FedDC: Federated Learning with Non-IID Data via Local Drift Decoupling and Correction

Liang Gao[1] Huazhu Fu[2] Li Li[3,§] Yingwen Chen[1,§] Ming Xu[1] Cheng-Zhong Xu[3]
[1]National University of Defense Technology, China. [2]IHPC, ASTAR, Singapore. [3]University of Macau, IOTSC, China.

Abstract

Federated learning (FL) allows multiple clients to collectively train a high-performance global model without sharing their private data. However, the key challenge in federated learning is that the clients have significant statistical heterogeneity among their local data distributions, which would cause inconsistent optimized local models on the client-side. To address this fundamental dilemma, we propose a novel federated learning algorithm with local drift decoupling and correction (FedDC). Our FedDC only introduces lightweight modifications in the local training phase, in which each client utilizes an auxiliary local drift variable to track the gap between the local model parameter and the global model parameters. The key idea of FedDC is to utilize this learned local drift variable to bridge the gap, i.e., conducting consistency in parameter-level. The experiment results and analysis demonstrate that FedDC yields expediting convergence and better performance on various image classification tasks, robust in partial participation settings, non-iid data, and heterogeneous clients.

1. Introduction

Federated learning (FL) is an emerging distributed machine learning paradigm that leverages decentralized data from multiple clients to jointly train a shared global model under the coordination of a central server, without sharing the individuals' raw data [4, 8, 19, 20, 31]. This makes FL surpass traditional parallel optimization to avoid systemic privacy risk [5, 7, 15, 16, 24]. FedAvg [19] is a widely used FL aggregation algorithm, in which each client executes multiple stochastic gradient descent (SGD) steps in each communication round to minimize the local empirical risk. After that, a central server updates the parameters of the global model with the updates returned by the clients. However, recent researches [9, 17, 18] demonstrate that FedAvg could not converge well with heterogeneous data (non-iid). The data distribution of clients in FL can be highly differential because clients independently collect the local data with their own preferences and sampling space. The non-iid distributed data leads to inconsistency in clients' local objective functions and optimization directions. The studies in [9, 10] prove that the data heterogeneity introduces drift in clients' local updates, which slows down the convergence speed. The parameter drift between an FL model and a centralized learning model comes from two parts: the residual parameter drift in the last round, and the gradient drift in the current round [30]. Due to the difference in data distribution, there is a fundamental contradiction between minimizing local empirical loss and reducing global empirical loss. Therefore, in a highly heterogeneous environment, FedAvg lacks a convergence guarantee, which only obtains compromised convergence speed and model performance.

To address this client drift, some methods have been proposed to reduce the variance of local updates [9, 17]. For example, FedProx [17] adds a proximal term to force reduction of model differences between local and the global model. However, the proximal term hinders the global model from moving towards the global stationary point. Scaffold [9] corrects client-drift with a control gradient variate. However, it only approximately reduces the gradient drift in each round but it is not able to eliminate it. The residual deviation would be accumulatively amplified during training according to the research of [30], which is the primary factor that slows down the convergence speed and causes lower performance. In fact, most of the previous FL methods force the local models to be consistent to the global model. They finally get a model that neglects the inconsistency between local objectives and global objectives. They have a certain effect by reducing gradient drift, but the gradually enlarged parameter deviation persists.

We admit the fact that the local optimal points of clients are fundamentally inconsistent with the global optimal point in the heterogeneous FL setup. The local stationary points of clients can be arbitrarily different from the global stationary point. Based on this observation, we propose a new *federated learning algorithm with local drift decoupling and correction (FedDC)*, to handle the inconsistent

[§]Li Li (LLiLi@um.edu.mo) and Yingwen Chen (ywch@nudt.edu.cn) are corresponding authors.

objectives with auxiliary drift variables to track the local parameter drift between the local models and the global model. Our FedDC dynamically updates the local objective function of each client, which contains (1) a constraint penalty term that indicates the relationship among the global parameter, drift variables and the local parameters, and (2) a gradient correction term to reduce the gradient drift in each training round. We decouple the local models and the global model in the training process by introducing the drift variables, which reduces the impact of the local drift to the global objective and makes it converge quickly and reach better performance. We execute experiments on several public datasets, including MNIST, fashion MNIST, CIFAR10, CIFAR100, EMNIST-L, tiny ImageNet and a synthetic dataset. The results demonstrate that our FedDC achieves the best performance and significantly faster converge speed compared with the competing FL methods (e.g., FedAvg [19], FedProx [17], Scaffold [9] and FedDyn [1]) in both iid and non-iid client settings[*].

2. Related Work

Recently, FL has become a hot research topic [3, 8, 11]. As a pioneering work, FedAVG [19] conducts weighted parameter averaging in order to update parameters from multiple clients. The works in [10, 13] show that FedAvg reaches asymptotic convergence for homogeneous clients. However, Woodworth *et al.* [25] demonstrate that the bound of FedAVG convergence can be totally different for heterogeneous clients. The studies in [9, 18] claim that the client drift in clients' updates caused by non-iid data is the main culprit that damages convergence rates in heterogeneous settings. Prior works have shown that non-iid data would introduce challenges in FL such as gradient divergence, optimization direction biases, and unguaranteed convergence. Some works try to reduce the variance of clients' updates to speed up convergence. Minimizing the empirical risk function using a uniform global model over different clients which contains non-iid distributed data makes it difficult to converge to a splendid global model. FedProx [17] surmounts statistical heterogeneity and strengthens stability by adding a proximal regularization on the local model against the global model. The proximal term keeps the updated local parameter close to the global model, in this way it reduces potential gradient divergence. However, it violated the fact that the optimal points of local empirical objectives are different from the global optimal point, leading to a low performance. *The major limitation of these methods is that they ignore the differences in client models, leading to suboptimal performance and slowly converge speed in non-iid data distributions.*

