + (1 - \beta_m) \times \text{mean}(h_m), \\ \mu_m^{(t+1)} &= \beta_m \mu_m^{(t)} + (1 - \beta_m) \times \text{varaince}(h_m), \end{aligned} \tag{7}$$

where t is the iteration of the local training, β_m is the momentum coefficient. On the server side, σ_g and μ_g are aggregated as:

$$\sigma_g = \frac{1}{|\mathcal{S}^r|} \sum_{i \in \mathcal{S}^r} \sigma_i^T, \mu_g = \frac{1}{|\mathcal{S}^r|} \sum_{i \in \mathcal{S}^r} \mu_i^T, \tag{8}$$

where T stands for the maximum iteration of local training.

We perform the second step of the proposed decoupling strategy by optimizing the designed objective function, i.e., Eq. 6. Built upon the above analysis, the decoupling strategy can benefit both the performance contribution, i.e., conclusion of Theorem 4.1, and the convergence rate, i.e., Theorem 4.2.

Table 1: Best test accuracy (%) of all experimental results.

Dataset	Cent. Acc.	FL Setting			FL Test Accuracy				
		a	E	M	FedAvg	FedProx	SCAFFOLD	FedNova	Ours
CIFAR-10	92.53	0.1	1	10	83.65	83.22	82.33	84.97	**88.45**
		0.05	1	10	75.36	77.49	33.6	73.49	**81.75**
		0.1	5	10	85.69	85.33	84.4	86.92	**88.10**
		0.1	1	100	73.42	68.59	59.22	74.94	**77.56**
		Average			79.53	78.66	64.89	80.08	**83.97**
FMNIST	93.7	0.1	1	10	88.67	88.92	87.81	87.97	**90.83**
		0.05	1	10	82.73	83.66	76.16	81.89	**86.42**
		0.1	5	10	87.6	88.41	88.44	87.66	**89.87**
		0.1	1	100	90.12	90.39	88.24	90.40	**90.98**
		Average			87.28	87.85	85.16	86.98	**89.53**
SVHN	95.27	0.1	1	10	88.20	87.04	83.87	88.48	**92.37**
		0.05	1	10	80.67	82.39	82.29	84.01	**90.25**
		0.1	5	10	86.32	86.05	83.14	88.10	**91.58**
		0.1	1	100	92.42	92.29	92.06	92.44	**93.42**
		Average			86.90	86.94	85.34	88.26	**91.91**
CIFAR-100	74.25	0.1	1	10	69.38	69.78	65.74	69.52	**70.28**
		0.05	1	10	63.80	64.75	61.49	64.57	**66.60**
		0.1	5	10	68.39	68.71	68.67	67.99	**68.79**
		0.1	1	100	53.22	54.10	23.77	55.40	**56.07**
		Average			63.70	64.34	54.92	64.37	**65.44**

"Cent." means centralized training. "Acc." means the test accuracy.

5 EXPERIMENTS

5.1 EXPERIMENT SETUP

Federated Datasets and Models. We verify our method with four datasets commonly used in the FL community, i.e., CIFAR-10 (Krizhevsky & Hinton, 2009), FMNIST (Xiao et al., 2017), SVHN (Netzer et al., 2011) and CIFAR-100 (Krizhevsky & Hinton, 2009). We use the Latent Dirichlet Sampling (LDA) partition method to simulate the Non-IID data distribution, which is the most used partition method in FL (He et al., 2020b; Li et al., 2021b; Luo et al., 2021). We conduct experiments with two different Non-IID degrees, $a = 0.1$ and $a = 0.05$.

Baselines and Metrics. We choose the classical FL algorithm, FedAvg (McMahan et al., 2017), and recent effective FL algorithms proposed to address the client drift problem, including FedProx (Li et al., 2020a), SCAFFOLD (Karimireddy et al., 2019), and FedNova (Wang et al., 2020a), as our baselines. The detailed hyper-parameters of all experiments are reported in Appendix D. We use two metrics, the best accuracy and the number of communication rounds to achieve a target accuracy, which is set to the best accuracy of FedAvg. We also measure the client drift (Karimireddy et al., 2019), $\frac{1}{|\mathcal{S}^r|}\sum_{i\in\mathcal{S}^r}\|\bar{\theta}-\theta_i\|$, as it reflects the effect on gradient dissimilarity reduction.

5.2 EXPERIMENTAL RESULTS

Basic FL setting. As shown in Table 1, using the classical FL training setting, i.e. $a = 0.1$, $E = 5$ and $M = 10$, for CIFAR-10, FMNIST and SVHN, our method achieves much higher generalization performance than other methods. We also find that, for CIFAR-100, the performance of our method is similar to FedProx. We conjecture that CIFAR-100 dataset has more classes than other datasets, leading to the results. Thus, a powerful feature estimation approach instead of a simple Gaussian assumption can be a promising direction to enhance the performance.

Impacts of Non-IID Degree. As shown in Table 1, for all datasets with high Non-IID degree ($a = 0.05$), our methods obtain more performance gains than the case of lower Non-IID degree ($a = 0.1$). For example, we obtain 92.37% test accuracy on SVHN with $a = 0.1$, higher than the FedNova by 3.89%. Furthermore, when Non-IID degree increases to $a = 0.05$, we obtain 90.25% test accuracy, higher than FedNova by 6.14%. And for CIFAR-100, our method shows benefits when $a = 0.05$, demonstrating that our method can defend against more severe data heterogeneity.

Different Number of Clients. We also show the results of 100-client FL setting in Table 1. Our method works well with all datasets, demonstrating excellent scalability with more clients.

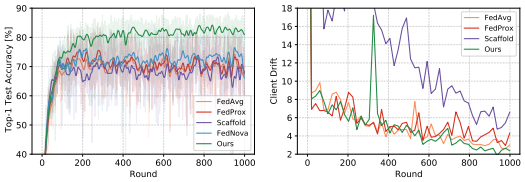
Table 2: Communication Round to attain the target accuracy.

Dataset	FL Setting			Target	Communication Round to attain the target accuracy				
	a	E	M	Acc.	FedAvg	FedProx	SCAFFOLD	FedNova	Ours
CIFAR-10	0.1	1	10	82.0	142	**128**	863	142	**128**
	0.05	1	10	73.0	247	121	NaN	407	**112**
	0.1	5	10	84.0	128	128	360	80	**78**
	0.1	1	100	73.0	957	NaN	NaN	992	**706**
FMNIST	0.1	1	10	87.0	83	76	275	83	**32**
	0.05	1	10	81.0	94	94	NaN	395	**52**
	0.1	5	10	87.0	147	31	163	88	**17**
	0.1	1	100	90.0	375	470	NaN	**317**	441
SVHN	0.1	1	10	87.0	292	247	NaN	251	**50**
	0.05	1	10	80.0	578	68	358	242	**50**
	0.1	5	10	86.0	251	350	NaN	NaN	**11**
	0.1	1	100	92.0	471	356	669	356	**346**
CIFAR-100	0.1	1	10	69.0	712	857	NaN	733	**614**
	0.05	1	10	61.0	386	386	755	366	**313**
	0.1	5	10	68.0	335	307	**182**	282	300
	0.1	1	100	53.0	992	939	NaN	910	**854**

"NaN." means that this algorithm does not achieve the target.

Different Local Epochs. More local training epochs E could reduce the communication rounds, saving communication cost in practical scenarios. In Table 1, the results of $E = 5$ on CIFAR-10, FMNIST, and SVHN verify that our method works well when increasing local training time.

Client Drift. We show the client drift of different methods in Figure 2 (b), where FedNova is excluded due to its significant client drift. At the initial training stage, the client drift is similar for different methods. During this stage, the low-level model is still unstable and the feature estimation is not accurate. After about 500 communication rounds, our method begins to show lower client drift than others.



(a) Test Accuracy (b) Client Drift

Figure 2: CIFAR10 with $a = 0.1$, $E = 1$, $M = 10$.

Convergence Speed. Figure 2 (a) shows that our method can accelerate the convergence of FL.[4] And we compare the communication rounds that different algorithms need to attain the target accuracy in Table 2. The results show that our method can improve the convergence speed. The possible reasons for failure cases may be due to the too many categories in the dataset.

