FedDNA: Federated Learning with Decoupled Normalization-Layer Aggregation for Non-IID Data

Jian-Hui Duan, Wenzhong Li[0000−0002−9199−3655] ✉, and Sanglu Lu

Nanjing University, Nanjing, China
djhbarca@163.com, {lwz,sanglu}@nju.edu.cn

Abstract. In the federated learning paradigm, multiple mobile clients train their local models independently based on the datasets generated by edge devices, and the server aggregates the model parameters received from multiple clients to form a global model. Conventional methods aggregate gradient parameters and statistical parameters without distinction, which leads to large aggregation bias due to cross-model distribution covariate shift (CDCS), and results in severe performance drop for federated learning under non-IID data. In this paper, we propose a novel decoupled parameter aggregation method called FedDNA to deal with the performance issues caused by CDCS. With the proposed method, the gradient parameters are aggregated using the conventional federated averaging method, and the statistical parameters are aggregated with an importance weighting method to reduce the divergence between the local models and the central model to optimize collaboratively by an adversarial learning algorithm based on variational autoencoder (VAE). Extensive experiments based on various federated learning scenarios with four open datasets show that FedDNA achieves significant performance improvement compared to the state-of-the-art methods.

Keywords: Federated learning · Deep learning · Machine learning.

1 Introduction

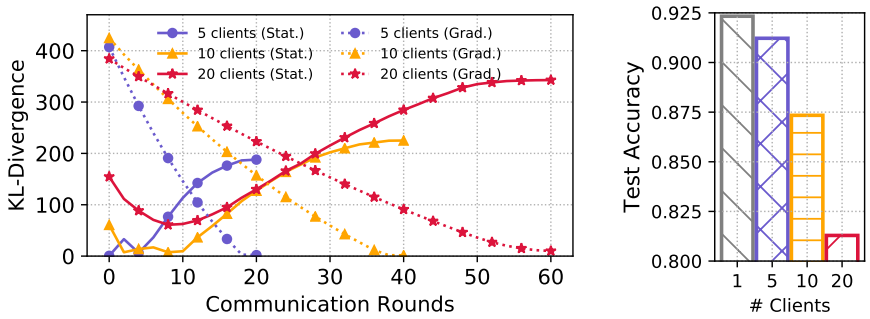
Federated learning (FL) has emerged as a novel distributed machine learning paradigm that allows a global machine learning model to be trained by multiple mobile clients collaboratively while protecting their private data in local devices. In such a paradigm, mobile clients train local models based on datasets generated by edge devices such as sensors and smartphones, and the server is responsible to aggregate parameters from local models to form a global model without transferring data to a central server. Federated learning has been drawn much attention in mobile-edge computing with its advantages in preserving data privacy [35,9] and enhancing communication efficiency [22,28].

Parameter aggregation is the key technology of federated learning, which typically involves the following three steps repeated periodically during the

training process: (1) the involved clients train the same type of models with their local data independently; (2) when the server sends an aggregation signal to the clients, the clients transmit their model parameters to the server; (3) after receiving the local models' parameters, the server applies an aggregation method to the received parameters to form a global model, and broadcast the global model's parameters to the involved clients for the next round of federated training. The standard aggregation method FedAvg [22] and its variants such as q-FedSGD [19] applied a synchronous parameter averaging method to form the global model. Several efforts had been made to deal with non-IID data in federated learning. Zhao et al. proposed to use a globally shared dataset for training to address data heterogeneity [34]. FedProx [18] modified FedAvg by adding a heterogeneity bound on local datasets to tackle the non-IID condition. FedMA [28] demonstrated that permutations of layers could affect the parameter aggregation results and proposed a layer-wise parameter-permutation aggregation method to improve the accuracy of the global model.

Despite the efforts that have been made, applying the existing parameter aggregation methods for a large number of heterogeneous clients in federated learning suffers from performance degrade. It was reported in [34] that the accuracy of a convolutional neural network (CNN) model trained by FedAvg reduced by up to 55% for a highly skewed heterogeneous dataset. The work of [28] showed that the accuracy of FedProx [18] dropped over 11% when the client number increases from 5 to 20 under non-IID data partition. A key issue to cause the performance drop in federated learning could be the covariate shift of data distribution among clients due to non-IID data, which is known as *cross-model distribution covariate shift (CDCS)*. Such issue has not been addressed appropriately by the previous parameter aggregation methods. We use the following example to illustrate the impact of CDCS in federated learning.

Aggregation bias due to CDCS: A CNN model typically consists of the *convolutional (Conv) layers*, the *full connected (FC) layers*, and the *normalization layers*. Those layers are formulated by two different types of parameters to be trained: (1) the *gradient parameters* that represent the weights of the CNN model, which are commonly contained in all layers; (2) the *statistical parameters* that represent the statistical information such as mean and variance of the feature maps, which are solely contained in the normalization layers (e.g., Batch-Normalization and Layer-Normalization). Conventional federated learning approaches such as FedAvg simply average the local model parameters indiscriminately to form a global model, which will lead to bias on the statistical information for non-IID data. Figure 1(a) illustrates the KL-divergence [10] between the parameters of a FedAvg-aggregated model and that of a centrally-trained model with varying number of clients on non-IID data. It is shown that with the increasing of communication rounds, the divergence of gradient parameters approaches to 0, but that of statistical parameters increases to a large value. The more heterogeneous clients involved, the higher divergence is observed. The divergence is mainly caused by the model aggregation methods such as FedAvg that fail to address the CDCS of non-IID local datasets. As a



(a) (b)

Fig. 1. Illustration of divergence and performance drop caused by CDCS on non-IID data. (a) The KL-divergence of gradient parameters and statistical parameters between a federated learning model and a central model of ResNet18@CIFAR-10. (b) The test accuracy of ResNet18 on CIFAR-10 with different #clients.

result, the aggregated models' test accuracy decreases dramatically, as shown in Figure 1(b).

In this paper, we propose a novel decoupled model aggregation method called FedDNA (shorten for federated learning with decoupled normalization-layer parameter aggregation) to address the performance issues caused by CDCS on non-IID data for federated learning. FedDNA aggregates gradient parameters and statistical parameters in a decoupled way. The gradient parameters are aggregated using the conventional distributed stochastic gradient descent (SGD) method, which is theoretically converged during distributed training. The statistical parameters are aggregated with an importance weighting method to reduce the divergence between the local models and the central model, and they are optimized collaboratively by an adversarial learning algorithm based on variational autoencoder (VAE). Extensive experiments based on a variety of federated learning scenarios with four open datasets show that FedDNA significantly outperforms the state-of-the-art methods.