[*]The code is available at https://github.com/gaoliang13/FedDC

In order to further analyze the correlation between client drift and data heterogeneity, some works conduct personalized local objectives with statistical variables. Scaffold [9] customizes gradients for each client to fix the client drifts between local models and the global model. Similarly, FedDyn [1] proposes a dynamic regularizer for each device to align the global and device solutions and save transmission costs. Another type of work tries to optimize the parameter aggregation step on the central server to get a better global model. [29] dynamic calculates the optimally weighted combination of clients' local model by figuring out how much a client can benefit from the global model. Reddi *et al.* [22] propose federated adaptive optimization based on the interplay between client heterogeneity and communication efficiency to prevent unfavourable convergence behaviour. Yang *et al.* [27] achieve linear speedup with non-iid data with two-sided learning rates in local update and global update. These methods are compatible with our method, which could be easily integrated into our method. These improved methods achieve better speedup in convergence and enjoy better performance than FedAvg. However, the theory of Zhao *et al.* [30] indicates that the parameter deviation would be accumulated and cause a suboptimal solution. *In this paper, we propose the FedDC, which decouples the local and global models by tracking and bridging the local drift.*

3. Local Drift in Federated Learning

In FL, we assume that there are N clients in a federation, and suppose D_i is client i's private local dataset. The goal is to get a global model w^* training over the global dataset $D = \bigcup_{i\in[N]} D_i$ that solves the objective:

$$w^* = \arg\min_{w} L(w) = \sum_{i=1}^{N} \frac{|D_i|}{|D|} L_i(w), \tag{1}$$

where w is the parameter of the global model, $L(w)$ is the empirical loss on the global dataset D, $|D_i|$ is the number of samples on D_i, $|D|$ is the number of samples on D, $L_i(w) = \mathbb{E}_{(x,y)\in D_i} l(w; (x, y))$ is the local empirical loss on client i's local dataset $|D_i|$. In order to avoid privacy leaking, any client can not share its raw data with others. FedAvg is proposed to coordinate multiple clients to cooperatively train the global model with a central server while preserving data privacy [19]. Specifically, in FedAvg, for each training round, all clients optimize their local models on the local datasets, then the server takes the expectation of the local model parameters to update the global model as follows:

$$w = \sum_{i=1}^{N} \frac{|D_i|}{|D|} \theta_i, \tag{2}$$

where w is the global model parameter, θ_i is client i's local model parameters. Then, the updated global model param-

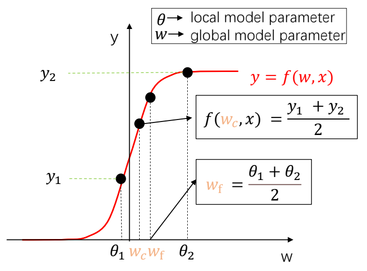


Figure 1. Illustration of the local drift in FedAvg with a $Sigmod$ activation function f. w_c is the parameter of the model trained with centralized data (ideal model), w_f is the parameter of the model generated by FedAvg. θ_1 and θ_2 are the parameters of local models of client 1 and client 2, respectively.

eter is broadcast to clients and utilized as the start point of local models in the next round.

There is a drift between each client's local model trained on the local dataset and the global model trained on the global dataset directly [17, 23]. If the drift is ignored, the server would get an skewed global model. FL faces the challenge of heterogeneous data. With the highly skewed non-IID data in FL, the performance of FedAvg is significantly reduced [17, 23], which indicates that the FedAvg method that ignores local drift leads to the deviation of the global model. In Figure 1, we show a simple example that client's local drift would result in a biased global model in FedAvg. We suppose that there is a non-linear transformation function f (e.g., $Sigmoid$ function in the activation layer) in the model. Suppose θ_1 and θ_2 are local parameters of client 1 and client 2, w_c is the ideal model parameter and w_f is the model parameter generated through FedAvg. The local drifts (denoted as h) of client 1 and client 2 are $(h_1 = w_c - \theta_1)$ and $(h_2 = w_c - \theta_2)$, respectively. x is a data point, the corresponding outputs on client 1 is $y_1 = f(\theta_1, x)$ and $y_2 = f(\theta_2, x)$ on client 2. Then the model parameter generated by FedAvg can be represented as $w_f = \frac{\theta_1+\theta_2}{2}$. The centralized model is an ideal model that would get the ideal output, that is, $f(w_c, x) = \frac{y_1+y_2}{2}$. Thus, the parameter of centralized model is $w_c = f^{-1}(\frac{y_1+y_2}{2})/x$, where f^{-1} is the inverse function of f. Since f is a non-linear function, we have $w_f \neq w_c$ and $f(w_f, x) \neq \frac{y_1+y_2}{2}$. That indicates the global model in FedAvg is skewed, which is likely to converge slowly and with poor accuracy. Therefore, *we can learn the local drift between the global model and the local model, and bridge the local drift before uploading the local model parameters to the server.* This is in line with the intuition of FL.

4. Proposed Method

Based on the above observation, we propose a novel federated learning algorithm with local drift decoupling and correction (FedDC), which aims to improve the robustness and speed of model convergence by learning the model drift and bridging the drift on the client-side. Our FedDC introduces lightweight modifications in the training phase to decouple the global model from clients' local models using the local drift. Specifically, in the local training phase, each client learns a local drift variable that represents the gap between its local model and the global model. Then, the local drift variable is used to correct the local model parameters before the parameter aggregating phase. In this way, FedDC decreases the distance between the local model parameters and the global model parameters, which also decreases the negative influence of the skewed local model on the global model.