5.3 Ablation Study

To verify the impacts of the depth of gradient decoupling, we conduct experiments by splitting at different layers, including the 5-th, 9-th, 13-th and 17-th layers.

Table 3: Splitting at different layers.

Layer	5-th	9-th	13-th	17-th
Test Acc. (%)	87.64	88.45	87.86	84.08

Results are shown in Table 3, demonstrating that our method can obtain benefits at low or middle layers. Decoupling at the 17-th layer will decrease the performance, which is consistent with our conclusion in Sec. 4.2. Specifically, decoupling at a very high layer may not be enough to resist gradient dissimilarity, leading to weak data heterogeneity mitigation. Interestingly, according to Theorem 4.2, decoupling at the 5-th layer should diminish more gradient dissimilarity than the 9-th and 13-th layers; but it does not show performance gains. We conjecture that it is due to the difficulty of distribution estimation, since

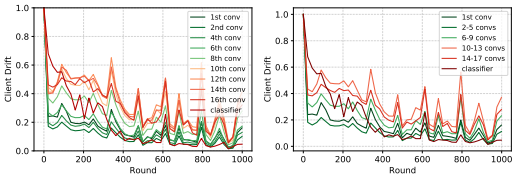
[4]Due to the high instability of training with severe data heterogeneity, we show the actual test accuracy as semitransparent lines and the smoothed test accuracy as opaque lines for better visualization. We also provide more convergence figures in Appendix D.

biased estimation leads to poor generalization contribution. As other works (Lin et al., 2021) indicate, features at the lower level usually are richer larger than at the higher level. Thus, estimating the lower-level features is much more difficult than the higher-level.

5.4 Discussion

In this section, we provide some more experimental support for our method. All experiment results of this section are conducted on CIFAR-10, $a = 0.1$, $E = 1$ and $M = 10$.

Our method only guarantees the reduction of high-level gradient dissimilarity without considering the low-level part. We experimentally find that low-level client drift shrinks faster than high-level. Here, we show the layer-wise divergence (Karimireddy et al., 2019; Li et al., 2020a) $\frac{1}{|\mathcal{S}^r|}\sum_{i\in\mathcal{S}^r}\|\bar{\theta}_l - \theta_{i,l}\|$ of ResNet-18 trained by FedAvg in Figure 3. We choose and show the divergence of 10 layers of ResNet-18 in Figure 3 (a), and the different stages of ResNet-18 in Figure 3 (b). As we hope to demonstrate the divergence trend, we normalize each line with its maximum value. The results show that the low-level divergence shrinks faster than the high-level divergence. This means that reducing the high-level gradient dissimilarity is more important than the low-level.



(a) Some Choosen layers. (b) Different Parts.

Figure 3: Layer divergence of FedAvg.

We conduct FedAvg with many communication rounds with and without learning rate decay. We show the results of the first 5000 rounds in Figure 4 (We offer the results of longer training time in Appendix D). FedAvg without learning rate decay can only achieve 86.96% accuracy, and FedAvg with learning rate decay only achieves 82.65% accuracy. The results show that the longer training time cannot fill the generalization performance gap between FedAvg and centralized training, encouraging us to develop new optimization schemes to improve the performance of FL.

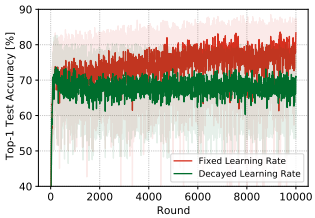


Figure 4: Long Training Time.

6 Limitations

Estimation of Feature Distribution. In this work, we only use the Gaussian Distribution to estimate the feature distribution. This significantly limits the performance of this framework, while it can work well in our experiments. Future works may exploit better feature estimators like generative models (Goodfellow et al., 2014; Karras et al., 2019) to sample higher-quality features.

Extra Communication and Calculation Cost. Our method only needs to communicate the parameters of the estimated feature distribution, which is much less than all features of clients. However, in some low-bandwidth scenarios like IoT devices, people may not want to increase this communication overhead. Furthermore, our method doubles the calculation costs of the forward and backward process of the high-level model. Thus, more reducing gradient dissimilarity, more calculation costs. This plays as a trade-off and needs to be further studied in the future.

7 Conclusion

In this paper, we raise a series of fundamental questions related to measuring the generalization contribution of local training from the clients. Then, we theoretically show the relationship of this generalization contribution with the conditional Wasserstein distance between clients' distributions. The theoretical conclusion inspires us to propose decoupling gradient dissimilarity, which greatly reduces the gradient dissimilarity by training with a shared feature distribution without privacy concerns. We theoretically verify the gradient dissimilarity reduction and experimentally validate our methods' benefits on generalization performance. Our work opens a new view of promoting FL performance from a generalization perspective.

ETHIC STATEMENT

This paper does not raise any ethical concerns. This study does not involve any human subjects, practices to data set releases, potentially harmful insights, methodologies and applications, potential conflicts of interest and sponsorship, discrimination/bias/fairness concerns, privacy and security issues, legal compliance, and research integrity issues.

REPRODUCIBILITY STATEMENT

To ensure the reproducibility, we have listed all hyper-parameters, hardware and software of all experiments in the Appendix D. Due to the privacy concerns, we will upload the anonymous link of codes and instructions during the rebuttal to make it only visible to reviewers. All explanations of assumptions can be found in Section 3 and 4, and the complete proof of Theorem 4.1 and 4.2 can be found in Appendix B.

REFERENCES

Durmus Alp Emre Acar, Yue Zhao, Ramon Matas, Matthew Mattina, Paul Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=B7v4QMR6Z9w.

Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers. *CoRR*, abs/1912.00818, 2019. URL http://arxiv.org/abs/1912.00818.

Ilai Bistritz, Ariana Mann, and Nicholas Bambos. Distributed distillation for on-device learning. *Advances in Neural Information Processing Systems*, 33:22593–22604, 2020.

H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. *arXiv e-prints*, art. arXiv:1602.05629, February 2016.

Kuntai Cai, Xiaoyu Lei, Jianxin Wei, and Xiaokui Xiao. Data synthesis via differentially private markov random fields. *Proc. VLDB Endow.*, 14(11):2190–2202, jul 2021. ISSN 2150-8097. doi: 10.14778/3476249.3476272. URL https://doi.org/10.14778/3476249.3476272.

Hongyan Chang, Virat Shejwalkar, Reza Shokri, and Amir Houmansadr. Cronus: Robust and heterogeneous collaborative learning with black-box knowledge transfer. *arXiv preprint arXiv:1912.11279*, 2019.

Zachary Charles, Zachary Garrett, Zhouyuan Huo, Sergei Shmulyian, and Virginia Smith. On large-cohort training for federated learning. *Advances in Neural Information Processing Systems*, 34, 2021.

A Chatalic, V Schellekens, F Houssiau, Y A de Montjoye, L Jacques, and R Gribonval. Compressive learning with privacy guarantees. *Information and Inference: A Journal of the IMA*, 05 2021. ISSN 2049-8772. doi: 10.1093/imaiai/iaab005. URL https://doi.org/10.1093/imaiai/iaab005. iaab005.

Hong-You Chen and Wei-Lun Chao. On bridging generic and personalized federated learning for image classification. In *International Conference on Learning Representations*, 2021.

Yae Jee Cho, Jianyu Wang, and Gauri Joshi. Client selection in federated learning: Convergence analysis and power-of-choice selection strategies. *arXiv preprint arXiv:2010.01243*, 2020.

Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 2089–2099. PMLR, 18–24 Jul 2021.

Jean-Yves Franceschi, Alhussein Fawzi, and Omar Fawzi. Robustness of classifiers to uniform ℓ_p and gaussian noise. In *International Conference on Artificial Intelligence and Statistics*, pp. 1280–1288. PMLR, 2018.

Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. In *International Conference on Machine Learning*, pp. 2242–2251. PMLR, 2019.

Jack Goetz and Ambuj Tewari. Federated learning via synthetic data. *arXiv preprint arXiv:2008.04489*, 2020.

Jack Goetz, Kshitiz Malik, Duc Bui, Seungwhan Moon, Honglei Liu, and Anuj Kumar. Active federated learning. *arXiv preprint arXiv:1909.12641*, 2019.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pp. 2672–2680, Cambridge, MA, USA, 2014. MIT Press.