The contributions of our work are summarized as follows.

- We illustrate that cross-model distribution covariate shift (CDCS) can cause large divergence in the aggregated statistical parameters of multiple heterogeneous local models, which is a key problem of performance drop for federated learning under non-IID data.
- We propose a novel decoupled model aggregation method to deal with the performance issues caused by CDCS, where the gradient parameters and statistical parameters are aggregated separately, aiming to reduce the divergence between the local models and the central model.
- We propose an adversarial learning algorithm to derive the optimal probabilistic weights for statistical parameters aggregation. It enables a data-free solution for the federated server with a variational autoencoder

(VAE) to minimize the divergence of unknown distributions based on limited information received from the clients.
– We conduct extensive experiments using five mainstream CNN models based on three federated datasets under non-IID conditions. Compared to the de facto standard FedAvg, and the state-of-the-art for non-IID data (FedProx, FedMA), the proposed FedDNA has the lowest divergence of the aggregated parameters, and the test accuracy improves up to 9%.

The rest of the paper is organized as follows: Section-2 presents the related works on federated learning and optimizing federated learning under non-IID data. Section-3 proposes the detailed decoupled mechanism of `FedDNA`. Section-4 describes the adversarial learning algorithm on optimizing FedDNA, inference and optimization on VAE and the detailed algorithm of `FedDNA`. Section-5 shows the performance evaluation for `FedDNA` with 5 state-of-the-art baselines. And the paper is concluded in section-6.

2 Related Work

Federated learning [14,31,21,26,24,32] is an emerging distributed machine learning paradigm that aims to build a global model based on datasets distributing across multiple clients. One of the standard parameter aggregation methods is FedAvg [22], which combined local stochastic gradient descent (SGD) on each client with a server that performs parameter averaging. Later, the lazily aggregated gradient (Lag) method [2] allowed clients running multiple epochs before model aggregation to reduce communication cost. The q-FedSGD [19] method improved FedAvg with a dynamic SGD update step using a scale factor to achieve fair resources allocation among heterogeneous clients. The FedDyn [1] method proposed a dynamic regularizer for each round of aggregation, so that different models are aligned to alleviate the inconsistency between local and global loss.

Several works focused on optimizing federated learning under non-IID data. Zhao et al. used the earth mover's distance (EMD) to quantify data heterogeneity and proposed to use globally shared data for training to deal with non-IID [34]. FedProx [18] modified FedAvg by adding a heterogeneity bound on local datasets to tackle heterogeneity. The RNN-based method in [8] adopted a meta-learning method to learn a new gradient from the received gradients and then applied it to update the global model. The FedMA [28] method, derived from AFL [23] and PFNM [33], demonstrated that permutations of layers can affect the parameter aggregation results, and proposed a layer-wise parameter-permutation aggregation method to solve the problem. FedBN [20] suggested keeping the local Batch Normalization parameters not synchronized with the global model to mitigate feature shifts in Non-IID data. FedGN [5] replaced Batch Normalization with Group Normalization to avoids the accuracy loss induced by the skewed distribution of data labels. SCAFFOLD [11] used variance reduction to correct the "client-drift" in each client's local updates to prevent unstable and slow convergence for heterogeneous data. FedRobust [25] adopted

a distributed optimization method via gradient descent ascent to address affine distribution shifts across users in federated settings. Instead of averaging the cumulative local gradient, FedNova [29] aggregated normalized local gradients to eliminate objective inconsistency while preserving fast error convergence.

To the best of our knowledge, the problem of performance drop of federated learning caused by CDCS has not been explored in the literature. In this paper, we make the first attempt to deal with CDCS by decoupling the aggregation of gradient parameters and statistical parameters, and adopt an adversarial learning algorithm to optimize model aggregation to achieve high accuracy in non-IID conditions.

3 Decoupled Federated Learning with CDCS

3.1 Optimization Objectives

As discussed in section-1, conventional federated learning methods suffers from noteworthy bias when aggregating statistical parameters and gradient parameters without distinction. To address this issue, we propose a decoupled method to optimize model parameters aggregation.

Consider a federated learning scenario with K clients that train their local deep neural network (DNN) models independently based on local datasets $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_K$, and report their model parameters to a central server. The objective of the server is to form an aggregate global DNN model to minimize the following loss function.

$$\min_{\mathbf{w}} \mathcal{L}(\mathbf{w}, \mathbf{x}) := \sum_{k=1}^{K} \frac{|\mathbf{x}_k|}{|\mathbf{x}|} \mathcal{L}_k(\hat{\mathbf{w}}_k, \tilde{\mathbf{w}}_k, \mathbf{x}_k), \tag{1}$$

where $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_K\}$ is the total dataset; $\hat{\mathbf{w}}_k, \tilde{\mathbf{w}}_k$ are the gradient parameters and statistical parameters of local model received from the k-th client; $\mathcal{L}_k(\cdot)$ indicates the loss functions of the k-th local model; $\mathbf{w} = F(\hat{\mathbf{w}}_1, \tilde{\mathbf{w}}_1, \hat{\mathbf{w}}_2, \tilde{\mathbf{w}}_2, \cdots, \hat{\mathbf{w}}_K, \tilde{\mathbf{w}}_K)$ are the global model's parameters and F is the aggregation method to be derived that maps K local models to a global model.

To deal with the CDCS problem, we aggregate the gradient parameters and statistical parameters separatively. Different from the gradient parameters, which can be optimized with the conventional distributed stochastic gradient descent (SGD), the statistical parameters should be optimized to eliminate their covariate shifts. Since the real distribution of global data is unknown to the server, we propose a *collaborative optimization* approach that enables multiple clients to update their statistical parameters to gradually reduce their distribution divergence.