4.1. Objectives in FedDC

First, we define a local drift variable h_i for each client. In an ideal condition, the local drift variable should satisfy the restriction: $h_i = w - \theta_i$, where θ_i is the parameter of client i'local model, and w is the parameter of the global model. In the whole training process, we need to keep this restriction to prevent the local drift variable from getting out of our control. Therefore, for client i, we further convert this restriction as a penalized term as:

$$R_i(\theta_i, h_i, w) = ||h_i + \theta_i - w||^2, \forall i \in [N]. \tag{3}$$

Each client utilizes this penalized term with its empirical loss term on the corresponding dataset to train the model parameters and the local drift variables. In this way, we transform an equation-constrained optimization problem into an unconstrained optimization problem.

In FedDC, the objective function of each client contains three components: the local empirical loss term, the penalized term, and a gradient correction term. Specifically, for client i ($\forall i \in [N]$), the local objective of θ_i is to minimize the following objective function:

$$F(\theta_i; h_i, D_i, w) = L_i(\theta_i) + \frac{\alpha}{2} R_i(\theta_i; h_i, w) + G_i(\theta_i; g_i, g), \tag{4}$$

where L_i is the typical empirical loss, R_i is the penalized term in Eq. 3, α is a hyper-parameter that controls the weight of R_i, and G_i is the gradient correction term that controls the gradient stochastic optimization. Inspired by Scaffold [9], we set the gradient correction term as $G_i(\theta_i; g_i, g) = \frac{1}{\eta K}\langle\theta_i, g_i - g\rangle$, where η is the learning rate, K is the amount of training iterations in one round. g_i is the local update value of i-th client's local parameters in last round, g is the average update value of all clients' local parameters in last round. In t-th round, we have $g_i = \theta_i^t - \theta_i^{t-1}$ and $g = \mathbb{E}_{i\in[N]} g_i$, where θ_i^t and θ_i^{t-1} are client i's local model parameters in t-th round and $(t-1)$-th round, respectively. The role of term G_i is to reduce the variance of local gradients.

Updating the local model parameters. At the beginning of each round, the server first sends the global parameters of the previous round to all clients. Each client i ($\forall i \in [N]$) loads the global model parameter to the local model (set $\theta_i = w$) and then updates the local model by minimizing the objective function in Eq. 4. We assume each training round contains K local training iterations, in k-th local training iteration of t-th round, the local model parameter is updated as follows:

$$\theta_i^{t,k+1} = \theta_i^{t,k} - \eta \frac{\partial F(\theta_i^{t,k}; h_i^t, D_i, w^t)}{\partial \theta_i^{t,k}}, \tag{5}$$

where η is the learning rate. The Eq. 5 is executed K times in each round.

Updating the local drift variables. Then, we introduce the updating method of the local drift variable h_i. We use the superscripted $^+$ symbol to indicate the updated parameters at K-th local iteration. In FedDC, the local drift variables track the gaps between local models and the global model. In a training round, we suppose the global model parameter w is updated to w^+ while the local model parameter is fixed. Then we can update the local drift variable using $h_i^+ = h_i + (w_i^+ - w)$. However, it is impossible to update the global model directly due to the unavailable global data.

Another way to optimize h_i is minimizing the objective loss using the partial derivative of h_i in Eq. 4 with θ_i and w fixed on the client-side. However, that costs K training iterations of back-propagation. In order to reduce the calculation, assuming that we have first updated the local model parameters from θ_i to θ_i^+ which is a must-do step. Then we consider the following two points: 1) at the beginning of each round, the local model parameters is assigned with the global model parameter: $\theta_i = w$. 2) for client i, the local model parameter θ_i^+ is an estimation of the updated global model parameter w_i^+. Thus, instead of $h_i^+ = h_i + (w_i^+ - w)$, we can approximately update the local drift variable using:

$$h_i^+ = h_i + (w_i^+ - w_i) \approx h_i + (\theta_i^+ - \theta_i), \tag{6}$$

where θ_i in Eq. 6 is the shorthand of $\theta_i^{t,0}$ and θ_i^+ the shorthand of $\theta_i^{t,K}$ in t-th round. In this way, we reuse the updates of local model parameter to update the local drift and avoid performing the back-propagation process for h_i.

Updating the global model parameters. To update the global model parameters, before the model aggregation phase each client corrects its local model parameters using the local drift variables: $(\theta_i^+ + h_i^+)$. Then each client uploads the corrected local parameters to the server. Similar to FedAvg, the server performs a weighted average of the corrected local parameters to obtain the global model parameters:

$$w^+ = \sum_{i=1}^{N} \frac{|D_i|}{|D|}(\theta_i^+ + h_i^+), \tag{7}$$

where $|D_i|$ is the sample amount on client i, w^+ is the updated global model.

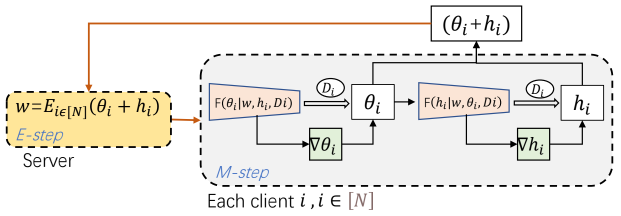


Figure 2. The training procedure of FedDC using Expectation-Maximum (EM) algorithm. In each round, the local parameters and the global parameters are iteratively updated on the client-side (M-step) and the server-side (E-step) respectively.

4.2. Training Process

We summarize the training procedure of FedDC with the Expectation-Maximum (EM) algorithm. The EM algorithm is used to solve the parameter optimization problem in the case there is missing information. In FedDC, the traditional machine learning method of directly optimizing parameters is not applicable as there are three types of variables. Moreover, the local parameters and the global parameters are updated on different devices. We can iteratively fix two variables while optimizing the other one at a time. In this way, we seek the extreme value of one variable one step, and finally, approach the extreme value of these variables step by step. The training process of FedDC is shown in Figure 2. In each round, we execute the Maximization step (M-step) on the client-side to optimize the local model parameter θ_i and the local drift variables h_i. Then we execute the Expectation step (E-step) on the server-side to update the global model parameter w.