Weituo Hao, Mostafa El-Khamy, Jungwon Lee, Jianyi Zhang, Kevin J Liang, Changyou Chen, and Lawrence Carin Duke. Towards fair federated learning with zero-shot data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3310–3319, 2021.

Moritz Hardt and Guy N. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pp. 61–70, 2010. doi: 10.1109/FOCS.2010.85.

Moritz Hardt, Katrina Ligett, and Frank Mcsherry. A simple and practical algorithm for differentially private data release. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL `https://proceedings.neurips.cc/paper/2012/file/208e43f0e45c4c78cafadb83d2888cb6-Paper.pdf`.

Chaoyang He, Murali Annavaram, and Salman Avestimehr. Group knowledge transfer: Federated learning of large cnns at the edge. In *Advances in Neural Information Processing Systems 34*, 2020a.

Chaoyang He, Songze Li, Jinhyun So, Mi Zhang, Hongyi Wang, Xiaoyang Wang, Praneeth Vepakomma, Abhishek Singh, Hang Qiu, Li Shen, Peilin Zhao, Yan Kang, Yang Liu, Ramesh Raskar, Qiang Yang, Murali Annavaram, and Salman Avestimehr. Fedml: A research library and benchmark for federated machine learning. *arXiv preprint arXiv:2007.13518*, 2020b.

T. Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *ArXiv*, abs/1909.06335, 2019.

Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Federated visual classification with real-world data distribution. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part X 16*, pp. 76–92. Springer, 2020.

Zhouyuan Huo, Bin Gu, and Heng Huang. Training neural networks using features replay. *Advances in Neural Information Processing Systems*, 31, 2018.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. PMLR, 2015.

Max Jaderberg, Wojciech Marian Czarnecki, Simon Osindero, Oriol Vinyals, Alex Graves, David Silver, and Koray Kavukcuoglu. Decoupled neural interfaces using synthetic gradients. In *International conference on machine learning*, pp. 1627–1635. PMLR, 2017.

Eunjeong Jeong, Seungeun Oh, Hyesung Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. *NeurIPS*, 2018.

Noah Johnson, Joseph P Near, and Dawn Song. Towards practical differential privacy for sql queries. *Proceedings of the VLDB Endowment*, 11(5):526–539, 2018.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J Reddi, Sebastian U Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. *arXiv preprint arXiv:1910.06378*, 2019.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4401–4410, 2019.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017.

Anastasia Koloskova, Nicolas Loizou, Sadra Boreiri, Martin Jaggi, and Sebastian Stich. A unified theory of decentralized SGD with changing topology and local updates. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5381–5393. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/koloskova20a.html.

Vladimir Koltchinskii and Dmitry Panchenko. Empirical margin distributions and bounding the generalization error of combined classifiers. *The Annals of Statistics*, 30(1):1–50, 2002.

A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

Fan Lai, Xiangfeng Zhu, Harsha V. Madhyastha, and Mosharaf Chowdhury. Oort: Efficient federated learning via guided participant selection. In *15th USENIX Symposium on Operating Systems Design and Implementation (OSDI 21)*, pp. 19–35. USENIX Association, July 2021. ISBN 978-1-939133-22-9. URL https://www.usenix.org/conference/osdi21/presentation/lai.

Ang Li, Jingwei Sun, Binghui Wang, Lin Duan, Sicheng Li, Yiran Chen, and Hai Li. Lotteryfl: Empower edge intelligence with personalized and communication-efficient federated learning. In *2021 IEEE/ACM Symposium on Edge Computing (SEC)*, pp. 68–79, 2021a. doi: 10.1145/3453142.3492909.

Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*, 2019.

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, 2021b.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems*, volume 2, pp. 429–450, 2020a. URL https://proceedings.mlsys.org/paper/2020/file/38af86134b65d0f10fe33d30dd76442e-Paper.pdf.

Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. In *International Conference on Learning Representations*, 2020b. URL https://openreview.net/forum?id=HJxNAnVtDS.

Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pp. 5330–5340, 2017.

Paul Pu Liang, Terrance Liu, Liu Ziyin, Nicholas B Allen, Randy P Auerbach, David Brent, Ruslan Salakhutdinov, and Louis-Philippe Morency. Think locally, act globally: Federated learning with local and global representations. *arXiv preprint arXiv:2001.01523*, 2020.

Ji Lin, Wei-Ming Chen, Han Cai, Chuang Gan, and song han. Memory-efficient patch-based inference for tiny deep learning. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=C1mPUP7uKNp.

Tao Lin, Lingjing Kong, Sebastian U. Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. In *NeurIPS*, 2020.

Zelei Liu, Yuanyuan Chen, Han Yu, Yang Liu, and Lizhen Cui. Gtg-shapley: Efficient and accurate participant contribution evaluation in federated learning. *ACM Trans. Intell. Syst. Technol.*, 13(4), may 2022. ISSN 2157-6904.

Yunhui Long, Boxin Wang, Zhuolin Yang, Bhavya Kailkhura, Aston Zhang, Carl Gunter, and Bo Li. G-pate: Scalable differentially private data generator via private aggregation of teacher discriminators. *Advances in Neural Information Processing Systems*, 34, 2021.

Sindy Löwe, Peter O'Connor, and Bastiaan Veeling. Putting an end to end-to-end: Gradient-isolated learning of representations. *Advances in neural information processing systems*, 32, 2019.

Mi Luo, Fei Chen, Dapeng Hu, Yifan Zhang, Jian Liang, and Jiashi Feng. No fear of heterogeneity: Classifier calibration for federated learning with non-IID data. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=AFiH_CNnVhS.

Enrique S Marquez, Jonathon S Hare, and Mahesan Niranjan. Deep cascade learning. *IEEE transactions on neural networks and learning systems*, 29(11):5475–5485, 2018.

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, 2017.

Omar Montasser, Steve Hanneke, and Nathan Srebro. Vc classes are adversarially robustly learnable, but only improperly. In *Conference on Learning Theory*, pp. 2512–2530. PMLR, 2019.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. URL http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf.

Kang Loon Ng, Zichen Chen, Zelei Liu, Han Yu, Yang Liu, and Qiang Yang. A multi-player game for studying federated learning incentive schemes. In *IJCAI International Joint Conference on Artificial Intelligence*, pp. 5279, 2020.

Arild Nøkland and Lars Hiller Eidnes. Training neural networks with local error signals. In *International conference on machine learning*, pp. 4839–4850. PMLR, 2019.

Seungeun Oh, Jihong Park, Praneeth Vepakomma, Sihun Baek, Ramesh Raskar, Mehdi Bennis, and Seong-Lyun Kim. Locfedmix-sl: Localize, federate, and mix for improved scalability, convergence, and latency in split learning. In *Proceedings of the ACM Web Conference 2022*, pp. 3347–3357, 2022.

Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and Hugh Brendan McMahan. Adaptive federated optimization. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=LkFG3lB13U5.

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

MyungJae Shin, Chihoon Hwang, Joongheon Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. Xor mixup: Privacy-preserving data augmentation for one-shot federated learning. *arXiv preprint arXiv:2006.05148*, 2020.

Rachael Hwee Ling Sim, Yehong Zhang, Mun Choon Chan, and Bryan Kian Hsiang Low. Collaborative machine learning with incentive-aware model rewards. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. Towards personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–17, 2022. doi: 10.1109/TNNLS.2022.3160699.

Chandra Thapa, Mahawaga Arachchige Pathum Chamikara, Seyit Camtepe, and Lichao Sun. Splitfed: When federated learning meets split learning. *arXiv preprint arXiv:2004.12088*, 2020.

Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H. Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. In *Advances in Neural Information Processing Systems*, volume 33, pp. 7611–7623, 2020a. URL https://proceedings.neurips.cc/paper/2020/file/564127c03caab942e503ee6f810f54fd-Paper.pdf.

Yulin Wang, Zanlin Ni, Shiji Song, Le Yang, and Gao Huang. Revisiting locally supervised learning: an alternative to end-to-end training. In *International Conference on Learning Representations*, 2020b.