For the k-th client, we use $\tilde{\mathbf{w}}_k$ to denote its own statistical parameters, and $\tilde{\mathbf{w}}_{\neg k}$ to denote the averaged statistical parameters of the other clients except client-k. We refer to $\tilde{\mathbf{w}}_k$ and $\tilde{\mathbf{w}}_{\neg k}$ as the *twin statistics*. We assume that $\tilde{\mathbf{w}}_k$ is drawn from client-k's local distribution $\tilde{\mathbf{w}}_k \sim p_k(\tilde{\mathbf{w}})$, and $\tilde{\mathbf{w}}_{\neg k}$ is drawn from

the distribution without client-k, i.e., $\tilde{\mathbf{w}}_{\neg k} \sim q_{\neg k}(\tilde{\mathbf{w}})$, where $\tilde{\mathbf{w}}$ represent the statistical parameters of the global model. Aggregation of statistical parameters should eliminate the discrepancy between $\tilde{\mathbf{w}}_k$ and $\tilde{\mathbf{w}}_{\neg k}$ so that they converge to the same objective distribution, which can be represented by the following optimization problem:

$$\min_{\tilde{\mathbf{w}}} \tilde{\mathcal{L}}(\tilde{\mathbf{w}}) := \frac{1}{K} \sum_{k=1}^{K} \mathbb{D}[p_k(\tilde{\mathbf{w}}), q_{\neg k}(\tilde{\mathbf{w}})], \tag{2}$$

where $\mathbb{D}[\cdot]$ represents the divergence of two distributions.

3.2 `FedDNA` Mechanism

Based on the optimization objectives, we proposed a decoupled method called `FedDNA` to optimize parameters aggregation for federated learning. The federated server received model parameters from the clients periodically. In the t-th communication round, the received parameters are:

- $\hat{\mathbf{w}}_k^t$: the gradient parameters of client-k in round t;
- $\tilde{\mathbf{w}}_k^t = [\tilde{\mathbf{mean}}_k^t, \tilde{\mathbf{var}}_k^t]$: the statical parameters of client-k in round t, where $\tilde{\mathbf{mean}}_k^t$ and $\tilde{\mathbf{var}}_k^t$ represent the parameters of statical mean and variance accordingly.

The parameter update process is as follows.

(1) Aggregation of Gradient Parameters: The gradient parameters can be aggregated using the principle of distributed stochastic gradient descent (SGD), the same as the method of FedAvg [22]:

$$\hat{\mathbf{w}}^{t+1} = \sum_{k=1}^{K} \frac{|\mathbf{x}_k|}{|\mathbf{x}|} \hat{\mathbf{w}}_k^t. \tag{3}$$

(2) Aggregation of Statistical Parameters: The statistical parameters are aggregated collaboratively to reduce the divergence between the individual's distribution and the overall distribution, which are updated with the following reweighting manner:

$$\begin{aligned} \tilde{\mathbf{mean}}_k^{t+1} &= \gamma_k^t \tilde{\mathbf{mean}}_k^t + (1-\gamma_k^t) \sum_{i=1, i\neq k}^{K} \frac{1}{K-1} \tilde{\mathbf{mean}}_i^t, \\ \tilde{\mathbf{var}}_k^{t+1} &= \gamma_k^t \tilde{\mathbf{var}}_k^t + (1-\gamma_k^t) \sum_{i=1, i\neq k}^{K} \frac{|\mathbf{x}_i|-1}{|\mathbf{x}|-K-1} \tilde{\mathbf{var}}_i^t. \end{aligned} \tag{4}$$

In the above equations, the term $\sum_{i=1,i\neq k}^{K} \frac{1}{K-1} \tilde{\mathbf{mean}}_i^t$ is the average of other statistical means, and the term $\sum_{i=1,i\neq k}^{K} \frac{|\mathbf{x}_i|-1}{|\mathbf{x}|-K-1} \mathbf{var}_i^t$ is the weighted pooled variance [12] which gives an unbias estimation of the variance of the overall

dataset without client-k. The adjustable parameter γ_k^t is called the *importance weight*, which tends to reduce the discrepancy between the twin statistics $\tilde{\mathbf{w}}_k$ and $\tilde{\mathbf{w}}_{\neg k}$. The weight γ_k^t can be formulated by

$$\gamma_k^t = f(\mathbb{D}[p_k(\tilde{\mathbf{w}}^t), q_{\neg k}(\tilde{\mathbf{w}}^t)]), \tag{5}$$

where f: $\mathbb{R}^+ \to [0, 1)$ is a function that projects the divergence into the interval $[0, 1)$. Generally speaking, the mapping f could be non-linear, and it can be implemented by a neural network in practice.

The proposed method minimize the loss function in Eq. (2) by gradually updating the statistical parameters to approach the overall distribution with an importance weight. By minimizing Eq. (2), the importance weight γ_k^t approaches to 0, and the statistical parameters of the ith client converges to that of the other clients. Thereafter, the final statistical parameters of the global model can be represented by $\tilde{\mathbf{w}}_k = [\frac{1}{K}\sum_{k=1}^K \tilde{\mathbf{mean}}_k^T, \sum_{k=1}^K \frac{(|\mathbf{x}_k|-1)}{|\mathbf{x}|-K}\tilde{\mathbf{var}}_k^T]$ where T is the total communication rounds.

To determine the value of importance weight γ_k^t, it needs to derive the divergence $\mathbb{D}$ and the mapping function f, which can be solved by the adversarial learning algorithm discussed in the following section.

4 Adversarial Learning Algorithm

`FedDNA` aggregates the statistical parameters with an importance weight γ_k^t intending to eliminate the discrepancy between the twin statistics $\tilde{\mathbf{w}}_k$ and $\tilde{\mathbf{w}}_{\neg k}$. The updating process can be viewed as sampling from $p_k(\tilde{\mathbf{w}})$ to match $q_{\neg k}(\tilde{\mathbf{w}})$ with acceptance rate $\alpha(\tilde{\mathbf{w}}) \in [0, 1]$. Here $\alpha(\tilde{\mathbf{w}})$ can be interpreted as a binary classifier between "from $p_k(\tilde{\mathbf{w}})$" and "from $q_{\neg k}(\tilde{\mathbf{w}})$".