4.3. Convergence of FedDC

We proved the convergence of FedDC in non-convex case. For non-convex and β-Lipschitz smooth local empirical loss function $L_i, \forall i \in [N]$, there exists a $\beta_d > 0$, where $\bar{\alpha} = \alpha - \beta_d > 0$ and $\nabla^2 L_i \geq -\beta_d I$. We assume the local empirical loss L_i is non-convex and B-dissimilarity, in which $B(\theta^t) \leq B$. The global empirical loss of FedDC decreases as follows:

$$\mathbb{E}_{C_t} L(w^t) \leq L(w^{t-1}) - 2p||\nabla L(w^{t-1})||^2, \tag{8}$$

where $p = (\frac{\gamma}{\alpha} - \frac{B(1+\gamma)\sqrt{2}}{\bar{\alpha}\sqrt{N}} - \frac{\beta B(1+\gamma)}{\alpha\bar{\alpha}} - \frac{\beta(1+\gamma)^2 B^2}{2\bar{\alpha}^2} - \frac{\beta B^2(1+\gamma)^2(2\sqrt{2C}+2)}{\bar{\alpha}^2 N}) > 0$, and C_t is the active client set in round t which contains C clients. The more details of the convergence guarantee are provided in Appendix B.

4.4. Discussion

Our FedDC appears has similar goal with the previous methods like SCAFFOLD, FedProx and FedDyn as they all try to reduce the gap between the local model parameters and the global model parameters caused by non-iid data, but there are fundamental differences. The general approach of the previous methods (e.g. SCAFFOLD, FedProx and FedDyn) is to limit the local optimization direction to reduce the parameter gap between the local models and the global model, that is, restricting θ_i to be close to w (that is $\min ||\theta_i - w||$). However, restricting the optimization direction of the local model hinder it in fitting the local dataset distributions, because the local distribution and global distribution can be inconsistent. In FedDC, we think learning the parameter gap is better than limiting it. FedDC utilizes the local drift variable to learn the parameter gap between the local model and the global model. And then the local drift variable is used to bridge the gap, where we learn the local drift h_i to achieve the goal $\min ||\theta_i + h_i - w||$. In other words, FedDC does not hinder the local models from learning local features and minimizing the local empirical risks. We attribute the advantages of FedDC to that it learns the local drift and well bridges the parameter gap without hindering the local training process.

5. Experiments

In this section, we evaluate the effectiveness of FedDC and compare FedDC with several advanced methods in various datasets and settings. Specifically, the evaluation is mainly conducted from two perspectives: 1) convergence speed and 2) model accuracy. Due to the space limitation, more detailed experiment results and the ablation study are given in Appendix A.

5.1. Dataset and Baselines

We explore on six benchmark datasets: MNIST [14], fashion MNIST [26], CIFAR10, CIFAR100 [12], EMNIST-L [2], Tiny ImageNet [21] and the Synthetic [17] datasets. For all of them, we adopt the same training/testing splits as previous works [1, 17, 19]. In the iid setting, training samples are randomly selected and equally assigned to clients. All the clients have the same amount of training data, and each client's data points are evenly distributed in all categories. In the non-iid data settings, the label ratios follow the Dirichlet distribution [28]. We set two non-iid data settings, and they are denoted as D1 and D2 in which the Dirichlet parameters are 0.6 and 0.3 respectively. Besides, we produce unbalanced data by samplings samples with a lognormal distribution, in which we set the variance as 0.3. For the Synthetic dataset, following the setting in [1], we generate three types of data settings, including homogeneity setting which is denoted as "Synthetic(0,0)", objective heterogeneity setting which is denoted as "Synthetic(1,0)", data heterogeneity setting which denoted as "Synthetic(0,1)". More detailed settings are given in Appendix A.

We verify the experimental results based on four network architectures in order to emphasize the versatility of the proposed method. We use a multi-class logistic classification model for the Synthetic dataset. For the MNIST digit classification task, the same fully-connected network (FCN) is adopted as [19]. A convolutional neural network (CNN) is adopted to classify the samples on CIFAR10 and CIFAR100, as used in [19]. On Tiny ImageNet, a pre-trained ResNet18 [6] is adopted to show the efficiency of FedDC on the pre-trained models.

We compare FedDC with several advanced methods, including FedAvg [19], FedProx [17], Scaffold [9] and FedDyn [1]. FedProx uses the proximal term to reduce the gradient variance. Scaffold attempts to correct the local updates with a gradient correction term, and FedDyn aligns the client models using a dynamic regularizer. Different from FedDC, these methods all emphasize the consistency of client models and the global model and ignore the local drift in the parameter aggregation phase.

5.2. Hyper-parameter Settings

We apply the typical FL architecture, where multiple clients get their local updates in each communication round through training models with their local datasets, and a central server aggregates client updates to update the global model. We utilize the SGD algorithm as the local optimizer for all methods. In addition, in order to maintain consistency, for all methods on the true world datasets, we set batch size as 50 in the local training phase, the local training epochs as 5 in each round, the initial learning rate as 0.1, and the decay rate as 0.998. All the above settings follow the previous work [1]. We set the hyper-parameter $\alpha = 0.01$ of FedDC on CIFAR10, CIFAR100 and Tiny ImageNet, $\alpha = 0.1$ of FedDC on MNIST, fashion MNIST and EMNIST-L. In the Synthetic dataset, we set the number of clients as 20 and the local batch size as 10, $\alpha = 0.005$ for FedDC. As for specific hyper-parameters of the baselines, we keep the same settings as their referred papers. We set FedDyn's hyper-parameter $\alpha = 0.01$ and FedProx's hyper-parameter $\mu = 10^{-4}$. If there are parameter settings different from the above described, it will be specifically explained in the Appendix. We also explore the effect of different values of α in FedDC (See Appendix A).