Yulin Wang, Gao Huang, Shiji Song, Xuran Pan, Yitong Xia, and Cheng Wu. Regularizing deep networks with semantic data augmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

Blake E Woodworth, Kumar Kshitij Patel, and Nati Srebro. Minibatch vs local sgd for heterogeneous distributed learning. *Advances in Neural Information Processing Systems*, 33:6281–6292, 2020.

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

Haibo Yang, Minghong Fang, and Jia Liu. Achieving linear speedup with partial worker participation in non-iid federated learning. In *International Conference on Learning Representations*, 2020.

Tehrim Yoon, Sumin Shin, Sung Ju Hwang, and Eunho Yang. Fedmix: Approximation of mixup under mean augmented federated learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Ogga20D2HO-.

Han Yu, Zelei Liu, Yang Liu, Tianjian Chen, Mingshu Cong, Xi Weng, Dusit Niyato, and Qiang Yang. A sustainable incentive scheme for federated learning. *IEEE Intelligent Systems*, 35(4): 58–69, 2020. doi: 10.1109/MIS.2020.2987774.

Honglin Yuan, Warren Richard Morningstar, Lin Ning, and Karan Singhal. What do we mean by generalization in federated learning? In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=VimqQq-i_Q.

Yufeng Zhan, Peng Li, Zhihao Qu, Deze Zeng, and Song Guo. A learning-based incentive mechanism for federated learning. *IEEE Internet of Things Journal*, 7(7):6360–6368, 2020. doi: 10.1109/JIOT.2020.2967772.

Nanxuan Zhao, Zhirong Wu, Rynson W. H. Lau, and Stephen Lin. What makes instance discrimination good for transfer learning? In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=tC6iW2UUbJf.

Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

Huiping Zhuang, Zhenyu Weng, Fulin Luo, Toh Kar-Ann, Haizhou Li, and Zhiping Lin. Accumulated decoupled learning with gradient staleness mitigation for convolutional neural networks. In *International Conference on Machine Learning*, pp. 12935–12944. PMLR, 2021.

APPENDIX

A BROADER IMPACT

Measuring Client Contribution During Local Training. As discussed in the section 2, current works mainly focus on measuring generalization contribution from clients from participating during the whole training process. We consider measuring this contribution during each communication round, which opens a new angle toward the convergence analysis of FL. Future works may fill the generalization gap between FL and centralized training with all datasets.

Relationship between Privacy and Performance. We analyze the relationship between the sharing features and the raw data in section 4.2 and Appendix. However, we do not deeply investigate how sharing features or parameters of estimated feature distribution threatens the privacy of private raw data. Sharing features at a lower level may reduce gradient dissimilarity and high generalization performance of FL, yet leading to higher risks of data privacy. Future works may consider figuring out the trade-off between data privacy and the generalization performance with sharing features.

Connections of our work to knowledge distillation and domain generalization. The approximation of features generated based on the client data and low-level models can be seen as a kind of knowledge distillation of other clients. More in-depth analyses of this problem would be an exciting direction, which will be added to our future works. The domain generalization is also an exciting connection to federated learning. It is interesting to connect the measurements of client contribution to the domain generalization.

B PROOF

B.1 BOUNDED GENERALIZATION CONTRIBUTION

Given client m, we quantify the generalization contribution, in FL systems as follows:

$$\mathbb{E}_{\Delta:\mathbf{L}(\mathcal{D}_m)} W_d(\rho(\theta+\Delta), \mathcal{D}\backslash\mathcal{D}_m)), \tag{9}$$

where Δ is a pseudo gradient obtained by applying a learning algorithm $\mathbf{L}(\mathcal{D}_m)$ to a distribution $\mathcal{D}_m$, W_d is the quantification of generalization, and $\mathcal{D}\backslash\mathcal{D}_m)$ means the distribution of all clients except for client m.

Theorem B.1. *With the pseudo gradient Δ obtained by $\mathbf{L}(\mathcal{D}_m)$, the generalization contribution is lower bounded:*

$$\begin{aligned}\mathbb{E}_{\Delta:\mathbf{L}(\mathcal{D}_m)} W_d(\rho, \mathcal{D}\backslash\mathcal{D}_m)) \geq\ & \mathbb{E}_{\Delta:\mathbf{L}(\mathcal{D}_m)} W_d(\rho, \tilde{\mathcal{D}}_m) - |\mathbb{E}_{\Delta:\mathbf{L}(\mathcal{D}_m)} W_d(\rho, \mathcal{D}_m)) - W_d(\rho, \tilde{\mathcal{D}}_m))| \\ & - 2C_d(\mathcal{D}_m, \mathcal{D}\backslash\mathcal{D}_m),\end{aligned}$$

where $\tilde{\mathcal{D}}_m$ represents the dataset sampled from $\mathcal{D}_m$.

Proof. To derive the lower bound, we decompose the conditional quantification of generalization, i.e., $W_d(\rho(\theta+\Delta), \mathcal{D}\backslash\mathcal{D}_m)$:

$$W_d(\rho, \mathcal{D}\backslash\mathcal{D}_m) = W_d(\rho, \mathcal{D}\backslash\mathcal{D}_m) - W_d(\rho, \mathcal{D}_m) + W_d(\rho, \mathcal{D}_m) - W_d(\rho, \tilde{\mathcal{D}}_m) + W_d(\rho, \tilde{\mathcal{D}}_m), \tag{10}$$

where we denote ρ as $\rho(\theta+\Delta)$ for brevity and $\tilde{\mathcal{D}}_m$ stands for the dataset sampled from $\mathcal{D}_m$. Built upon the decomposition, we have:

$$\begin{aligned}\mathbb{E}_{\Delta:\mathbf{L}(\mathcal{D}_m)} W_d(\rho, \mathcal{D}\backslash\mathcal{D}_m)) \geq\ & \mathbb{E}_{\Delta:\mathbf{L}(\mathcal{D}_m)} W_d(\rho, \tilde{\mathcal{D}}_m) - |\mathbb{E}_{\Delta:\mathbf{L}(\mathcal{D}_m)} W_d(\rho, \mathcal{D}_m)) - W_d(\rho, \tilde{\mathcal{D}}_m))| \\ & - |\mathbb{E}_{\Delta:\mathbf{L}(\mathcal{D}_m)} W_d(\rho, \mathcal{D}\backslash\mathcal{D}_m)) - W_d(\rho, \mathcal{D}_m))|.\end{aligned} \tag{11}$$

The first term in Eq. 11 represents the empirical generalization performance. The second term in Eq. 11 means that the performance gap between the model trained on sampled dataset and that trained on the distribution, rigorous analysis can be found in (Montasser et al., 2019). Note that, the first two terms are independent on the distribution $\mathcal{D}\backslash\mathcal{D}_m)$, so the focus of generalization contribution is mainly on the last term, i.e., $|\mathbb{E}_{\Delta:\mathbf{L}(\mathcal{D}_m)} W_d(\rho, \mathcal{D}\backslash\mathcal{D}_m)) - W_d(\rho, \mathcal{D}_m))|$.

The proof is relatively straightforward, as long as we derive the upper bound of $W_d(\rho, \mathcal{D}_m)$ and $W_d(\rho, \mathcal{D}\backslash\mathcal{D}_m)$. For $W_d(f, \mathcal{D}_m)$, we have:

$$\begin{aligned}
W_d(\rho, \mathcal{D}_m) &= \mathbb{E}_{(\cdot|y)\sim\mathcal{D}_m}\mathbb{E}_{x\sim\mathcal{D}_m|y} \inf_{\rho(x')\neq y} d(x, x') = \mathbb{E}_{(\cdot|y)\sim\mathcal{D}_m}\mathbb{E}_{(x,x'')\sim J_y} \inf_{\rho(x')\neq y} d(x, x') \\
&\leq \mathbb{E}_{(\cdot|y)\sim\mathcal{D}_m}\mathbb{E}_{(x,x'')\sim J_y} \inf_{\rho(x')\neq y} d(x', x'') + d(x, x'') \\
&= \mathbb{E}_{(\cdot|y)\sim\mathcal{D}_m}\mathbb{E}_{(x,x'')\sim J_y} \inf_{\rho(x')\neq y} d(x', x'') + \mathbb{E}_{(\cdot|y)\sim\mathcal{D}_m}\mathbb{E}_{(x,x'')\sim J_y} d(x, x'') \\
&= \mathbb{E}_{(\cdot|y)\sim\mathcal{D}_m}\mathbb{E}_{x''\sim\mathcal{D}\backslash\mathcal{D}_m|y} \inf_{\rho(x')\neq y} d(x', x'') + \mathbb{E}_{(\cdot|y)\sim\mathcal{D}_m}\mathbb{E}_{(x,x'')\sim J_y} d(x, x''),
\end{aligned}$$

where J_y stands for the optimal transport between the conditional distribution $\mathcal{D}_m|y$ and $\mathcal{D}\backslash\mathcal{D}_m|y$. Similarly, we have:

$$W_d(\rho, \mathcal{D}\backslash\mathcal{D}_m) \leq \mathbb{E}_{(\cdot|y)\sim\mathcal{D}\backslash\mathcal{D}_m}\mathbb{E}_{x''\sim\mathcal{D}_m|y} \inf_{\rho(x')\neq y} d(x', x'') + \mathbb{E}_{(\cdot|y)\sim\mathcal{D}\backslash\mathcal{D}_m}\mathbb{E}_{(x,x'')\sim J_y} d(x, x'').$$