More formally, we let $y = 1$ to represent "from $p_k(\tilde{\mathbf{w}})$" and $y = 0$ to represent "from $q_{\neg k}(\tilde{\mathbf{w}})$". The overall dataset can be represent by a a joint distribution $p(\tilde{\mathbf{w}}, y)$, which can be written as $p(\tilde{\mathbf{w}}, y) = p(\tilde{\mathbf{w}}|y = 1)p(y = 1) + p(\tilde{\mathbf{w}}|y = 0)p(y = 1)$. So the classifier can be represented by $\alpha(\tilde{\mathbf{w}}) = p(y = 1|\tilde{\mathbf{w}}_k)$. By Bayes' theorem, we have:

$$\begin{aligned}\alpha(\tilde{\mathbf{w}}) &= \frac{p(\tilde{\mathbf{w}}|y=1)p(y=1)}{p(\tilde{\mathbf{w}}, y)} = \frac{p_k(\tilde{\mathbf{w}})p(y=1)}{p(\tilde{\mathbf{w}}|y=1)p(y=1) + p(\tilde{\mathbf{w}}|y=0)p(y=0)} \\ &= \frac{p_k(\tilde{\mathbf{w}})p(y=1)}{p_k(\tilde{\mathbf{w}})p(y=1) + q_{\neg k}(\tilde{\mathbf{w}})p(y=0)} \propto \frac{p_k(\tilde{\mathbf{w}})}{p_k(\tilde{\mathbf{w}}) + q_{\neg k}(\tilde{\mathbf{w}})}.\end{aligned} \tag{6}$$

Therefore, the $\alpha(\tilde{\mathbf{w}})$ is only related with $p_k(\tilde{\mathbf{w}})$ and $q_{\neg k}(\tilde{\mathbf{w}})$.

By introducing the classifier α, the derivation of the importance weight γ_k^t can be described as an adversarial learning framework, which is illustrated in Figure 2(a). The classifier α works as an "adversarial discriminator" to distinguish $p_k(\tilde{\mathbf{w}})$ and $q_{\neg k}(\tilde{\mathbf{w}})$ as possible, which forms a *divergence measure* of the twin statistics. Based on the divergence from α, the project function f generates an importance weight γ_k^t, which is used to adjust the statistical parameters with $\tilde{\mathbf{w}}_k$ and $\tilde{\mathbf{w}}_{\neg k}$ to reduce their discrepancy. The intuition is that

the twin statistics are more likely to converge to the objective distribution if they are harder to be differentiated by a classifier.

4.1 Adversarial Training

Based on the adversarial learning framework with f and α, the training object to optimize Eq. (2) can be written as $\min_f \frac{1}{K}\sum_{k=1}^{K}\mathbb{D}(p_k(\tilde{\mathbf{w}}, f, \alpha), q_{\neg k}(\tilde{\mathbf{w}}, f, \alpha))$, where the divergence measure can be expressed by:

$$\mathbb{D}(p_k(\tilde{\mathbf{w}}, f, \alpha), q_{\neg k}(\tilde{\mathbf{w}}, f, \alpha)) = \max_{\alpha} \mathbb{E}_{\tilde{\mathbf{w}}\sim p_k}[\log \alpha(\tilde{\mathbf{w}})] + \mathbb{E}_{\tilde{\mathbf{w}}\sim q_{\neg k}}[\log(1-\alpha(\tilde{\mathbf{w}}))]. \tag{7}$$

Therefore f and α can be derived by solving the following min-max optimization:

$$\min_{f}\max_{\alpha}\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}_{\tilde{\mathbf{w}}\sim p_k}[\log \alpha(\tilde{\mathbf{w}})] + \mathbb{E}_{\tilde{\mathbf{w}}\sim q_{\neg k}}[\log(1-\alpha(\tilde{\mathbf{w}}))]. \tag{8}$$

To form a learning model to solve the min-max problem, $f(\cdot)$ can be parameterized by a fully-connected neural network, and $\alpha(\cdot)$ can be parameterized by a variational auto-encoder (VAE) as illustrated in Figure 2(a), where the latent variable $\mathbf{z}$ represents the divergence between $p_k(\tilde{\mathbf{w}})$ and $q_{\neg k}(\tilde{\mathbf{w}})$ in priori distribution. Next we inference $\alpha(\cdot)$ with VAE.

4.2 Inference with VAE

We construct a variational encoder-decoder model that takes the observed statistical parameters as input, encodes them to a latent variable, and decodes the statistical information from the latent variable to minimize the reconstruction error. The plate notions of the generative model are shown in Figure 2(b), which are explained as follows.

- $\tilde{\mathbf{w}}_k^t$ is the observed statistical parameters from the local model of client-k. In communication round t, the server receives a set statistical parameters $\tilde{\mathbf{w}}^t = \{\tilde{\mathbf{w}}_1^t, \ldots, \tilde{\mathbf{w}}_K^t\}$, and they are used to train the VAE to generate the latent representation of the twin statistics.
- $\mathbf{z}_k^t$ is a latent variable whose prior is the joint distribution of $p(\tilde{\mathbf{w}}, \mathbf{y}) \sim Gaussian(\boldsymbol{\mu}, \boldsymbol{\sigma})$, where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are inferred parameters that are used to generate $\mathbf{z}_k^t$.
- $\boldsymbol{\theta}$ are the generative model parameters (decoder), and $\boldsymbol{\phi}$ are the variational parameters (encoder).

The solid lines in Figure 2(b) denote the generative process $p_{\boldsymbol{\theta}}(\mathbf{z}_k^t)p_{\boldsymbol{\theta}}(\tilde{\mathbf{w}}_k^t|\mathbf{z}_k^t)$, and the dashed lines denote the variational approximation $q_{\boldsymbol{\phi}}(\mathbf{z}_k^t|\tilde{\mathbf{w}}_k^t)$ to the intractable posterior $p_{\boldsymbol{\theta}}(\mathbf{z}_k^t|\tilde{\mathbf{w}}_k^t)$. We approximate $p_{\boldsymbol{\theta}}(\mathbf{z}_k^t|\tilde{\mathbf{w}}_k^t)$ with $q_{\phi}(\mathbf{z}_k^t|\tilde{\mathbf{w}}_k^t)$ by minimizing their divergence:

$$\boldsymbol{\phi}^*, \boldsymbol{\theta}^* = arg\min_{\boldsymbol{\theta},\boldsymbol{\phi}} \mathbb{D}(q_{\boldsymbol{\phi}_k}(\mathbf{z}_k^t|\tilde{\mathbf{w}}_k^t) \,||\, p_{\boldsymbol{\theta}_k}(\mathbf{z}_k^t|\tilde{\mathbf{w}}_k^t)). \tag{9}$$