5.3. Results and Analysis

We run vast experiments to determine the superiority of FedDC on the convergence speed and the model performance. Besides, we also demonstrate the robustness and superiority of FedDC in different participation levels, different client scale and different data heterogeneity. All results are reported based on the global model. As the baselines and FedDC consume the same computational resource in each round, so that we report the number of communication rounds instead of the FLOPS. The goal of FedDC mainly includes two perspectives: (1) speeding the model convergence rate to reduce the communication cost, and (2) improving the model performance trained on different

Table 1. The communication rounds in different methods to achieve the same target accuracy. The left half is the result of full participation, and the right is the result of partial participation, each of which includes one iid setting and two non-iid settings where the 0.6-Dirichlet non-iid setting is denoted as "D1", and the 0.3-Dirichlet non-iid setting is denoted as "D2". In addition, we denote the communication round of each method to achieve the target accuracy as "$R\#$", the corresponding convergence speedup relative to FedAvg as "$S\uparrow$". We use $>$ sign to represent the method that could not achieve the target accuracy within the communication constraint.

Model	Full Participation						Partial Participation (15%)					
	D1		D2		iid		D1		D2		iid	
	$R\#$	$S\uparrow$	$R\#$	$S\uparrow$	$R\#$	$S\uparrow$	$R\#$	$S\uparrow$	$R\#$	$S\uparrow$	$R\#$	$S\uparrow$
MNIST, 100 client, Target accuracy 98%												
FedAvg	258	-	492	-	142	-	361	-	>600	-	158	-
FedProx	263	0.98×	480	1.03×	136	1.04×	383	0.94×	418	1.44×	149	1.06×
Scaffold	58	4.45×	58	8.48×	53	2.68×	62	5.82×	72	8.33×	50	3.16×
FedDyn	46	5.61×	51	9.65×	27	5.26×	122	2.96×	153	3.92×	71	2.23×
FedDC	35	7.37×	37	13.3×	26	5.46×	60	6.02×	62	9.68×	46	3.43×
fashion MNIST, 100 client, Target accuracy 89%												
FedAvg	>300	-	273	-	112	-	>300	-	>300	-	144	-
FedProx	>300	1×	>300	0.91×	130	0.86×	>300	1×	>300	1×	128	1.13×
Scaffold	117	2.56×	169	1.61×	85	1.32×	133	2.26×	>300	1×	108	1.33×
FedDyn	150	2×	211	1.29×	38	2.95×	>300	1×	267	1.12×	85	1.69×
FedDC	86	3.49×	126	2.17×	24	4.67×	87	3.49×	252	1.19×	63	2.29×
EMNIST-L, 100 client, Target accuracy 94%												
FedAvg	142	-	192	-	107	-	153	-	245	-	108	-
FedProx	135	1.05×	198	0.97×	92	1.16×	145	1.06×	240	1.02×	105	1.03×
Scaffold	43	3.30×	52	3.69×	30	3.57×	44	3.48×	68	3.6×	42	2.57×
FedDyn	30	4.73×	52	3.69×	27	3.96×	73	2.1×	81	3.06×	61	1.61×
FedDC	43	3.3×	60	3.2×	21	5.1×	48	3.19×	74	3.31×	47	2.3×
CIFAR10, 100 client, Target accuracy 80%												
FedAvg	>1000	-	>1000	-	286	-	616	-	>1000	-	>1000	-
FedProx	474	2.11×	>1000	1×	277	1.03×	459	1.34×	>1000	1×	307	3.28×
Scaffold	165	6.06×	218	4.59×	120	2.38×	200	3.08×	263	3.80×	126	7.93×
FedDyn	60	16.67×	75	17.54×	55	5.2×	193	3.19×	195	5.12×	145	6.9×
FedDC	53	18.86×	70	14.28×	43	6.65×	141	4.37×	143	6.99×	108	9.26×
CIFAR100, 100 client, Target accuracy 40%												
FedAvg	476	-	847	-	>1000	-	615	-	520	-	724	-
FedProx	502	0.95×	507	1.67×	273	3.66×	980	0.63×	503	1.03×	650	1.11×
Scaffold	91	5.23×	94	9.01×	84	11.9×	106	5.8×	114	3.56×	113	6.41×
FedDyn	51	9.33×	53	15.98×	56	17.85×	149	4.42×	148	3.51×	143	5.06×
FedDC	39	12.2×	41	20.65×	37	27.03×	102	6.03×	103	5.05×	100	7.04×

Table 2. The top-1 test accuracy on Tiny ImageNet with 20 clients training for 10 rounds on iid and non-iid settings.

Method	D1	D2	iid
FedAvg	43.86	42.62	44.30
FedProx	43.55	42.25	44.11
Scaffold	44.38	43.38	45.07
FedDyn	45.37	44.71	45.61
FedDC	**46.44**	**46.60**	**47.91**

datasets. Our results highlight the benefit of FedDC compared to the existing FL optimization approaches.