Combining these two inequality, we have:

$$|W_d(\rho, \mathcal{D}_m) - W_d(\rho, \mathcal{D}\backslash\mathcal{D}_m)| \leq 2C_d(\mathcal{D}_m, \mathcal{D}\backslash\mathcal{D}_m)) + \max\left\{\delta(\mathcal{D}_m, \mathcal{D}\backslash\mathcal{D}_m), \gamma(\mathcal{D}_m, \mathcal{D}\backslash\mathcal{D}_m)\right\}, \tag{12}$$

where

$$\delta(\mathcal{D}_m, \mathcal{D}\backslash\mathcal{D}_m) = \mathbb{E}_{(\cdot|y)\sim\mathcal{D}_m}\mathbb{E}_{x''\sim\mathcal{D}\backslash\mathcal{D}_m|y} \inf_{\rho(x')\neq y} d(x', x'') - \mathbb{E}_{(\cdot|y)\sim\mathcal{D}\backslash\mathcal{D}_m}\mathbb{E}_{x''\sim\mathcal{D}\backslash\mathcal{D}_m|y} \inf_{\rho(x')\neq y} d(x', x''),$$

and

$$\gamma(\mathcal{D}_m, \mathcal{D}\backslash\mathcal{D}_m) = \mathbb{E}_{(\cdot|y)\sim\mathcal{D}\backslash\mathcal{D}_m}\mathbb{E}_{x''\sim\mathcal{D}_m|y} \inf_{\rho(x')\neq y} d(x', x'') - \mathbb{E}_{(\cdot|y)\sim\mathcal{D}_m}\mathbb{E}_{x''\sim\mathcal{D}_m|y} \inf_{\rho(x')\neq y} d(x', x'').$$

The upper bound is straightforward. For example, if the label distributions are the same, i.e. $y \sim \mathcal{D}\backslash\mathcal{D}_m$ is equal to $y \sim \mathcal{D}_m$, we have:

$$|W_d(\rho, \mathcal{D}_m) - W_d(\rho, \mathcal{D}\backslash\mathcal{D}_m)| \leq 2C_d(\mathcal{D}_m, \mathcal{D}\backslash\mathcal{D}_m)).$$

According to Eq. 12, the last term in Eq. 11 is bounded:

$$|\mathbb{E}_{\Delta:\mathbf{L}(\mathcal{D}_m)} W_d(\rho, \mathcal{D}_m)) - W_d(\rho, \tilde{\mathcal{D}}_m))| \leq \mathbb{E}_{\Delta:\mathbf{L}(\mathcal{D}_m)} |W_d(\rho, \mathcal{D}_m)) - W_d(\rho, \tilde{\mathcal{D}}_m))|, \tag{13}$$

which is further upper bounded by conditional Wasserstein distance when the label distributions are not the same:

$$\mathbb{E}_{\Delta:\mathbf{L}(\mathcal{D}_m)} |W_d(\rho, \mathcal{D}_m)) - W_d(\rho, \tilde{\mathcal{D}}_m))| \leq 2C_d(\mathcal{D}_m, \mathcal{D}\backslash\mathcal{D}_m)) + \max\left\{\delta(\mathcal{D}_m, \mathcal{D}\backslash\mathcal{D}_m), \gamma(\mathcal{D}_m, \mathcal{D}\backslash\mathcal{D}_m)\right\}. \tag{14}$$

Thus, the label distribution will have additional impact on the bound. If the label distributions are the same, then we have

$$\mathbb{E}_{\Delta:\mathbf{L}(\mathcal{D}_m)} |W_d(\rho, \mathcal{D}_m)) - W_d(\rho, \tilde{\mathcal{D}}_m))| \leq 2C_d(\mathcal{D}_m, \mathcal{D}\backslash\mathcal{D}_m)), \tag{15}$$

which completes the proof. □

B.2 Justification of Decoupling Gradient Varaince

The derivation of Equation 4. Because $\nabla f_m = \left\{\nabla_{\theta_{low}} f_m, \nabla_{\theta_{high}} f_m\right\} \in \mathbb{R}^d$, $\nabla_{\theta_{low}} f_m \in \mathbb{R}^{d_l}$ and $\nabla_{\theta_{high}} f_m \in \mathbb{R}^{d_h}$, we have

$$\begin{aligned}
&\mathbb{E}_{\xi_m} ||\nabla f_m(\theta; \xi_m) - \nabla F(\theta)||^2 \qquad (16) \\
=&\sum_{i=1}^{d} (\nabla f_m(\theta; \xi_m)_{(i)} - \nabla F(\theta)_{(i)})^2 \\
=&\sum_{i=1}^{d_l} (\nabla f_m(\theta; \xi_m)_{(i)} - \nabla F(\theta)_{(i)})^2 + \sum_{i=d_l+1}^{d_h} (\nabla f_m(\theta; \xi_m)_{(i)} - \nabla F(\theta)_{(i)})^2 \\
=&\mathbb{E}_{\xi_m} \left[||\nabla_{\theta_{low}} f_m(\theta; \xi_m) - \nabla_{\theta_{low}} F(\theta)||^2 + ||\nabla_{\theta_{high}} f_m(\theta; \xi_m) - \nabla_{\theta_{high}} F(\theta)||^2\right]
\end{aligned}$$

The derivation of Equation 5. Assuming a multi-layers neural network consists L linear layers, each of which is followed by an activation function. And the loss function is $CE(\cdot)$. The forward function can be formulated as:

$$f(\theta, x) = CE(\tau_n(\theta_n(\tau_{n-1}(\theta_{n-1}\tau_{n-2}(...\tau_1(\theta_1 x)))))) \tag{17}$$

Then the gradient on l-th weight should be:

$$g_l = \frac{\partial f}{\partial \theta_l} = \frac{\partial f}{\partial \tau_n(z_n)} \frac{\partial \tau_n(z_n)}{\partial z_n} \frac{\partial z_n}{\partial \tau_{n-1}(z_{n-1})} \frac{\partial \tau_{n-1}(z_{n-1})}{\partial z_{n-1}} \frac{\partial z_{n-1}}{\partial \tau_{n-2}(z_{n-2})} ... \frac{\partial \tau_{l+1}(z_{l+1})}{\partial z_{l+1}} \frac{\partial z_l}{\partial \theta_l} \tag{18}$$

$$= \frac{\partial f}{\partial \tau_n(z_n)} \tau_n'(z_n)\theta_n \tau_n'(z_{n-1})\theta_{n-1} ... \tau_{l+1}'(z_{l+1})\tau_l(z_l) \tag{19}$$

$$= \frac{\partial f}{\partial \tau_n(z_n)} \left(\prod_{i=l+2}^{n} \tau_i'(z_i)\theta_i \right) \tau_{l+1}'(z_{l+1})\tau_l(z_l), \tag{20}$$

in which θ_l, τ_l, z_l, is the weight, activation function, output of the l-th layer, respectively. Thus, we can see that the gradient of l-th layer is independent of the data, hidden features, and weights before l-th layer if we directly input a z_l to l-th layer.