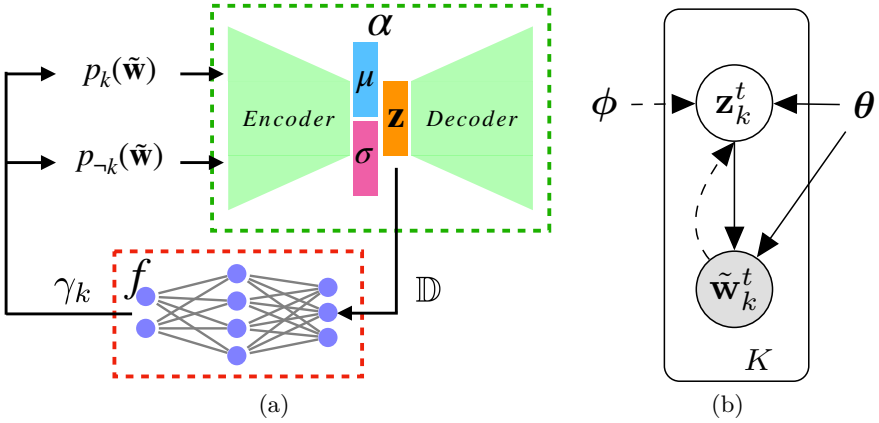


Fig. 2. Illustration of adversarial learning. (a) The framework. (b) The plate notations of VAE.

To derive the optimal value of the parameters $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$, we compute the marginal likelihood of $\tilde{\mathbf{w}}_k^t$:

$$\log p(\tilde{\mathbf{w}}_k^t) = \mathbb{D}_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}_k^t|\tilde{\mathbf{w}}_k^t) \,||\, p_{\boldsymbol{\theta}}(\mathbf{z}_k^t|\tilde{\mathbf{w}}_k^t)) + \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}_k^t|\tilde{\mathbf{w}}_k^t)}\left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{z}_k^t, \tilde{\mathbf{w}}_k^t)}{q_{\boldsymbol{\phi}}(\mathbf{z}_k^t|\tilde{\mathbf{w}}_k^t)}\right]. \quad (10)$$

In Eq. (10), the first term is the KL-divergence [10] of the approximate distribution and the posterior distribution; the second term is called the ELBO (Evidence Lower BOund) on the marginal likelihood of dataset in the k-th client.

Since $\log p(\tilde{\mathbf{w}}_k^t)$ is non-negative, the minimization problem of Eq. (9) can be converted to maximize the ELBO. To solve the problem, we change the form of ELBO as:

$$\begin{aligned} &\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}_k^t|\tilde{\mathbf{w}}_k^t)}\left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{z}_k^t, \tilde{\mathbf{w}}_k^t)}{q_{\boldsymbol{\phi}}(\mathbf{z}_k^t|\tilde{\mathbf{w}}_k^t)}\right] = \\ &\underbrace{\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}_k^t|\tilde{\mathbf{w}}_k^t)}\left[log \frac{p(\mathbf{z}_k^t)}{q_{\boldsymbol{\phi}}(\mathbf{z}_k^t|\tilde{\mathbf{w}}_k^t)}\right]}_{\text{Encoder}} + \underbrace{\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}_k^t|\tilde{\mathbf{w}}_k^t)}[\log p_{\boldsymbol{\theta}}(\tilde{\mathbf{w}}_k^t|\mathbf{z}_k^t)]}_{\text{Decoder}}. \end{aligned} \quad (11)$$

The above form is a variational encoder-decoder structure: the model $q_{\boldsymbol{\phi}}(\mathbf{z}_k^t|\tilde{\mathbf{w}}_k^t)$ can be viewed as a probabilistic encoder that given an observed statistics $\tilde{\mathbf{w}}_k^t$ it produces a distribution over the possible values of the latent variables $\mathbf{z}_k^t$; The model $p_{\boldsymbol{\theta}}(\tilde{\mathbf{w}}_k^t|\mathbf{z}_k^t)$ can be refered to as a probabilistic decoder that reconstructs the value of $\tilde{\mathbf{w}}_k^t$ based on the code $\mathbf{z}_k^t$. According to the theory of variational inference [13], the problem in Eq. (11) can be solved with the SGD method using a fully-connected neural network to optimize the mean squared error loss function.

After training the model, we feed the twin statistics $\{\tilde{\mathbf{w}}_k^t, \tilde{\mathbf{w}}_{\neg k}^t\}$ into the VAE and obtain the corresponding latent variables $\{\mathbf{z}_k^t, \mathbf{z}_{\neg k}^t\}$. The latent variables are

Algorithm 1 FedDNA

Initialize $\mathbf{w}^0$ and γ_k.
for each round $t = 0, 1, \ldots, T-1$ **do**
  $S^t :=$ (random set of m clients)
  Send $\mathbf{w}^t$ to client in S^t as $\mathbf{w}_k^t$
  for each client $k \in S^t$ **in parallel do**
    for each local epoch $e = 0, 1, \ldots, E-1$ **do**
      $\mathbf{w}_k^t := \mathbf{w}_k^t$ - $\eta \nabla l(\mathbf{w}_k^t; \mathbf{x}_k)$
    end for
  end for
  Receive $\mathbf{w}_k^t$ from client in S^t
  for each client $k \in S^t$ **do**
    $\tilde{\mathbf{w}}_k^t := [\tilde{\mathbf{mean}}_k^t, \tilde{\mathbf{var}}_k^t]$.
  end for
  Update $\hat{\mathbf{w}}_k^t$ by Eq. (3)
  Compute $\tilde{\mathbf{w}}_{\neg k}^t$ based on Eq. (4)
  repeat
    $\boldsymbol{\mu}, \boldsymbol{\sigma} := \boldsymbol{\phi}(\tilde{\mathbf{w}}_k^t), \boldsymbol{\phi}(\tilde{\mathbf{w}}_{\neg k}^t)$
    Sample $\mathbf{z}_k^t$ and $\mathbf{z}_{\neg k}^t$ from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$
    $\gamma_k^t := f(\mathbb{D}[\mathbf{z}_k, \mathbf{z}_{\neg k}])$
    $(\boldsymbol{\phi}, \boldsymbol{\theta}) := (\boldsymbol{\phi}, \boldsymbol{\theta}) - \eta \nabla l(\boldsymbol{\phi}, \boldsymbol{\theta}; \tilde{\mathbf{w}}_k^t)$
    $f = f - \eta \nabla l(f; \mathbb{D}[\mathbf{z}_k, \mathbf{z}_{\neg k}])$
    Update every client's statistical parameters based on Eq. (4)
  until α and f Converge.
  Form $\mathbf{w}_k^t$ by $\hat{\mathbf{w}}_k^t$ and $\tilde{\mathbf{w}}_k^t$.
end for

further used by the nueral network of f to derive their divergence and output the important weight γ_k^t, which is used to update the statistical parameters in Eq. (4). The pseudo-code of `FedDNA` is shown in Algorithm 1.