Fast convergence of FedDC. Table 1 compares the convergence speed of FedDC and the mentioned baselines. The results show that FedDC is the best one to handle the local drift and speeds up the convergence speed compared with other methods. Specifically, FedDC could achieve a target accuracy using fewer communication rounds than the FedAvg, FedProx, Scaffold and FedDyn. For instance, in the iid setting, FedDC spends 37 communication rounds to achieve 40% accuracy while 100 clients full participating in training on CIFAR100, while FedAvg spending over 1000 rounds to achieve 40% accuracy in the same setting. That is, the convergence speed of FedDC relative to FedAvg is faster over 27.03×. We may attribute this to the fact that FedDC bridges the local drift and efficiently optimizes the objectives. The convergence speedup also leads to proportional communication-saving. And Figure 3 shows more vivid results of the convergence plots, in which FedDC is consistently the fastest one in all settings. Figure 3 (a, d) show the convergence plots in iid settings on CIFAR10 and CIFAR100. Figure 3 (b, e) are accuracy plots on non-iid settings. From these convergence plots, we intuitively observe that FedDC achieves better accuracy and greatly speeds up the convergence speed than baselines. It is obvious that convergence speedup of FedDC relative to baselines is larger on non-iid settings than in iid settings. As the increasing of data heterogeneity, the local models suffer from more significant client drift. FedDC handles the drift by bridging the gap using the local drift variables that are learned on the client-side, so that FedDC show an obvious advantage in convergence speed over other baselines. The results confirm that FedDC has a stronger ability to handle heterogeneous data. Figure 3 (c, f) are convergence plots on unbalanced data set settings. The unbalanced data introduces another type of system heterogeneity, making the

Table 3. The top-1 test accuracy (%) on iid, non-iid and unbalanced data for full client participation and partial client participation (15%) levels. There are three settings for the amount of clients: Setting 1 (100 clients), Setting 2 (500 clients) and Setting 3 (20 clients).

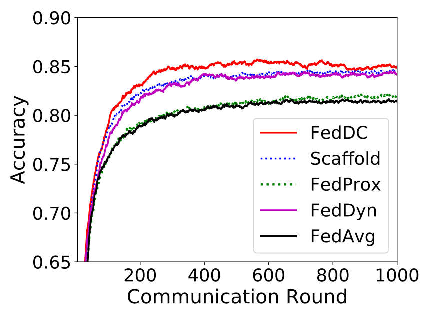
Method	FedAvg	FedProx	Scaffold	FedDyn	FedDC	FedAvg	FedProx	Scaffold	FedDyn	FedDC
Setting 1	100 clients full participation					100 clients partial participation				
CIFAR10-iid	82.16	81.85	84.61	85.26	**86.18**	81.67	82.16	84.68	84.50	**85.71**
CIFAR10-D1	80.42	80.70	84.13	85.26	**85.64**	81.05	81.32	83.57	84.10	**84.77**
CIFAR10-D2	79.14	78.89	82.96	84.14	**84.32**	79.77	79.84	82.53	82.30	**84.58**
CIFAR10-unbalance	81.37	81.90	84.45	85.68	**86.31**	81.68	81.88	84.44	84.30	**85.35**
CIFAR100-iid	39.68	40.39	51.26	52.07	**55.52**	40.80	40.67	49.80	51.20	**55.40**
CIFAR100-D1	40.48	40.15	51.16	52.84	**55.34**	41.76	41.83	50.01	51.75	**54.65**
CIFAR100-D2	40.11	40.93	50.44	51.89	**54.86**	41.81	41.84	50.25	51.13	**53.91**
CIFAR100-unbalance	40.03	39.93	51.30	52.81	**55.69**	40.90	41.05	50.57	51.01	**55.27**
MNIST-iid	98.12	98.12	98.32	**98.51**	98.45	98.15	98.11	98.45	98.38	**98.47**
MNIST-D1	98.09	98.05	98.39	98.44	**98.48**	98.13	98.12	98.45	98.30	**98.49**
MNIST-D2	97.98	97.96	98.45	98.46	**98.51**	98.00	98.04	98.37	98.30	**98.40**
MNIST-unbalance	98.12	98.10	98.35	**98.60**	98.46	98.15	98.13	98.50	98.34	**98.53**
Setting 2	500 clients full participation					500 clients partial participation				
CIFAR10-iid	73.43	72.77	81.56	84.07	**84.93**	73.26	72.58	81.58	82.49	**84.19**
CIFAR100-iid	26.03	28.22	45.62	50.22	**54.25**	27.36	26.50	30.45	44.11	**50.61**
Setting 3	20 clients full participation					20 clients partial participation				
Synthetic(0,0)	98.65	98.65	98.65	99.25	**99.35**	98.75	98.70	98.65	99.32	**99.57**
Synthetic(1,0)	97.83	97.82	97.90	98.65	**98.83**	97.70	97.67	97.90	98.82	**99.23**
Synthetic(0,1)	97.75	97.75	97.90	99.10	**99.30**	98.52	98.50	98.58	99.30	**99.62**

convergence speed slower than in the balanced data. The results show FedDC's superiority in both model performance and convergence speed in unbalanced settings, we find that FedDC also has the potential to handle the heterogeneity caused by unbalanced data. In addition, a widespread trend in these figures is that as the target accuracy improves, the communication-saving of FedDC relative to other methods become bigger. Another trend is that the improvement of FedDC over baselines in CIFAR100 is bigger than in CIFAR10 in the same settings. We attribute it to the fact that as the difficulty of optimization increases, FedDC's robustness advantage over other methods is further highlighted. FedDC can utilize the local drift variable to capture the system heterogeneity in clients' local datasets and capture the subtle features needed to classify confusing samples.