B.3 Proof of Theorem 4.2

We restate the optimization goals of using the private raw data of clients and the shared hidden features $\hat{h} \sim \mathcal{H}(h)$ as following:

$$\min_{\theta \in \mathbb{R}^d} \hat{F}(\theta) := \sum_{m=1}^{M} \hat{p}_m \mathbb{E}_{\substack{\xi_m \sim \mathcal{D}_m \\ \hat{h} \sim \mathcal{H}}} \hat{f}(\theta; \xi_m, \hat{h}) = \sum_{m=1}^{M} \hat{p}_m \mathbb{E}_{\substack{\xi_m \sim \mathcal{D}_m \\ \hat{h} \sim \mathcal{H}}} \left[f(\theta; \xi_m) + f(\theta; \hat{h}) \right], \tag{21}$$

Theorem B.2. *Under the gradient variance measure CGV (Definition 3), with $\hat{n}_m$ satisfying $\frac{\hat{n}_m}{n_m + \hat{n}_m} = \frac{\hat{N}}{N + \hat{N}}$, the objective function $\hat{F}(\theta)$ causes a tighter bounded gradient dissimilarity, i.e., the CGV$(\hat{F}, \theta) = \mathbb{E}_{\xi_m} ||\nabla_{\theta_{low}} f_m(\theta; \xi_m) - \nabla_{\theta_{low}} F(\theta)||^2 + \frac{N^2}{(N+\hat{N})^2} ||\nabla_{\theta_{high}} f_m(\theta; \xi_m) - \nabla_{\theta_{high}} F(\theta)||^2 \leq$ CGV(F, θ).*

Proof.

$$\begin{aligned}
\text{CGV}(\hat{F}, \theta) =& \mathbb{E}_{\substack{\xi_m \sim \mathcal{D}_m \\ \hat{h} \sim \mathcal{H}}} ||\nabla \hat{f}_m(\theta; \xi_m, \hat{h}) - \nabla \hat{F}(\theta)||^2 \\
=& \mathbb{E}_{\xi_m \sim \mathcal{D}_m} [||\nabla_{\theta_{low}} f_m(\theta; \xi_m) - \nabla_{\theta_{low}} F(\theta)||^2] \\
&+ \mathbb{E}_{\substack{\xi_m \sim \mathcal{D}_m \\ \hat{h} \sim \mathcal{H}}} [||\nabla_{\theta_{high}} f_m(\theta; \xi_m) + \nabla_{\theta_{high}} f_m(\theta; \hat{h}) - \nabla_{\theta_{high}} \bar{F}(\theta)||^2.
\end{aligned} \tag{22}$$

$$\tag{23}$$

On m-th client, the number of samples of ξ_m is n_m and the $\hat{h}_m$ is $\hat{n}_m$. Then the high-level gradient variance becomes:

$$\begin{aligned}
&\mathbb{E}_{\substack{\xi_m \sim \mathcal{D}_m \\ \hat{h} \sim \mathcal{H}}} [|| \frac{n_m}{n_m + \hat{n}_m} \nabla_{\theta_{high}} f_m(\theta; \xi_m) + \frac{\hat{n}_m}{n_m + \hat{n}_m} \nabla_{\theta_{high}} f_m(\theta; \hat{h}) - \nabla_{\theta_{high}} \bar{F}(\theta)||^2 \\
=& \mathbb{E}_{\substack{\xi_m \sim \mathcal{D}_m \\ \hat{h} \sim \mathcal{H}}} [|| \frac{n_m}{n_m + \hat{n}_m} \nabla_{\theta_{high}} f_m(\theta; \xi_m) + \frac{\hat{n}_m}{n_m + \hat{n}_m} \nabla_{\theta_{high}} f_m(\theta; \hat{h}) \\
&- \sum_{m=1}^{M} \frac{n_m + \hat{n}_m}{N + \hat{N}} (\frac{n_m}{n_m + \hat{n}_m} \nabla_{\theta_{high}} f_m(\theta; \xi_m) + \frac{\hat{n}_m}{n_m + \hat{n}_m} \nabla_{\theta_{high}} f_m(\theta; \hat{h}))||^2 \\
=& \mathbb{E}_{\xi_m \sim \mathcal{D}_m} || \frac{n_m}{n_m + \hat{n}_m} \nabla_{\theta_{high}} f_m(\theta; \xi_m) - \sum_{m=1}^{M} \frac{n_m}{N + \hat{N}} \nabla_{\theta_{high}} f_m(\theta; \xi_m)||^2 \\
=& \frac{N^2}{(N + \hat{N})^2} \mathbb{E}_{\xi_m \sim \mathcal{D}_m} ||\nabla_{\theta_{high}} f_m(\theta; \xi_m) - \sum_{m=1}^{M} \frac{n_m}{N} \nabla_{\theta_{high}} f_m(\theta; \xi_m)||^2.
\end{aligned} \tag{24}$$

Combining Equation 24 and 22, we obtain

$$\text{CGV}(\hat{F},\theta) = \mathbb{E}_{\xi_m}||\nabla_{\theta_{low}} f_m(\theta;\xi_m) - \nabla_{\theta_{low}} F(\theta)||^2 + \frac{N^2}{(N+\hat{N})^2}||\nabla_{\theta_{high}} f_m(\theta;\xi_m) - \nabla_{\theta_{high}} F(\theta)||^2,$$

which completes the proof. □

For the convergence analysis, there have been many convergence analyses of FedAvg from a gradient dissimilarity viewpoint (Woodworth et al., 2020; Lian et al., 2017; Karimireddy et al., 2019). Specifically, the convergence rate is upper bounded by many factors, among which the gradient dissimilarity plays a crucial role in the bound. In this work, we propose a novel approach inspired by the generalization view to reduce the gradient dissimilarity, we thus provide a tighter bound regarding the convergence rate. This is consistent with our experiments, see Table 2.

C More Related work

C.1 Addressing Non-IID problem in FL

The convergence and generalization performance of Federated Learning (FL) (McMahan et al., 2017) suffers from the heterogeneous data distribution across all clients (Zhao et al., 2018; Li et al., 2020b; Kairouz et al., 2019). There exists a severe divergence between local objective functions of clients, making local models of FL diverge (Li et al., 2020a; Karimireddy et al., 2019), which is called **client drift**.

Although researchers have designed many new optimization methods to address this problem, it is still an open problem. The performance of federated learning under severe Non-IID data distribution is far behind the centralized training. The previous methods that address Non-IID data problems can be classified into the following directions.

Model Regularization focuses on calibrating the local models to restrict them not to be excessively far away from the server model. A number of works (Li et al., 2020a; Acar et al., 2021; Karimireddy et al., 2019) add a regularizer of local-global model difference. FedProx (Li et al., 2020a) adds a penalty of the L2 distance between local models to the server model. SCAFFOLD (Karimireddy et al., 2019) utilizes the history information to correct the local updates of clients. FedDyn (Acar et al., 2021) proposes to dynamically update the risk objective to ensure the device optima is asymptotically consistent. FedIR (Hsu et al., 2020) applies important weight to the client's local objectives to obtain an unbiased estimator of loss. MOON (Li et al., 2021b) adds the local-global contrastive loss to learn a similar representation between clients. CCVR (Luo et al., 2021) transmits the statistics of logits and label information of data samples to calibrate the classifier.

Reducing Gradient Variance tries to correct the local updates directions of clients via other gradient information. This kind (Wang et al., 2020a; Hsu et al., 2019; Reddi et al., 2021) of methods aims to accelerate and stabilize the convergence. FedNova (Wang et al., 2020a) normalizes the local updates to eliminate the inconsistency between the local and global optimization objective functions. FedAvgM (Hsu et al., 2019) exploits the history updates of the server model to rectify clients' updates. FEDOPT (Reddi et al., 2021) proposes a unified framework of FL. It considers the clients' updates as the gradients in centralized training to generalize the optimization methods in centralized training into FL. FedAdaGrad and FedAdam are FL versions of AdaGrad and Adam.