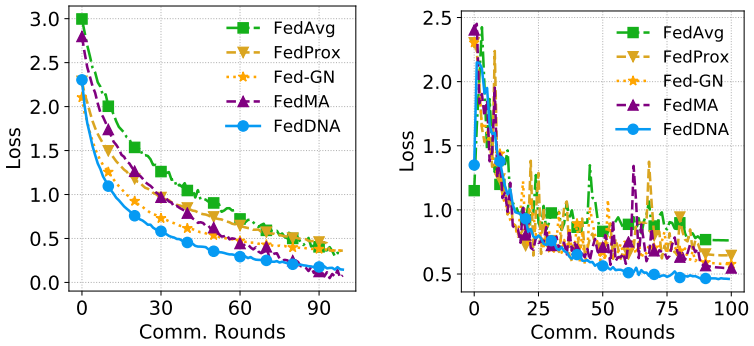
5 Performance Evaluation

5.1 Experimental Setup

Implementation. We implement `FedDNA`[1] and the considered baselines in PyTorch. We train the models in a simulated federated learning environment consisting of one server and a set of clients with wireless network connections. Unless explicitly specified, the default number of clients is 20 as FedMA [28], and the learning rate $\beta = 0.01$. We conduct experiments on a GPU-equipped personal computer (CPU: Intel Core i7-8700 3.2GHz, GPU: Nvidia GeForce RTX 2070, Memory: 32GB DDR4 2666MHz, and OS: 64-bit Ubuntu 16.04).

Models and datasets. We conduct experiments based on 6 mainstream neural network models: ResNet18 [4], LeNet [16], DenseNet121 [6], MobileNetV2 [27] and BiLSTM [7]. The first 5 models used Batch-Normalization (BN), which

[1] The source code will be publicly available after acceptance.



(a) ResNet18 on CIFAR-10 (b) BiLSTM on Sent140

Fig. 3. Convergence of different algorithms.

are commonly applied in computer vision (CV), and the last model used Layer-Normalization (LN), which is applied in natural language processing (NLP).

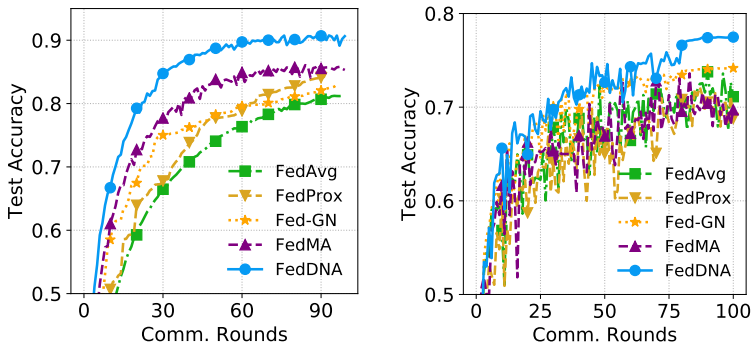
We use 4 real world datasets: MNIST [17], Fashion-MNIST [30], CIFAR-10 [15] and Sentiment140 [3]. MNIST is a dataset for hand written digits classification with 60000 samples of 28×28 greyscale image. Fashion-MNIST is an extended version of MNIST for benchmarking machine learning algorithms. CIFAR-10 is a large image dataset with 10 categories, each of which has 6000 samples of size 32×32. Sentiment140 is a natural language process dataset containing 1,600,000 extracted tweets annotated in scale 0 to 4 for sentiment detection.

We generate non-IID data partition according to the work [22]. For each dataset, we use 80% as training dada to form non-IID local datasets as follows. We sort the data by their labels and divide each class into 200 shards. Each client draw samples from the shards to form a local dataset with probability $pr(x) = \begin{cases} \eta \in [0,1], & \text{if x} \in class_j, \\ \mathcal{N}(0.5,1), & \text{otherwise.} \end{cases}$ It means that the client draws samples from a particular class j with a fixed probability η, and from other classes based on a Gaussian distribution. The larger η is, the more likely the samples concentrate on a particular class, and the more heterogeneous the datasets are.

5.2 Performance Analysis

We compare the performance of `FedDNA` with 5 state-of-the-art methods: FedAvg [22], FedProx [18], Fed-GN [5], SCAFFOLD [11], and FedMA [28] with $\eta = 0.5$. The results are analyzed as follows.

Convergence: In this experiment, we study the convergence of `FedDNA` and all baselines by showing the total communication rounds versus train loss with local training epoch $E = 10$. Figure 3(a) and Figure 3(b) show the result of ResNet18 on CIFAR-10 and BiLSTM on Sent140. It is shown that the loss of all



(a) ResNet18 on CIFAR-10 (b) BiLSTM on Sent140

Fig. 4. Training efficiency of different algorithms.

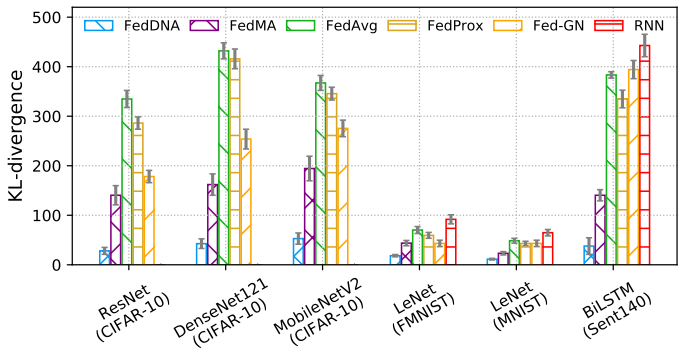


Fig. 5. Comparison of KL-divergence of statistical parameters of different models with different algorithms.

algorithms tends to be stable after a number of epochs. Clearly, `FedDNA` has the lowest loss among all algorithms, and it converges faster than all baselines.