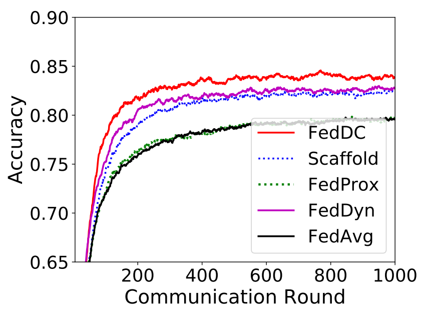
Better performance of FedDC. Table 3 compares the best accuracy of FedDC with baselines on evaluation datasets with various settings. On CIFAR10 and CIFAR100, FedDC always achieves the best test accuracy, where FedAVG and FedProx have the least. For instance, when training on the data of 0.3-Dirichlet distribution (Dw) CIFAR10 with 100 clients full participating, the test accuracy of FedDC is 84.32%, the accuracy of FedAvg achieves 79.14% and the accuracy of Scaffolf achieves 82.96%. FedDC also achieves appreciable improvement in top-1 test accuracy on the unbalanced settings. Besides, the results in setting 2 (500 clients) and setting 3 (20 clients) indicate that FedDC is efficient in the practically relevant massively distributed settings. The improvements of FedDC indicates that tracking and correcting client drift effectively prevent the model performance from decreasing. Compared with Scaffold, FedDC not only uses the gradient correction term to reduce gradient drift but also introduces the local drift variable to track the deviation between the global model and local models, so that FedDC is the best one to prevent the accuracy reduction. Table 2 shows the accuracy of ResNet18 that training for 15 rounds on Tiny ImageNet, where the ResNet18 is started from an ImageNet pre-trained model. The performance of FedDC significantly outperforms the baselines in all settings. This shows that FedDC is still efficient in tasks that use pre-trained models.

Robustness on heterogeneous data. A more extensive non-iid data or unbalanced data can greatly slow down the model convergence [19]. Comparing the convergence plots of Figure 3 (a,b,d,e), the results show that the data distribution has a prominent influence on both the model convergence speed and accuracy. It reveals that the convergence speed on the iid data is faster than on the non-iid data in which the local dataset can not well approximate the overall distribution. As shown in Table 3, FedDC outperforms baselines on iid, non-iid and unbalanced settings. FedDC gets more communication-saving gains relative to other methods when we increase the target accuracy or training on a harder task. The data heterogeneity does damage to the model performance of all methods. While training with 100 clients and full participation on CIFAR100, the accuracy of FedDC is 85.71% in iid setting, 84.77% in 0.6-Dirichlet (D1) distribution, and 84.58% in 0.3-Dirichlet distribution (D2, it is more non-iid than 0.6-Dirichlet). However, even in these heterogeneous data settings, FedDC maintains its competitive advantage compared with baselines because it is able to neutralize the local drifts.

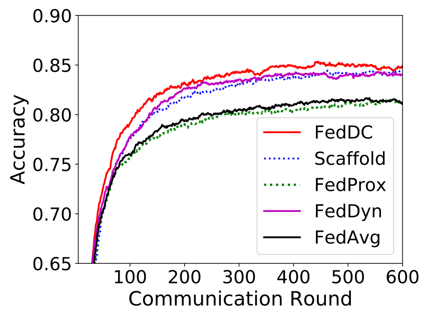
Robustness to massive clients. We conduct experiments to analyze the effectiveness of FedDC while adopting different amounts of clients to participate in the training process. We report the model accuracy in Table 3 with 100 and 500 clients of both partial participation and full participation on CIFAR10 and CIFAR100 datasets. FedDC



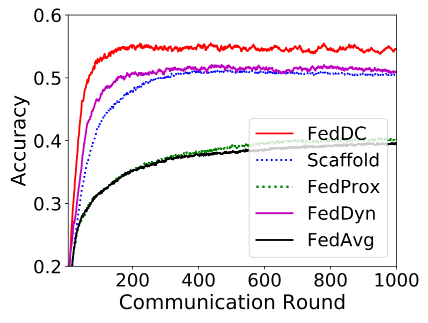
(a) CIFAR10, iid, partial participation



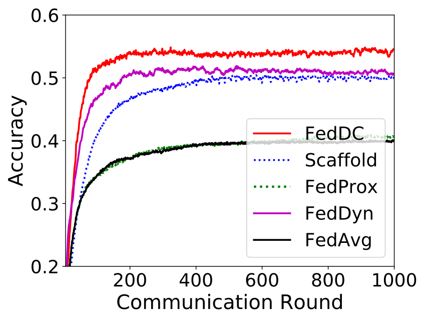
(b) CIFAR10, D2, partial participation



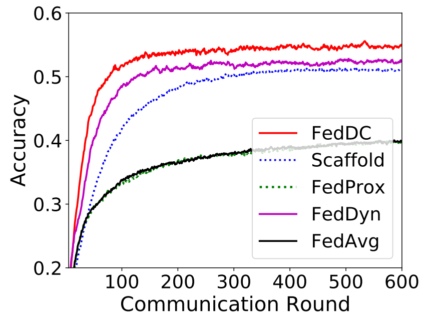
(c) CIFAR10, unbalance, partial participation



(d) CIFAR100, iid, full participation



(e) CIFAR100, D2, full participation



(f) CIFAR100, unbalance, full participation

Figure 3. Convergence plots for FedDC and other baselines in different settings that with 100 clients partial (15%) client participating on iid, D2 non-iid (Dirichlet-0.3) and unbalanced data of CIFAR10 and CIFAR100 datasets. (a), (b) and (c) are training on CIFAR10 with partial participation. (d), (e) and (f) are training on CIFAR100 with full participation.

achieved the best performance consistently. FedDC converges to a better stationary point than other methods. In the setting with 100 clients (setting 1) and full client participating, the test accuracy of FedDC is 84.93% on CIFAR10, while FedAVG only achieves 74.43% (11.5% lower than FedDC) on CIFAR10. Scaffold and FedDyn methods always get intermediate accuracy. Moreover, the performance gap between FedDC and other methods increases when the client size increases from 100 to 500. We attribute it to that a smaller number of samples per device (with massive clients) brings a greater risk of optimization dispersion.