Sharing Features. Personalized Federated Learning hopes to make clients optimize different personal models to learn knowledge from other clients and adapt their own datasets (Tan et al., 2022). The knowledge transfer of personalization is mainly implemented by introducing personalized parameters (Liang et al., 2020; Thapa et al., 2020; Li et al., 2021a), or knowledge distillation (He et al., 2020a; Lin et al., 2020; Li & Wang, 2019; Bistritz et al., 2020) on shared local features or extra datasets. Due to the preference for optimizing local objective functions, however, personalized federated models do not have a comparable generic performance (evaluated on global test dataset) to normal FL (Chen & Chao, 2021). Our main goal is to learn a better generic model. Thus, we omit comparisons to personalized FL algorithms.

Except Personalized Federated Learning, some other works propose to share features to improve federated learning. Cronus (Chang et al., 2019) proposes sharing the logits to defend the poisoning

attack. CCVR (Luo et al., 2021) transmit the logits statistics of data samples to calibrate the last layer of Federated models. CCVR (Luo et al., 2021) also share the parameters of local feature distribution. However, we do not need to share the number of different labels with the server, which protects the privacy of label distribution of clients. Moreover, our method acts as a framework for exploiting the sharing features to reduce gradient dissimilarity. The feature estimator does not need to be the Gaussian distribution of local features. One may utilize other estimators or even features of some extra datasets rather than the private ones.

Sharing Data. The original cause of client drift is data heterogeneity. Some researchers find that sharing a part of private data can significantly improve the convergence speed and generalization performance (Zhao et al., 2018), yet it sacrifices the privacy of clients' data.

Thus, to both reduce data heterogeneity and protect data privacy, a series of works (Hardt & Rothblum, 2010; Hardt et al., 2012; Chatalic et al., 2021; Johnson et al., 2018; Cai et al., 2021) add noise on data to implement sharing data with privacy guarantee to some degree. Some other works focus on sharing a part of synthetic data(Jeong et al., 2018; Long et al., 2021; Goetz & Tewari, 2020; Hao et al., 2021) or data statistics (Shin et al., 2020; Yoon et al., 2021) to help reduce data heterogeneity rather than raw data.

FedDF (Lin et al., 2020) utilizes other data and conducts knowledge distillation based on these data to transfer knowledge of models between server and clients. The core idea of FedDF is to conduct finetuning on the aggregated model via the knowledge distillation with the new shared data.

C.2 Measuring Contribution from Clients

Generalization Contribution. Clients are only willing to participate a FL training when given enough rewards. Thus, it is important to measure their contributions to the model performance (Yu et al., 2020; Ng et al., 2020; Liu et al., 2022; Sim et al., 2020).

There have been some works (Yuan et al., 2022; Yu et al., 2020; Ng et al., 2020; Liu et al., 2022; Sim et al., 2020) proposed to measure the generalization contribution from clients in FL. Some works (Yuan et al., 2022) propose to experimentally measure the performance gaps from the unseen client distributions. Data shapley (Ghorbani & Zou, 2019; Yu et al., 2020) is proposed to measure the generalization performance gain of client participation. (Liu et al., 2022) improves the calculation efficiency of Data Shapley. And there is some other work that proposes to measure the contribution by learning-based methods (Zhan et al., 2020). Our proposed questions are different from these works. Precisely, these works measure the generalization performance gap with or without some clients that never join the collaborative training of clients. However, we hope to understand the contribution of clients at each communication round. Based on this understanding, we can further improve the FL training and obtain a better generalization performance.

It has been empirically verified that a large number of selected clients introduces new challenges to optimization and generalization of FL (Charles et al., 2021), although some theoretical works show the benefits from it (Yang et al., 2020). This encourages us to understand what happens during the local training and aggregation.

Client Selection. Several works (Cho et al., 2020; Goetz et al., 2019; Ribero & Vikalo, 2020; Lai et al., 2021) propose new algorithms to strategically select clients rather than randomly. However, these methods only consider the hardware resources or local generalization ability. How local training affects the global generalization ability has not been explored.

C.3 Split Training

To efficiently train neural networks, split training instead of end-to-end training is proposed to break the forward, backward, or model updating dependency between layers of neural networks.

To break the backward dependency on subsequent layers, hidden features could be forwarded to another loss function to obtain the **Local Error Signals** (Marquez et al., 2018; Nøkland & Eidnes, 2019; Löwe et al., 2019; Wang et al., 2020b; Zhuang et al., 2021). How to design a suitable local error still remains as an open problem. Some works propose to utilize extra modules to synthesize gradients (Jaderberg et al., 2017), so that the backward and updates of different layers can be decoupled. **Features Replay** (Huo et al., 2018) is to reload the history features of the preceding

Table 4: Demystifying different FL algorithms related to the sharing data and features.

	Shared Thing	Low-level Model	Objective
(Chatalic et al., 2021; Cai et al., 2021)	Raw Data With Noise	Shared	Others
(Long et al., 2021; Hao et al., 2021)	Params. of Data Generator	Shared	Global Model Performance
(Yoon et al., 2021; Shin et al., 2020)	STAT. of raw Data	Shared	Global Model Performance
(Luo et al., 2021)	STAT. of Logis, Label Distribution	Shared	Global Model Performance
(Chang et al., 2019)	Hidden Features	Shared	Defend Poisoning Attack
(Li & Wang, 2019; Bistritz et al., 2020)	logits	Private	Personalized FL
(He et al., 2020a; Liang et al., 2020)	Hidden Features	Private	Personalized FL
(Thapa et al., 2020; Oh et al., 2022)	Hidden Features	Shared	Accelerate Training
Ours	Params. of Estimated Feat. Distribution	Shared	Global Model Performance

Note: "STAT." means statistic information, like mean or standard deviation, "Feat." means hidden features, "Params." means parameters.

layers into the next layers. By reusing the history features, the calculation on different layers could be asynchronously conducted.

Some works propose Split FL (SFL) to utilize split training to accelerate federated learning (Oh et al., 2022; Thapa et al., 2020). In SFL, the model is split into client-side and server-side parts. At each communication round, the client only downloads the client-side model from the server, conducts forward propagation, and sends the hidden features to the server for computing loss and backward propagation. This method aims to accelerate FL's training speed on the client side and cannot support local updates. In addition, sending all raw features could introduce a high data privacy risk. Thus, we omit the comparisons to these methods.

We demystify different FL algorithms related to the shared features in Table 4.

D Details of Experiment Configuration and Additional Experiments

D.1 Hardware and Software Configuration

We conduct experiments using GPU GTX-2080 Ti, CPU Intel(R) Xeon(R) Gold 5115 CPU @ 2.40GHz. The operating system is Ubuntu 16.04.6 LTS. The Pytorch version is 1.8.1. The Cuda version is 10.2.

D.2 Hyper-parameters

The learning rate configuration has been listed in Table 5. We report the best results and their learning rates (grid search in $\{0.0001, 0.001, 0.01, 0.1\}$).

And for all experiments, we use SGD as optimizer for all experiments, with batch size of 128 and weight decay of 0.0001. Note that we set momentum as 0 for baselines, as we find the momentum of 0.9 may harm the convergence and performance of FedAvg in severe Non-IID situations. We also report the best test accuracy of baselines that are trained with momentum of 0.9 in Table 6.

For $K = 10$ and $K = 100$, the maximum communication round is 1000, For $K = 10$ and $E = 5$, the maximum communication round is 400 (due to the $E = 5$ increase the calculation cost). The number of clients selected for calculation is 5 per round for $K = 10$, and 10 for $K = 100$.

Except the Figure 2 in the main paper, we provide more convergence results as Figures 5, 6, 7 and 8. These results show that our method can accelerate FL training and obtain higher generalization performance.

D.3 Additional Experiments

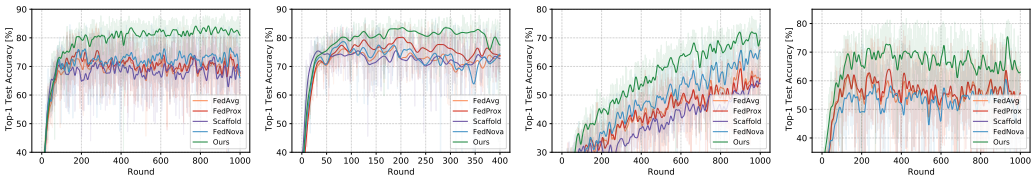
Training with Longer Time. To demonstrate the difficulty of optimization of FedAvg in heterogeneous-data environment, we show the results of training 10000 rounds, as shown in Figure 9 (a). During this 10000 rounds, the highest test accuracy of FedAvg with fixed learning rate is 88.5%, and it of the FedAvg with decayed learnign rate is 82.65%. Note that we set the learning rate decay exponentially decay at each communication round, wich rate 0.997. Even after 2000 rounds, the learning rate becomes as the around 0.0026 times as the original learning rate.