Training Efficiency: In this experiment, we study the test accuracy versus communication rounds during training with local training epoch $E = 10$. Figure 4(a) and Figure 4(b) show the results of training ResNet18 on CIFAR-10 and BiLSTM on Sent140. For ResNet18, it is shown that `FedDNA` reaches 0.8 accuracy after 16 communication rounds, while the others take 30 to 80 communication rounds to reach the same accuracy. `FedDNA` exceeds 0.9 accuracy after 63 communication rounds, while the accuracy of other algorithms is below 0.86. Similar results are found for BiLSTM, where `FedDNA` achieves the highest accuracy at the same communication round. It suggests that `FedDNA` trains much faster than the baseline algorithms, and it can reach higher accuracy with less communication cost.

Table 1. Average test accuracy (%) on datasets with local training epoch $E = 10$. The "Central" method trains the CNN models in the central server with global dataset.

Algorithm	LeNet@MNIST	LeNet@F-MNIST	BiLSTM@Sent140
Central	98.95	90.42	81.47
FedAvg	97.32 (± 0.09)	87.41 (± 0.29)	72.14 (± 0.93)
FedProx	97.55 (± 0.14)	88.33 (± 0.35)	71.08 (± 1.28)
Fed-GN	95.88 (± 0.25)	88.21 (± 0.27)	74.44 (± 1.04)
SCAFFOLD	97.47 (± 0.19)	89.36 (± 0.21)	73.83 (± 0.79)
FedMA	97.86 (± 0.18)	89.02 (± 0.46)	72.81 (± 1.38)
`FedDNA`	**98.49** (± 0.12)	**90.11** (± 0.18)	**77.51** (± 0.87)
Algorithm	ResNet18@CIFAR-10	DenseNet121@CIFAR-10	MobileNetV2@CIFAR-10
Central	92.33	93.24	92.51
FedAvg	81.29 (± 0.83)	81.86 (± 0.46)	80.11 (± 0.87)
FedProx	83.47 (± 0.68)	85.03 (± 0.65)	80.68 (± 0.79)
Fed-GN	83.08 (± 0.63)	83.43 (± 0.53)	82.82 (± 0.61)
SCAFFOLD	85.72 (± 0.45)	86.94 (± 0.39)	85.27 (± 0.85)
FedMA	86.44 (± 0.59)	87.12 (± 0.71)	85.59 (± 1.07)
`FedDNA`	**90.31** (± 0.45)	**90.29** (± 0.42)	**89.17** (± 0.72)

Divergence: We compare the KL-divergence of statistical parameters between the global model aggregated by different algorithms and the central model, which are shown in Figure 5. It is shown that FedMA, FedProx, Fed-GN, and FedAvg have exceptional high divergence varying from 140.4 to 447.1 on the CIFAR-10 dataset, while `FedDNA` has significantly lower divergences than all baselines for all models and datasets. It suggests that the statistical parameters aggregated by `FedDNA` are much more close to the central model in non-IID settings.

Global Model Accuracy: In this experiment, we compare the global model accuracy of different federated parameter aggregation algorithms after training to converge. We repeat the experiment for 20 rounds and show the average results in Table 1. As shown in the table, the central method yields the highest accuracy. In the comparison of different federated learning methods, `FedDNA` significantly outperforms the other algorithms in global model accuracy. It performs better than the state-of-the-art method FedMA with 3.87%, 3.17%, 3.58%, and 4.09% accuracy improvement on ResNet18, DenseNet121, MobileNetV2, and 4-L CNN respectively for CIFAR-10; 1.09% and 0.63% improvement on LeNet for F-MNIST and MNIST; 4.70% improvement on BiLSTM for Sent140. Compared to Fed-GN, `FedDNA` achieves accuracy improvement with 7.32%, 6.86%, 6.35%, and 3.08% on ResNet18, DenseNet121, MobileNetV2, and 4-L CNN respectively for CIFAR-10; 1.90% and 2.61% on LeNet for F-MNIST and MNIST; 3.07% on BiLSTM for Sent140 accordingly. Compared to FedAvg, `FedDNA` improves the test accuracy up to 9.02% for ResNet18 on CIFAR-10. In summary, `FedDNA` achieves the highest accuracy among all baselines, and it performs very close to the centralized method, whose accuracy drop is within 4% in all cases.

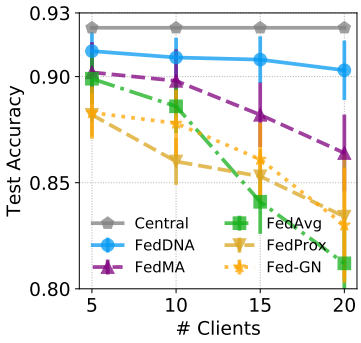


Fig. 6. Test accuracy with different number of clients (ResNet18 on CIFAR-10).

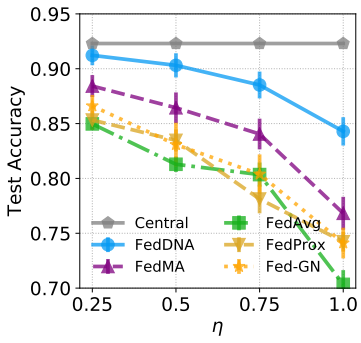


Fig. 7. Test accuracy on different level of heterogeneity (ResNet18 on CIFAR-10).

Hyperparameter Analysis: We further analyze the influence of two hyperparameters in federated learning: the number of clients and the heterogeneity of local datasets.

Figure 6 compares the test accuracy of the global model for a different number of involved clients. According to the figure, the performance of `FedDNA` remains stable. When the number of clients increases from 5 to 20, the test accuracy slightly decreases from 0.912 to 0.903. The other algorithms yield significant performance drop, and the accuracy of most baselines is below 0.85 for 20 clients. `FedDNA` achieves the highest test accuracy among all federated learning algorithms in all cases, and it performs very close to the central model.

In the experiment, the heterogeneity of local datasets is represented by η, the probability that a client tends to sample from a particular class. The more η approaches to 1, the more heterogeneous the local datasets are. Figure 7 shows the test accuracy under different levels of heterogeneity. As η increases, the test accuracy of all models decreases. `FedDNA` yields the highest test accuracy among all algorithms, and its performance drops much slower than that of the baselines. It verifies the effectiveness of the proposed decoupled aggregation approach under non-IID conditions.