Robustness to client sampling. The devices in FL are heterogeneous and flexible, which may join and exit at any time. To show that FedDC is resilient for clients sampling, we set the experiments with full participation and partial sampling participation (in this setting we randomly sample 15% client join training each round). We compared the final performance of FedDC and the baseline algorithms in Table 3. Partial client participating means the active data is only a subset of all training data, which leads to unstable and slower convergence. In full clients participating, the accuracy of FedDC with 100 clients on iid CIFAR10 is 86.18%, and in 15% client sampling, the accuracy decreases to 85.71%. Moreover, the results turn out that keeping all clients active is not necessary for FedDC, where the partial client participating could achieve similar accuracy as the full client participating. FedDC keeps the best accuracy in partial client participation compared to the other methods. Thus, FedDC is much resilient to client sampling compared to baselines as it utilizes the clients' parameter deviations to improve the performance of the global model. The clients in FedDC hold and update drift variables locally, so that occasionally interrupted training does not cause the loss of the drift state, which allows clients to train better in partial client participation settings.

6. Conclusion

In this work, we proposed a novel FL algorithm with local drift decoupling and correction, named FedDC, to solve the problem of local drift which caused by the heterogeneous data. FedDC dynamically bridges the gap between the local model and the global model with the learned local drift variable. Through extensive experiments on various image classification datasets, we demonstrated that our FedDC provides better performance and faster model convergence in FL. Moreover, FedDC is robust and efficient in homogeneous or heterogeneous data, in both full client participation and partial client participation.

Acknowledgement The work is supported by the National Natural Science Foundation (NSF) under grant 62072306 and 61872372, Open Fund of Science and Technology on Parallel and Distributed Processing Laboratory under grant 6142110200407, and A*STAR AI3 HTPO Seed Fund (C211118012).

References

[1] 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. 2, 5

[2] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. EMNIST: Extending MNIST to handwritten letters. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 2921–2926, 2017. 5

[3] Ittai Dayan, Holger R. Roth, et al. Federated learning for predicting clinical outcomes in patients with COVID-19. *Nature Medicine*, 27(10):1735–1743, oct 2021. 2

[4] Chun-Mei Feng, Yunlu Yan, Huazhu Fu, Yong Xu, and Ling Shao. Specificity-Preserving Federated Learning for MR Image Reconstruction. *arXiv*, dec 2021. 1

[5] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour. *arXiv e-prints*, page arXiv:1706.02677, June 2017. 1

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 5

[7] Forrest N. Iandola, Matthew W. Moskewicz, Khalid Ashraf, and Kurt Keutzer. FireCaffe: Near-Linear Acceleration of Deep Neural Network Training on Compute Clusters. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2592–2600, 2016. 1

[8] Peter Kairouz, H. Brendan McMahan, et al. Advances and Open Problems in Federated Learning. *arXiv*, dec 2019. 1, 2

[9] Sai Praneeth Reddy Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Jakkam Reddi, Sebastian Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic Controlled Averaging for Federated Learning. In *International Conference on Machine Learning (ICML)*, pages 5132–5143, 2020. 1, 2, 3, 5

[10] Ahmed Khaled, Konstantin Mishchenko, and Peter. Tighter theory for local sgd on identical and heterogeneous data. In *International Conference on Artificial Intelligence and Statistics*, pages 4519–4529, 2020. 1, 2

[11] Jakub Konečný, H. Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated Optimization: Distributed Machine Learning for On-Device Intelligence. *arXiv e-prints*, page arXiv:1610.02527, Oct. 2016. 2

[12] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5

[13] Kumar Kshitij Patel and Aymeric Dieuleveut. Communication trade-offs for synchronized distributed SGD with large step size. *arXiv e-prints*, page arXiv:1904.11325, Apr. 2019. 2

[14] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 5

[15] Li Li, Liang Gao, Huazhu Fu, Bo Han, Cheng-Zhong Xu, and Ling Shao. Federated Noisy Client Learning. *arXiv*, jun 2021. 1

[16] Mu Li, David G Andersen, Alex J Smola, and Kai Yu. Communication efficient distributed machine learning with the parameter server. In *Advances in Neural Information Processing Systems 27*, volume 27, pages 19–27, 2014. 1

[17] 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. 1, 2, 3, 5

[18] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019. 1, 2

[19] 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*, pages 1273–1282. PMLR, 2017. 1, 2, 5, 7

[20] Viraaji Mothukuri, Reza M. Parizi, Seyedamin Pouriyeh, Yan Huang, Ali Dehghantanha, and Gautam Srivastava. A survey on security and privacy of federated learning. *Future Generation Computer Systems*, 115:619–640, 2021. 1

[21] Hadi Pouransari and Saman Ghili. Tiny imagenet visual recognition challenge. 2014. 5

[22] Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konecný, Sanjiv Kumar, and H. Brendan McMahan. Adaptive federated optimization. *CoRR*, abs/2003.00295, 2020. 2

[23] 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, pages 7611–7623, 2020. 3

[24] Shusen Wang, Fred Roosta, Peng Xu, and Michael W. Mahoney. GIANT: Globally Improved Approximate Newton Method for Distributed Optimization. In *Advances in Neural Information Processing Systems*, pages 2332–2342, 2018. 1

[25] Blake E. Woodworth, Jialei Wang, Adam D. Smith, Brendan McMahan, and Nati Srebro. Graph oracle models, lower bounds, and gaps for parallel stochastic optimization. In *Advances in Neural Information Processing Systems*, volume 31, pages 8496–8506, 2018. 2

[26] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017. 5

[27] 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*, 2021. 2

[28] Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan H. Greenewald, Trong Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. In *International Conference on Machine Learning*, pages 7252–7261, 2019. 5

[29] Michael Zhang, Karan Sapra, Sanja Fidler, Serena Yeung, and Jose M. Alvarez. Personalized federated learning with

first order model optimization. In *International Conference on Learning Representations (ICLR)*, 2021. 2

[30] 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. 1, 2

[31] Hankz Hankui Zhuo, Wenfeng Feng, Qian Xu, Qiang Yang, and Yufeng Lin. Federated reinforcement learning. *CoRR*, abs/1901.08277, 2019. 1