Table 5: Learning rate of all experiments.

Dataset	FL Setting			FedAvg	FedProx	SCAFFOLD	FedNova	Ours
	a	E	K					
CIFAR-10	0.1	1	10	0.1	0.1	0.1	0.1	0.1
	0.05	1	10	0.1	0.1	0.01	0.1	0.1
	0.1	5	10	0.1	0.1	0.1	0.1	0.1
	0.1	1	100	0.1	0.1	0.01	0.1	0.1
FMNIST	0.1	1	10	0.1	0.1	0.1	0.1	0.1
	0.05	1	10	0.1	0.1	0.001	0.1	0.1
	0.1	5	10	0.1	0.1	0.1	0.1	0.1
	0.1	1	100	0.1	0.1	0.01	0.1	0.1
SVHN	0.1	1	10	0.1	0.1	0.01	0.1	0.1
	0.05	1	10	0.1	0.1	0.01	0.1	0.1
	0.1	5	10	0.1	0.01	0.01	0.01	0.1
	0.1	1	100	0.1	0.1	0.001	0.1	0.1
CIFAR-100	0.1	1	10	0.1	0.1	0.1	0.1	0.1
	0.05	1	10	0.1	0.1	0.1	0.1	0.1
	0.1	5	10	0.1	0.1	0.1	0.1	0.1
	0.1	1	100	0.1	0.1	0.1	0.1	0.1

Table 6: Baselines with Momentum-SGD.

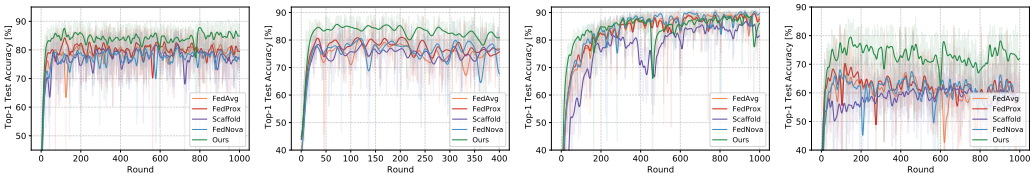
Dataset	FL Setting			FedAvg	FedProx	SCAFFOLD	FedNova
	a	E	K				
CIFAR-10	0.1	1	10	79.98	83.56	83.58	81.35
	0.05	1	10	69.02	78.66	38.55	64.78
	0.1	5	10	84.79	82,18	86.20	86.09
	0.1	1	100	49.61	49.97	52.24	46.53
FMNIST	0.1	1	10	86.81	87.12	86.21	86.99
	0.05	1	10	78.57	81.96	76.08	79.06
	0.1	5	10	87.45	86.07	87.10	87.53
	0.1	1	100	90.11	90.71	85.99	87.09
SVHN	0.1	1	10	88.56	86.51	80.61	89.12
	0.05	1	10	82.67	78.57	74.23	82.22
	0.1	5	10	87.92	78.43	81.07	88.17
	0.1	1	100	89.44	89.51	89.55	82.08
CIFAR-100	0.1	1	10	67.95	65.29	67.14	68.26
	0.05	1	10	62.07	61.52	59.04	60.35
	0.1	5	10	69.81	62.62	70.68	70.05
	0.1	1	100	48.33	48.14	51.63	48.12



(a) $a = 0.1$, $K = 10$, $E = 1$ (b) $a = 0.1$, $K = 10$, $E = 5$ (c) $a = 0.1$, $K = 100$, $E = 1$ (d) $a = 0.05$, $K = 10$, $E = 1$

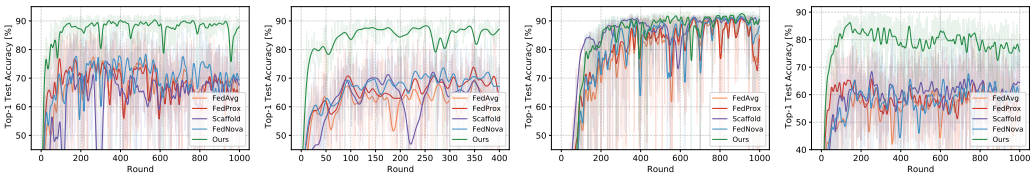
Figure 5: Convergence comparison of CIFAR-10.

Sharing Estimating Parameters with noise. To enhance the security of the sharing feature distribution, we add the noise $\epsilon \sim \mathcal{N}(0, \mu_\epsilon)$ on the σ_m and μ_m. The privacy degree could be enhanced by the larger μ_ϵ. We show the results of our method with different μ_ϵ in Figure 9 (b) and Table 7. The



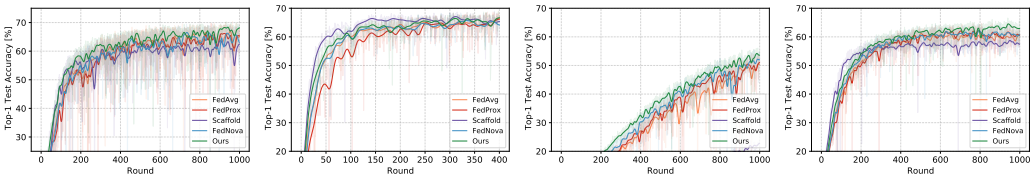
(a) $a = 0.1$, $K = 10$, $E = 1$ (b) $a = 0.1$, $K = 10$, $E = 5$ (c) $a = 0.1$, $K = 100$, $E = 1$ (d) $a = 0.05$, $K = 10$, $E = 1$

Figure 6: Convergence comparison of FMNIST.



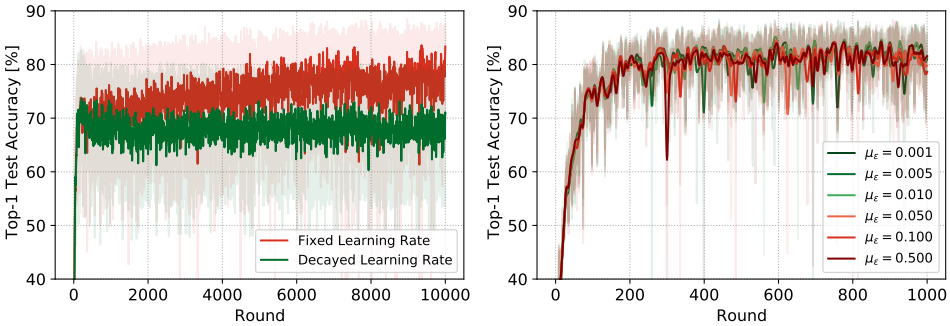
(a) $a = 0.1$, $K = 10$, $E = 1$ (b) $a = 0.1$, $K = 10$, $E = 5$ (c) $a = 0.1$, $K = 100$, $E = 1$ (d) $a = 0.05$, $K = 10$, $E = 1$

Figure 7: Convergence comparison of SVHN.



(a) $a = 0.1$, $K = 10$, $E = 1$ (b) $a = 0.1$, $K = 10$, $E = 5$ (c) $a = 0.1$, $K = 100$, $E = 1$ (d) $a = 0.05$, $K = 10$, $E = 1$

Figure 8: Convergence comparison of CIFAR100.



(a) Long Training Time (b) Our method with different degrees of noise

Figure 9: Additional experiments on CIFAR-10 with $a = 0.1$, $K = 10$, $E = 1$.

Table 7: Test Accuracy of our method with different degrees of noise.

μ_ϵ	0.0	0.001	0.005	0.01	0.05	0.1	0.5
Test Accuracy (%)	88.45	88.43	88.23	88.26	88.07	88.11	88.3

results show that under the high perturbation of the estimated parameters, our method attains both high privacy and generalization gains.