6 Conclusion

Parameter aggregation played an important role in federated learning to form a global model. To address the problem of aggregation bias in federated learning for non-IID data, we proposed a novel parameter aggregation method called FedDNA that decoupled gradient parameters and statistical parameters to aggregate them separately with stochastic gradient descent and importance weighting method to reduce the divergence between the local models and the central model. FedDNA optimized parameter aggregation by

an adversarial learning algorithm based on variational autoencoder (VAE). Extensive experiments showed that `FedDNA` significantly outperforms the state-of-the-arts on a variety of federated learning scenarios.

Acknowledgment

This work was partially supported by the National Key R&D Program of China (Grant No. 2018YFB1004704), the National Natural Science Foundation of China (Grant Nos. 61972196, 61832008, 61832005), the Key R&D Program of Jiangsu Province, China (Grant No. BE2018116), the Collaborative Innovation Center of Novel Software Technology and Industrialization, and the Sino-German Institutes of Social Computing. The corresponding author is Wenzhong Li.

References

1. Acar, D.A.E., Zhao, Y., Matas, R., Mattina, M., Whatmough, P., Saligrama, V.: Federated learning based on dynamic regularization. In: Proceedings of ICLR. (2021)
2. Chen, T., Giannakis, G., Sun, T., Yin, W.: Lag: Lazily aggregated gradient for communication-efficient distributed learning. In: Proceedings of NIPS. pp. 5050–5060 (2018)
3. Go, A., Bhayani, R., Huang, L.: Twitter sentiment classification using distant supervision (2009), http://help.sentiment140.com/home
4. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of CVPR. pp. 770–778 (2016)
5. Hsieh,K., Phanishayee, A.,Mutlu,O., Gibbons,P.B.: The non-iid data quagmire of decentralized machine learning. In: Proceedings of ICML. (2020)
6. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of CVPR. pp. 2261–2269 (2017)
7. Huang, Z., Xu, W., Yu, K.: Bidirectional lstm-crf models for sequence tagging (2015), http://arxiv.org/abs/1508.01991, cite arxiv:1508.01991
8. Ji, J., Chen, X., Wang, Q., Yu, L., Li, P.: Learning to learn gradient aggregation by gradient descent. In: Proceedings of IJCAI. pp. 2614–2620 (2019)
9. Jiang,L., Tan,R., Lou,X., Lin,G.: On lightweight privacy-preserving collaborative learning for internet-of-things objects. In: Proceedings of IoTDI. pp. 70– 81 (2019)
10. Joyce, J.M.: Kullback-Leibler Divergence, pp. 720–722 (2011)
11. Karimireddy, S.P., Kale, S., Mohri, M., Reddi, S., Stich, S., Suresh, A.T.: SCAFFOLD: Stochastic controlled averaging for federated learning. In: Proceedings of ICML. (2020)
12. Killeen, P.R.: An alternative to null-hypothesis significance tests. Psychological science 16(5), 345–53 (2005)
13. Kingma, D., Welling, M.: Auto-encoding variational bayes. In: Proceedings of ICLR. (2014)

14. Konečnỳ, J., McMahan, H.B., Ramage, D.: Federated optimization: Distributed optimization beyond the datacenter. NIPS Optimization for Machine Learning Workshop 2015 p. pp.5 (2015)
15. Krizhevsky, A.: Learning multiple layers of features from tiny images. Tech. rep. (2009)
16. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE. pp. 2278–2324 (1998)
17. LeCun, Y., Cortes, C., Burges, C.: Mnist handwritten digit database. ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist (2010)
18. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. In: Proceedings of MLSys. pp. 429–450 (2020)
19. Li, T., Sanjabi, M., Smith, V.: Fair resource allocation in federated learning. In: Proceedings of ICLR. (2020)
20. Li, X., JIANG, M., Zhang, X., Kamp, M., Dou, Q.: FedBN: Federated learning on non-IID features via local batch normalization. In: Proceedings of ICLR. (2021)
21. Malinovsky, G., Kovalev, D., Gasanov, E., Condat, L., Richtárik, P.: From local sgd to local fixed-point methods for federated learning. In: Proceedings of ICML. (2020)
22. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Proceedings of AISTATS. 54, 1273–1282 (2017)
23. Mohri, M., Sivek, G., Suresh, A.T.: Agnostic federated learning. ICML. (2019)
24. Pathak, R., Wainwright, M.J.: Fedsplit: an algorithmic framework for fast federated optimization. In: Proceedings of NeurIPS. vol. 33 (2020)
25. Reisizadeh, A., Farnia,F., Pedarsani, R., Jadbabaie, A.: Robust federated learning: The case of affine distribution shifts. In: Proceedings of NeurIPS. (2020)
26. Rothchild, D., Panda, A., Ullah, E., Ivkin, N., Stoica, I., Braverman, V., Gonzalez, J., Arora, R.: Fetchsgd: Communication-efficient federated learning with sketching. In: Proceedings of ICML. (2020)
27. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of CVPR. pp. 4510–4520 (2018)
28. Wang, H., Yurochkin, M., Sun, Y., Papailiopoulos, D., Khazaeni, Y.: Federated learning with matched averaging. In: Proceedings of ICLR. (2020)
29. Wang, J., Liu, Q., Liang, H., Joshi, G., Poor, H.V.: Tackling the objective inconsistency problem in heterogeneous federated optimization. In: Proceedings of NeurIPS. (2020)
30. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms (2017)
31. Yu, F.X., Rawat, A.S., Menon, A., Kumar, S.: FedAwS: Federated learning with only positive labels. In: Proceedings of ICML. (2020)
32. Yuan, H., Ma, T.: Federated accelerated stochastic gradient descent. In: Proceedings of NeurIPS. vol. 33 (2020)
33. Yurochkin, M., Agarwal, M., Ghosh, S., Greenewald, K., Hoang, N., Khazaeni, Y.: Bayesian nonparametric federated learning of neural networks. In: Proceedings of ICML. (2019)
34. Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., Chandra, V.: Federated learning with non-iid data. ArXiv abs/1806.00582 (2018)
35. Zhu, H., Jin, Y.: Multi-objective evolutionary federated learning. IEEE Transactions on Neural Networks and Learning Systems 31(4), 1310–1322 (2020)